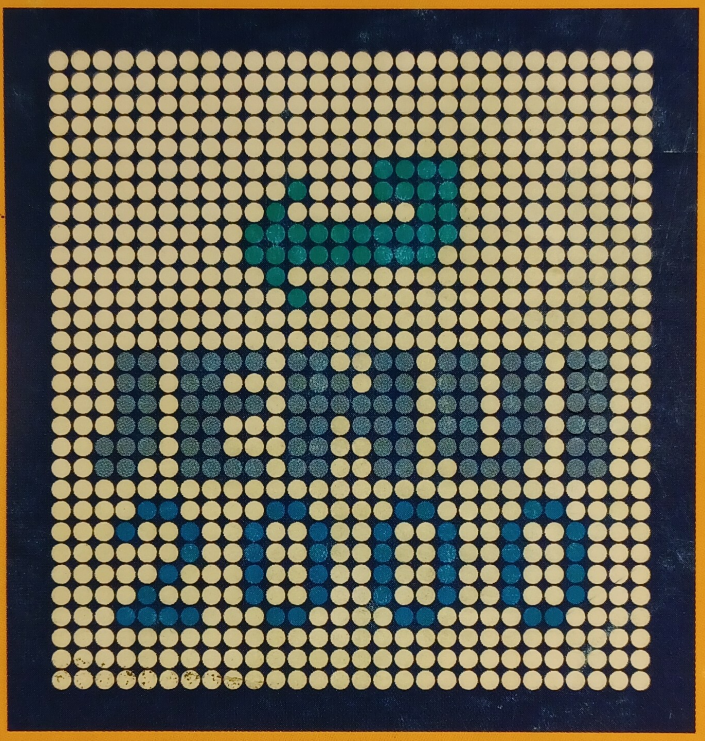
Jenui 2000 VI Jornadas sobre la Enseñanza Universitaria de la Informática



LIBRO DE COMUNICACIONES

UNIVERSIDAD DE ALCALÁ

Alcalá de Henares, 25 y 26 de Septiembre de 2000

© UNIVERSIDAD DE ALCALÁ

Edita: Servicio de Publicaciones
Universidad de Alcalá

Imprime: Manuel Ballesteros
Industrias Gráficas, S.L.

I.S.B.N.: 84-8138-409-7

Dep. Legal: M-38194-2000

VI Jornadas sobre la Enseñanza Universitaria de la Informática

(JENUI 2000)

LIBRO DE COMUNICACIONES.

Editores: Dra. Rosalía Peña.
Dr. Pedro Ramos.
D. Antonio J. de Vicente.

Escuela Politécnica
Universidad de Alcalá.
Alcalá de Henares, 25 y 26 de Septiembre de 2000.

INDICE

Autores..... pág. 2

Poencias..... pág. 8

Comité Organizador..... pág. 14

Sede..... pág. 14

Comié Científico..... pág. 15

Entidades Colaboradoras..... pág. 15

Presentación..... pág. 16

Área de Arquitectura de Computadores..... pág. 18

Área de Cartografía..... pág. 136

Informática en la E.M..... pág. 181

T.I y Gestión Empresarial..... pág. 195

Inteligencia Artificial..... pág. 253

Informática Industrial..... pág. 274

Ingeniería del Software..... pág. 311

Innovación..... pág. 381

Matemáticas..... pág. 484

Programación..... pág. 522

Telemática..... pág. 607

AUTORES

Grediaga,A. ..... pág. 19
Abad,P. ..... pág. 137
Abellanas,M. ..... pág. 430
Aguilar,S. ..... pág. 523
Aguilera,J.J. ..... pág. 298
Aguilera,J.J. ..... pág. 382
Albadalejo,J. ..... pág. 288
Albert,E. ..... pág. 557
Alfonseca,M. ..... pág. 388
Álvarez,J.Mª. ..... pág. 463
Arqués,P. ..... pág. 222
Balbastre,P. ..... pág. 304
Balsas,J.R. ..... pág. 298
Balsas,J.R. ..... pág. 382
Bernadó,E. ..... pág. 254
Bernadó,E. ..... pág. 261
Blanco,S. ..... pág. 217
Botti,V.J. ..... pág. 470
Brito,J. ..... pág. 268
Brotons,FJ ..... pág. 19
Burgos,J.M. ..... pág. 395
Cabañero,C.F. ..... pág. 228
Cabria,A. ..... pág. 174
Calvo,J.A. ..... pág. 312
Campos,C. ..... pág. 403
Camps,J. ..... pág. 254
Camps,J. ..... pág. 261
Cano,J.C. ..... pág. 60
Cano,J.C. ..... pág. 81
Cano,J.C. ..... pág. 625
Cañas,J.J. ..... pág. 418
Caracuel,J.E. ..... pág. 148
Cardenal,J. ..... pág. 148
Carrascosa,C. ..... pág. 470
Carrillo,J. ..... pág. 312
Carro,R.Mª. ..... pág. 388
Carsí,J.A. ..... pág. 353
Carvajal,P. ..... pág. 217
Colomina,O. ..... pág. 222
Compañ,P. ..... pág. 222
Córdoba,MªL. ..... pág. 109

Crespo,A. ..... pág. 304
Creus,P. ..... pág. 46
Cruz,L. ..... pág. 340
Cuenca,S.A. ..... pág. 39
Cuevas,G. ..... pág. 312
De las Heras,J.Mª. ..... pág. 409
de Vicente, A.J. ..... pág. 26
Del Val,J.L. ..... pág. 608
Delgado,A. ..... pág. 196
Delgado,J. ..... pág. 148
Dolado,J. ..... pág. 374
Domenech,A. ..... pág. 74
Domínguez,C. ..... pág. 288
Domínguez,J.J. ..... pág. 327
Escalona,Mª.J. ..... pág. 346
Escribano,C. ..... pág. 492
Escudero,F. ..... pág. 95
Esteso,E. ..... pág. 632
Fernández,J. ..... pág. 275
Fernández,M.J. ..... pág. 510
Ferrández,F.J. ..... pág. 282
Flich,J. ..... pág. 60
Flich,J. ..... pág. 81
Flich,J. ..... pág. 625
Fortés,J. ..... pág. 570
Fuster,A. ..... pág. 282
Galitó,M. ..... pág. 516
Gallego,J. ..... pág. 26
Gallego,R. ..... pág. 217
Galve,J. ..... pág. 395
Gámez,J.A. ..... pág. 503
García,C.R. ..... pág. 88
García,J.. ..... pág. 395
García,J.M. ..... pág. 282
García,M.E. ..... pág. 510
García,Mª I. ..... pág. 109
García,S. ..... pág. 26
Garell,J.M. ..... pág. 254
Garijo,M. ..... pág. 614
Garrell,J.M. ..... pág. 261
Garrido,R. ..... pág. 217
Gea,M. ..... pág. 418
Gibert,E. ..... pág. 46
Gil,J.A. ..... pág. 625
Gimbert,J. ..... pág. 516
Golobrades,E. ..... pág. 254

Golobrades,E. ..... pág. 261
Gómez,B. ..... pág. 95
Gómez,J.A. ..... pág. 115
Gonzáelz,A. ..... pág. 39
González,J.A. ..... pág. 340
González,M. ..... pág. 523
González,P. ..... pág. 340
Gould,M. ..... pág. 163
Grau,A. ..... pág. 275
Gregorio,C. ..... pág. 549
Guadalupe,M. ..... pág. 174
Guerrero,J.A. ..... pág. 503
Guerrero,J.M. ..... pág. 39
Gutierrez,V. ..... pág. 340
Hassan,H. ..... pág. 288
Hernández,P.V. ..... pág. 619
Hernández,V. ..... pág. 599
Herrero,J.R. ..... pág. 413
Ibarra,F. ..... pág. 19
Igelmo,A. ..... pág. 485
Iglesias,C.A. ..... pág. 614
Jacob,I. ..... pág. 563
Juan Lizandra,Mª.C. ..... pág. 584
Julian,V.J. ..... pág. 470
Lago,A. ..... pág. 608
Lemus,L. ..... pág. 102
Lemus,L. ..... pág. 632
Llana,L.F. ..... pág. 549
Llorá,X. ..... pág. 254
Llorá,X. ..... pág. 261
Llorens,F. ..... pág. 496
López,P. ..... pág. 53
López,P. ..... pág. 74
López,P. ..... pág. 129
Lorés,J. ..... pág. 418
Lozano,A. ..... pág. 143
Macias,E.M. ..... pág. 619
Macías,M. ..... pág. 423
Malumbres,M.P. ..... pág. 639
Marín,J. ..... pág. 477
Marqués,M. ..... pág. 403
Martel,E.A. ..... pág. 619
Martín,O. ..... pág. 346
Martínez,A. ..... pág. 26
Martinez,F.J. ..... pág. 268
Martínez,j.C. ..... pág. 53

Martínez,J.M. ..... pág. 288
Martínez,R ..... pág. 549
Mas,J. ..... pág. 437
Mateo Pla,M.A. ..... pág. 122
Mateo Pla,M.A. ..... pág. 444
Mejias,M. ..... pág. 346
Meseguer,J.M. ..... pág. 437
Miró,J. ..... pág. 451
Morales,D.G. ..... pág. 340
Moreno,J. ..... pág. 268
Moreno,J.M. ..... pág. 268
Moro,Q.I. ..... pág. 33
Navarro,J.J. ..... pág. 457
Nicolau,M. ..... pág. 46
Ojeda,C.N. ..... pág. 619
Oliver,J. ..... pág. 557
Oliver,J. ..... pág. 639
Oltra,J.V. ..... pág. 203
Oltra,J.V. ..... pág. 210
Ortega,M. ..... pág. 418
Palao,P. ..... pág. 549
Palomar,J. ..... pág. 155
Pareja,C. ..... pág. 549
Párraga,A. ..... pág. 19
Pavón,N. ..... pág. 463
Peña,R. ..... pág. 240
Perallos,A. ..... pág. 368
Perera,M. ..... pág. 592
Pérez Serrano,P.L. ..... pág. 360
Pérez, J.I. ..... pág. 26
Pérez,J.L. ..... pág. 599
Pérez,M.A. ..... pág. 234
Pertes,A. ..... pág. 288
Petit,S. ..... pág. 81
Piattini,M. ..... pág. 319
Pinar,Mª A. ..... pág. 102
Polo,A ..... pág. 530
Polo,M. ..... pág. 319
Pons,J. ..... pág. 122
Pons,J. ..... pág. 444
Posadas,J.L. ..... pág. 67
Posadas,J.L. ..... pág. 129
Poveda,J. ..... pág. 163
Poza,J.L. ..... pág. 74
Puerta,J.M. ..... pág. 503
Pujol,F.A. ..... pág. 282

Pulido,E. ..... pág. 388
Ramírez,C. ..... pág. 619
Ramos,I. ..... pág. 327
Ramos,I. ..... pág. 346
Real,J. ..... pág. 60
Rebollo,M. ..... pág. 470
Ribes,J.Mª. ..... pág. 95
Riquelme,J.C. ..... pág. 523
Robles,A. ..... pág. 67
Roda,J.L. ..... pág. 340
Rodríguez,G.M. ..... pág. 619
Rodríguez,J.J. ..... pág. 33
Rodríguez,J.M. ..... pág. 234
Rodríguez,J.P. ..... pág. 95
Rodríguez,P. ..... pág. 388
Rosso,P. ..... pág. 577
Rubio,MªL. ..... pág. 109
Ruiz Ben,E. ..... pág. 182
Ruiz,F. ..... pág. 319
Sáez,S. ..... pág. 129
Sahelices,B. ..... pág. 33
Sahuquillo,J. ..... pág. 60
Sahuquillo,J. ..... pág. 81
Sahuquillo,J. ..... pág. 625
Salamo,Mª. ..... pág. 254
Salamo,Mª. ..... pág. 261
Salas,M. ..... pág. 530
San Feliu,T. ..... pág. 312
Sánchez,F. ..... pág. 457
Sánchez,H. ..... pág. 516
Sánchez,J.L. ..... pág. 542
Sánchez,J.M. ..... pág. 115
Santos,J.M. ..... pág. 88
Sastre,M.A. ..... pág. 492
Segret,R. ..... pág. 228
Segura,J. ..... pág. 477
Sevilla,I ..... pág. 334
Sevilla,J. ..... pág. 163
Sutil,M. ..... pág. 395
Tendero,E. ..... pág. 39
Terrasa,S. ..... pág. 304
Tolmos,P. ..... pág. 217
Toralba,J.Mª. ..... pág. 247
Torres,M. ..... pág. 148
Tovar,E. ..... pág. 312
Tubella,J. ..... pág. 457

Tuya,J. ..... pág. 334
Ureña,L.A. ..... pág. 418
Usamentiaga,D. ..... pág. 334
Valero,M. ..... pág. 457
Valls,M. ..... pág. 516
Vega,M.A. ..... pág. 115
Velázquez,J.A. ..... pág. 549
Velez,M. ..... pág. 418
Vernet,D. ..... pág. 254
Vernet,D. ..... pág. 261
Vidal,A.M. ..... pág. 599
Vidal,G. ..... pág. 557
Virgós,F. ..... pág. 477

PONENCIAS

Área de Arquitectura.

- **Diseño de un procesador educativo. Implementación con componentes discretos.** A.Grediaga, F.J.Brotons, F.Ibarra y A.Párraga.
- **Técnicas pedagógicas y estudio de resultados para la enseñanza de MS-DOS y lenguaje ensamblador en primer curso de I.T Informática Gestión/Sistemas.** A.J de Vicente, J.Gallego, J.I.Pérez, A.Martínez y S.García.
- **Aplicaciones multiplataforma para el apoyo a la docencia universitaria de la Informática.** B.Sahelices, Q.I.Moro y J.J Rodríguez.
- **Desarrollo de proyectos Hardware de visión y procesamiento de la señal digital sobre FPGAs.** S.A. Cuenca, J.M. Guerrero. E. Tendero y A. González.
- **NNOS: un microkernel distribuido para docencia e investigación.** E. Gibert, P. Creus y M. Nicolau.
- **Simulador de un computador segmentado con especulación Hardware.** J.C. Martínez y P. López.
- **Pácticas de ensamblador.** J. Flich, J. Real, J.C. Cano y J. Sahuquilo.
- **Metodología para la enseñanza del procesador multiciclo.** J.L. Posadas y A. Robles.
- **WIN 8086: una herramienta para el análisis de la frecuencia de utilización de las instrucciones 8086.** J.L. Poza, A. Domenech y P. López.
- **Uso del lenguaje ensamblador para prácticas de aritmética.** J. Sahuquillo, J.C. Calvo, J. Flich y S. Petit.
- **Enseñanza de sistemas operativos con nachos: una experiencia española.** J.M. Santos y C.R. García.
- **Sistema de simulación para el diseño lógico y de computadores.** J.P. Rodríguez, B. Gómez, J.Mª. Ribes y F. Escudero.
- **Propuesta para la enseñanza de Arquitectura de computadores.** L. Lemus y Mª.A. Pinar.
- **CACHESIM: un simulador de cache configurable.** Mª.L. Córdoba y Mª. Luz Rubio.
- **Innovación docente de la asignatura de Arquitectura de computadores mediante el uso de simuladores.** M.A. Vega, J.M. Sánchez y J.A. Gómez.
- **Las asignaturas de Arquitectura de computadores en al titulación de Ingenieros de Telecomunicación.** M.A. Mateo Pla y J. Pons.
- **Simulador para la enseñanza de la planificación dinámica de instrucciones.** S.Sáez, J.L. Posadas y P. López.

ÁREA DE CARTOGRAFÍA.

- **Aplicaciones en entorno MATLAB para manipulación de ecuaciones resultantes de observaciones GPS.** P. Abad.
- **Contenidos de la asignatura "Fundamentos y lenguajes informáticos" y algunas experiencias en la aplicación de internet en su docencia.** A. Lozano.
- **Utilización de hojas de cálculo para la enseñanza de modelos matemáticos en Fotogrametría Analítica.** J. Cardeñal, J.E. Caracuel, J. Delgado y M. Torres.
- **Mejora de la enseñanza de algoritmos SIG en la E.T.S.I Geodésia, Cartografía y Fotogrametría de la U.P de Valencia.** J. Palomar.
- **Generación de modelos digitales de terreno a partir de mapas de elevación.** J. Poveda, J. Sevilla y M. Gould.
- **Edición automatizada del MTN 25: una experiencia docente de un proyecto cartográfico.** Mª. Guadalupe y A. Cabria.

ÁREA DE ENSEÑANZAS MEDIAS.

- **Las actitudes hacia el uso de ordenadores en la enseñanza secundaria en función del género.** E. Ruiz Ben.

ÁREA DE T.I Y GESTIÓN EMPRESARIAL.

- **La Informática de Gestión para empresarios como asignatura integrada en el proyecto Campus Extens de la UIB.** A. Delgado.
- **La legislación sobre seguridad de datos en Informática de Gestión. Ideas para una guía para la docencia.** J.V. Oltra.
- **Acercamiento del Derecho y la Deontología profesional al alumno de Ingeniería Técnica Informática. Un caso : aspectos legales de la Informática.** J.V. Oltra.
- **Una experiencia en la aplicación de la Informática a la enseñanza de Cálculo Financiero.** P. Carvajal, S. Blanco, R. Gallego, R. Garrido y P. Tolmos.
- **Evolución de contenidos en la asignatura Microinformática para economistas.** P. Arqués, O. Colomina y P. Compañ.
- **"Informática aplicada a la gestión", asignatura de los palnes de estudio de Ingeniería Técnica en Informática de Gestión y Ciencias Empresariales.** R. Segret y C.F. Cabañero.
- **Contenidos y objetivos de la asignatura de Informática aplicada a la gestión de la empresa.** J.M. rodríguez y M.A. pérez.
- **Enseñanza de la legislación para el profesional informático.** R. Peña.

- **Problemática de los contenidos interdisciplinares de tecnología de la formación y Admón. de dirección. El caso del presupuesto de proyectos de Software.** *J.M.Torralba.*

ÁREA DE INTELIGENCIA ARTIFICIAL.

- **Introducción a las técnicas de aprendizaje automático (parte I): estructuración de un caso docente.** E. Golobardes, E. Bernardo, X. Llorá, J.M. Garrell, Mª. Salamó. J. Camps y D. Vernet.
- **Introducción a las técnicas de aprendizaje automático (parte II).** E. Golobardes, E. Bernardo, X. Llorá, J.M. Garrell, Mª. Salamó, J. Camps y D. Vernet.
- **Desarrollo de una herramienta de ayuda a la enseñanza de heurísticas.** J. Brito, F.J. Martínez, J. Moreno y J.M. Moreno.

ÁREA DE INFORMÁTICA INDUSTRIAL.

- **La tecnología de computadores en las Ingenierías Informáticas.** J. Fernández y A. Grau.
- **La asignatura Adquisición y acondicionamiento de la señal en la Universidad de Alicante.** F.A. Pujol, F.J. Ferrández, A. Fuster y M. García.
- **Diseño de las páginas WEB de las asignaturas de Informática Industrial con capacidad para realizar adquisición y envío de datos a bajo nivel utilizando lenguaje JAVA.** H. Hassan, A. Pertes, J. Albadalejo, J.M. Martínez y C. Domínguez.
- **Aportaciones de la ingeniería de la programación al curriculum del Ingeniero TécnicoIndustrial.** J.R. Balsas y J.J. Aguilera.
- **La importancia del uso de procesos físicos reales en la enseñanza universitaria de la ingeniería.** S. Terrasa, P. Balbastre y A. Crespo.

ÁREA DE INGENIERÍA DEL SOFTWARE.

- **Análisis de experiencias en la impartición de Ingeniería Software I.** J.A. Calvo, J. Carrillo, G. Cuevas, T. San Feliu y E. yovar.
- **Utilización de SWEBOK y PMBOK para la organización de contenidos en la Escuela Superior de Informática de la UCLM.** F. Ruiz, M.Piattini y M.Polo.
- **Docencia en gestión y estimación de proyectos Software: un enfoque constructivista.** I. Ramos y J.J. Domínguez.
- **Auditoría informática del desarrollo de Aplicaciones.** D. Usamentiaga, I. Sevilla y J. Tuya.

- **EBD-ULL: una herramienta para la evaluación del rendimiento de sentencias SQL sobre diferentes sistemas gestores de Bases de datos.** L. Cruz, P. González, V. Gutierrez, J.A. González, D.G. Morales y J.L. Roda.
- **Un método docente para la realización de prácticas de Inf. Software.** H.Mejias, I. Ramos, O. Martín y Mª.J. Escalona.
- **Entornos de desarrollo de Software: contenidos y encuesta de satisfacción.** J.A. Carsí.
- **La asignatua de Informática aplicada a la gestión de empresas en diplomatura en Ciencias Empresariales.** P.L. Pérez Serrano.
- **Programa práctico de una asignatura de Ingeniería de Software disatribuido.** A. Perallos.
- **Los 330 créditos y la Ingeniería del Software.** J.J. Dolado.

ÁREA DE INNOVACIÓN.

- **Adaptación curricular de la asignatura "Teoría de Algoritmos".** J.J. Aguilera y J.R. Balsas.
- **Diseño de uncurso interactivo y adaptativo de procesadores de lenguajes.** M. alfonseca, R.Mª Carro, E. Pulido y P. Rodríguez.
- **Enseñanza de la programación basada en esquemas.** J. M. Burgos, J. Galve, J. García y M. Sutil.
- **Las estancias en prácticas de las titulaciones de Informática en la Universidad Jaume I de Castelló.** C. Campos y M. Marqués.
- **Utilización de tecnologías internet como recursos didácticos fundamentales para el desarrollo de las nuevas metodologías didácticas para la teleformación.** J.Mª. de las Heras.
- **Automatización de la recepción, clasificación y respuesta de prácticas y trabajos de asignaturas.** J.R. Herrero.
- **Una experiencia de diseño cooperativo de material docente virtual en Interacción Persona-Ordenador.** J.J. Cañas, M. Gea, J. Lorés, M. Ortega, L.A. Ureña y M. Velez.
- **La enseñanza de la Informática en otras titulaciones: el caso de las facultades de Ciencias del Deporte e INEFS.** M. Macias.
- **La Geometría en la Informática.** M. Abellanas.
- **Estudio sobre la heterogeneidad de conocimientos básicos en alumnos de primer curso de Universidades Politécnicas.** J. Mas y J.M Meseguer.
- **Experiencias de prácticas con PUM en la asignatura de Arquitectura de Computadores y Sist. Operativos 2.** M.A Mateo Pla y J. Pons.
- **Aprendizaje a través de la escritura: experiencias.** J. Miró.
- **Formulación de los objetivos de una asignatura en tres niveles jerárquicos.** J.J. Navarro, M. Valero, F. Sánchez y J. Tubella.
- **Aplicación de métodos hipermedia en la impartición de una asignatura de Fundamentos de Informática.** N. Pavón y J.Mª. Álvarez.
- **CALIOPE: una arquitectura para el aprendizaje autónomo colavorativo en entornos no presenciales.** M. Rebollo, V.J. Julian, C. Carrascosa y V.J. Botti.

- **Reingeniería docente basada en modelos de conocimiento orientada al concepto.** F. Virgós, J. Martín y J. Segura.

ÁREA DE MATEMÁTICAS.

- **Anotaciones sobre la importancia de la estadística en las ingenierías informáticas.** A. Igelmo.
- **Geometría Fractal.** C. Escribano y M.A. sastre.
- **Herramienta para la enseñanza de la Deducción Natural.** F. Llorens.
- **Una propuesta para las prácticas de la materia "Teoría de Autómatas y Lenguajes formales".** J.A. Gámez, J.A. Guerrero y J.M. Puerta.
- **Utilización del MATLAB para la realización de animaciones que simulen procesos interactivos para la resolución de ecuaciones no lineales.** M.J. Fernández y M.E. García.
- **Creación de una herramienta didáctica de apoyo al estudio de series y cálculo integral utilizando el mathematica.** M. Galitó, J. Gimbert, H. Sánchez y M. Valls.

ÁREA DE PROGRAMACIÓN.

- **Una propuesta para asignaturas de introducción a la programación.** S. Aguilar, M. González y J.C. Riquelme.
- **Un método general para asignaturas de Procesadores de Lenguajes.** A. Polo y M. Salas.
- **Una herramienta de apoyo a la evaluación de trabajos prácticos.** J.L. Sánchez.
- **EXERCITA: una base de datos de material para prácticas de Programación,** C. Gregorio, L.F. Llana, R. Martínez, P. Palao, C. Pareja y J.A Velázquez.
- **La enseñanza de JAVA en los estudios de Informática.** E. Albert, J. Oliver y G. Vidal.
- **Metodología y herramientas para el aprendizaje de la verificación y derivación formal de programas.** I. Jacob.
- **Una herramienta didáctica para la generación y utilización de analizadores LR directos e inversos.** J. Fortés.
- **Clases dirigidas a grupos especiales de alumnos para la asignatura "Algorítmos y Estructuras de datos 2".** P. Rosso.
- **Programación en Ingeniería Industrial.** Mª.C. Juan Lizandra.
- **Programación Literaria.** M. Perera.
- **Enseñanza de la programación paralela en la Universidad de Valencia.** J.L. Pérez, A.M. Vidal, V. Hernández.

ÁREA DE TELEMÁTICA.

- **Evaluación y diseño de redes: una asignatura cercana al mundo empresarial.** A. Lago y J.L. del Val.
- **Personalización de prácticas para el desarrollo de sistemas telemáticos avanzados.** C.A. Iglesias y M. Garijo.
- **Aproximación a la simulación de un sistema de telefonía móvil celular.** G.M. Rodríguez, C.N. Ojeda, E.A. Martel, E.A. Macias y C. Ramírez.
- **La asignatura de Redes de Área Local en la Ingeniería Informática de la UPV: teoría y prácticas.** J.C. Cano, J. Sahuquillo, J. Flich y J.A. Gil.
- **Una alternativa para motivar a los alumnos en la enseñanza de las Redes de Ordenadores.** L. Lemus y E. Esteso.
- **Herramienta para la experimentación de la codificación, compresión y transmisión de video.** J. Oliver, M.P. Malumbres.

COMITÉ ORGANIZADOR

El comité organizador está constituido por los siguientes miembros de la Universidad de Alcalá:

- Dra. Rosalía Peña. Titular de Escuela Universitaria del Dpto. de CC Sanitarias y Medico sociales.
- Dr. Pedro Ramos. Titular de Universidad del Dpto. de Matemáticas. Área: Matemática Aplicada.
- Dra. Ana Extremeño. Titular de Universidad del Área de Documentación.
- D. Antonio J. De Vicente. Titular Interino de Escuela Universitaria del Dpto. de Automática.
- Dña. María Guadalupe Rodríguez. Profesor Asociado del Dpto. de Matemáticas. Área: Cartografía, Geodesia y Fotogrametría.

SECRETÁRIA TÉCNICA

Fundación General Universidad de Alcalá – Departamento de Congresos.
P° de la Estación, 10
28807 Alcalá de Henares (Madrid).

Tlf: 91 880 29 11
Fax: 91 880 27 83
E-mail: congresos@fgua.es
http : //www.fgua.es./jenui2000

SEDE

Las jornadas se celebrarán en:

Escuela Politécnica de la Universidad de Alcalá.
Ctra. Madrid-Barcelona, Km. 33,600
Campus Universitario
28871 Alcalá de Henares (Madrid).

COMITÉ CIENTÍFICO.

- Arquitectura de Computadores. Dra. Rosa Mª. Badía.
- Cartografía, Geodesia y fotogrametría. Dr. Miguel Angel Bernabé Poveda.
- Informática en Enseñanza Media. Dña. Blanca Baños y dña. Carmen Arriero.
- TI y Gestión Empresarial. Dr. Ferran Virgós.
- Inteligencia Artificial. Dr. Josep Mª. Garrell.
- Ingeniería del Software. Dr. Edmundo Tovar.
- Informática Industrial. Dr. Iñaki Iñigo Ochoa de Chinchetru.
- Innovación docente y experiencias en la enseñanza de la Informática.Dr. Pedro Blesa y D. Antonio Martí Campoy.
- Matemáticas. Dr. Josep Miró.
- Programación. Dr. Joaquín Ezpeleta Mateo.
- Telemática. Dr. Gregorio Fernández Fernández

ENTIDADES COLABORADORAS.

- Universidad de Alcalá de Henares.
- Banco de Santander.
- A.L.I. (Asociación de Licenciados en Informática).
- VALMATICA, Servicios Informáticos, S.L.L.

PRESENTACIÓN.

Celebramos de nuevo las Jornadas sobre la Enseñanza Universitaria de la Informática, JENUI 2000, cumpliéndose este año su VI edición y demostrando así su total consolidación.

El principal objetivo de estas jornadas es compartir y debatir los contenidos de los programas y los métodos docentes utilizados en la impartición de las diferentes áreas tecnológicas tratadas en la enseñanza universitaria de la informática, para ello, en esta nueva edición, se han reunido noventa ponencias en once áreas de investigación:

- Innovación docente y experiencias en la enseñanza de la informática
- Programación
- Arquitectura de computadores
- Inteligencia artificial
- Ingeniería del software
- Telemática
- Matemáticas
- T.I. y gestión empresarial
- Informática industrial
- Cartografía, geodesia y fotogrametría
- Informática en enseñanza media

Es importante destacar la inclusión en esta edición de dos nuevas áreas: Cartografía, geodesia y fotogrametría, que se corresponde con una nueva carrera de segundo ciclo que está siendo implantada en varias universidades españolas, y que responde a una gran demanda en el mercado de nuevos productos como Sistemas de Información Geográfica e Informática en enseñanza media, con las que se pretende conocer la problemática en los institutos y la formación previa con la que llegan los alumnos a la Universidad.

También contamos en las jornadas con la conferencia invitada de D.Victor Castelo perteneciente a RedIris, que nos hablará sobre "Redes gigabit para la Educación y la Investigación". Por último el profesor Josep Miró nos presentará la edición JENUI 2001 que se celebrará en la Universitat de les Illes Balears.

Pretendemos que durante estos dos días en los que nos reunimos se mantenga el espíritu que ha guiado anteriores ediciones: un fuerte grado de integración de todos los participantes, la puesta en común de nuestros trabajos y la consolidación de la amistad que en muchos casos nos une y que en otros casos puede iniciarse dando lugar a futuras colaboraciones.

Queremos aprovechar esta presentación para agradecer la colaboración de las diferentes instituciones y empresas que han respaldado estas jornadas, tanto con su aportación económica, como con su respaldo humano y moral. Queremos también tener un reconocimiento especial para el esfuerzo realizado por el Comité Científico al efectuar la selección de las ponencias presentadas, que han superado con creces nuestras expectativas inciales y animar a todos aquellos compañeros que habéis participado en esta edición a que sigáis colaborando en futuras ediciones. Además queremos expresar muestro reconocimiento al Departamento de Congresos de la Fundación General de la Universidad de Alcalá al que hemos confiado la Secretaría Técnica.
No queremos dejar pasar por alto el hecho de que esta reunión se celebra en la Universidad y en la Ciudad de Alcalá de Henarés, declarada Patrimonio de la Humanidad, y que constituye un marco ideal para compartir conocimientos y el intercambio de ideas.

Por último, esperamos que nos disculpéis por aquellos errores que hayamos podido cometer en nuestra labor de organización.

Comité Organizador

PONENCIAS.

ÁREA DE ARQUITECTURA DE COMPUTADORES.

Responsable: Dra.Rosa Mª Badía
(Universidad Politécnica de Catalunya)
E-mail: rosab@ac.upc.es

DISEÑO DE UN PROCESADOR EDUCATIVO. IMPLEMENTACIÓN CON COMPONENTES DISCRETOS.

Ángel Grediaga[1], Francisco J. Brotons[2], Francisco Ibarra[3]

Antonio Párraga[4].

[1]*Universidad de Alicante*
e-mail: ***gredi@dtic.ua.es, brotons@dtic.ua.es, ibarra@dtic.ua.es, parraga@dtic.ua.es***

RESUMEN: En este articulo se describe un Procesador Educativo, desde su diseño con componentes discretos comerciales, atendiendo a los diferentes aspectos prácticos que influyen en el mismo, y su posterior simulación en un PC. Además se presenta el software que lo complementa elaborado para el estudio de la ruta de datos y la realización de prácticas en lenguaje ensamblador (MaNoTaS). El objetivo final es su utilización conjunta que permita comprender aspectos fundamentales relacionados con, la ejecución de las instrucciones, las señales de control necesarias, etc. y el resto de aspectos importantes que forman parte de un procesador real.

1.- INTRODUCCIÓN.

Las Estructuras del Computador son una parte importante en los curricular de las carreras de Informática [1]. Muchas son las aportaciones que han hecho otros autores, al respecto de la utilización de las herramientas de simulación para explicar la organización de los computadores e incluso su implementación con componentes discretos. Sin embargo, la potencia que ofrecen en estos momentos las herramientas CAD, permiten que los estudiantes estén familiarizados con el Diseño Electrónico Automático (EDA) y por lo tanto facilitar la comprensión de muchos conceptos. Siempre hemos creído que es muy conveniente que los alumnos entiendan los bloques funcionales del computador visualizando aunque sólo sea de forma virtual el movimiento de la información en su interior, aunque el objetivo final será conseguir que esa visión sea lo más práctica posible. Ya en 1987 Cutler y Eckert [2] describieron herramientas software que reunían estas características para el estudio de la arquitectura de los computadores las cuales mejoraron posteriormente [3]. Por esta época Devore y Hardin [4] diseñaron también un computador cuya misión era el aprendizaje de los aspectos software y hardware, con un procesador de 8 bits similar a los reales, y que utilizaron para introducir el lenguaje ensamblador, comprobando entre los estudiantes que el tiempo utilizado para la comprensión de estos tópicos fue sustancialmente inferior. Koneva y Denev [5] discuten el problema de la enseñanza de las arquitecturas reales y el lenguaje ensamblador, y describen un simulador de un hipotético procesador. El simulador fue diseñado para enseñar no solamente los conceptos del lenguaje ensamblador sino también la implementación y construcción de

lenguajes de alto nivel. En la misma línea podemos encontrar el trabajo de Barnett [6], Silvester [7] introduce en sus estudios la posibilidad de simulación de las diferentes estructuras que componen la máquina. Henderson [8] describe un paquete para el estudio de la arquitectura con diferentes niveles de detalle, que van desde la implementación electrónica hasta la simulación del procesador. Además, otros simuladores y emuladores han sido desarrollados por, Smith [9], Yen y Kim [10], Purvis, Yoho y Lamont [11], Tsuchiyama [12].

Nosotros realizamos mediante la utilización de herramientas CAD [13] [14] la implementación de un simulador software que permite aproximar al estudiante, mediante la animación y la utilización del lenguaje ensamblador, a la ruta de datos de un procesador. Además para que los alumnos puedan realizar variaciones al diseño del procesador, utilizaremos Pspice [15] para el diseño mediante componentes comerciales [16] [17] y utilizando las técnica top-down de diseño, del mismo procesador que se ha simulado mediante la herramienta software.

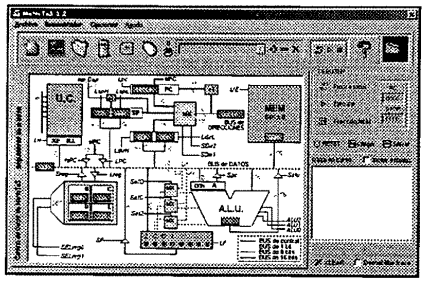


figura 1 Simulador de MaNoTaS

Llegados a este punto, los alumnos estarán familiarizados con la programación en ensamblador y la visualización de los datos moviéndose por la ruta de datos. Gracias al simulador, los alumnos empezarán a sentir la necesidad de conocer cómo se realiza el diseño de las diferentes estructuras que componen el computador. Éste será nuestro siguiente objetivo, para ello utilizaremos las técnicas de diseño convencionales de circuitos combinacionales y secuenciales.

2.- REPERTORIO DE INSTRUCCIONES.

Lo primero que debemos hacer es definir el repertorio de instrucciones y los modos de direccionamiento de las instrucciones que debe poseer la máquina, cuatro serán los modos con los que podremos acceder a los datos, inmediato, directo a memoria, directo a registro e indirecto a registro. El repertorio de instrucciones es el que se muestra en la tabla 1.

Tabla 1

Transferencia	Aritméticas	Lógicas	Control	E/S
LDA dir	ADD r1	ANA r1	JMP dir	IN #n
STA dir	ADI dato	ANI dato	JZ dir	OUT #n
LDAX	INR r1	ORA r1	JO dir	
STAX	DER r1	ORI dato	JC dir	
LFA	SUB r1	XRA r1	CALL dir	
SFA	SUI dato	XRI dato	RET	
MOV r1,r2	CMP r1	CMA	INT #n	
MVI $dato_8$,r1	CPI dato		IRET	
MVI $dato_{16}$,SP			CLI	
MVIH nom_eti,r1			STI	
PUSH r1			NOP	
POP r1				
PUSHF				
POPF				

3- UNIDAD ARITMÉTICO LÓGICA.

El diseño de esta unidad viene determinado por las operaciones que debe realizar. Aritméticas como la suma, resta, incrementos y decrementos, lógicas como la and, or y xor. Una vez elegidas las operaciones, debemos establecer el método de implementación, bien mediante operadores individuales o con un bloque multifuncional que se adapte a nuestras necesidades, la primera solución implica abundante circuitería, mientras que la segunda satisface los requerimientos buscados puesto que no se pierde de vista el concepto de unidad aritmético lógica y minimiza el diseño cumpliendo las especificaciones iniciales en cuanto a la utilización de componentes discretos, siendo esta segunda la solución adoptada. Necesitamos dos registros para realizar las operaciones, uno que es transparente al usuario, Registro Temporal (RT), y el otro el Acumulador (A) en el que además se almacenan los resultados de todas las operaciones. Como circuitos adicionales nos encontramos con el que se encarga de adaptar los códigos que se han establecido para las operaciones y los encargados de generar los señalizadores (flags) que se almacenarán en el Registro de Estado (RE).

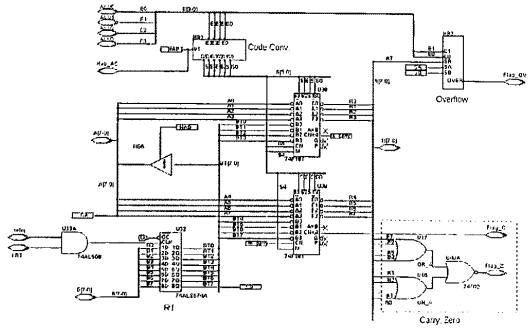


figura 2 Unidad Aritmético Lógica

4- BANCO DE REGISTROS.

Está formado por cuatro registros de propósito general (B, C, D y E) cuyo funcionamiento normal está asociado a la ALU y cuya finalidad es la de realización de operaciones de almacenamiento de variables de uso inmediato. El par de registros D-E tienen como función adicional la de almacenar una dirección para la utilización de las instrucciones con direccionamiento indirecto a registro de 16 bits (LDAX y STAX) Para la implementación de este Banco se han utilizado cuatro registros de 8 bits formados por biestables tipo D, así como un circuito decodificador que permitirá realizar las operaciones de carga y almacenamiento en cada uno de ellos.

5.- REGISTRO DE ESTADO.

Es fundamental conocer el resultado de la última operación realizada en la ALU, para ello implementaremos cada uno de los señalizadores (flags) mediante biestables independientes tipo D con precarga y reset, teniendo en cuenta que dichos señalizadores podrán ser alterados o bien por la propia ALU o por el contenido del Acumulador si se está ejecutando la instrucción SFA.

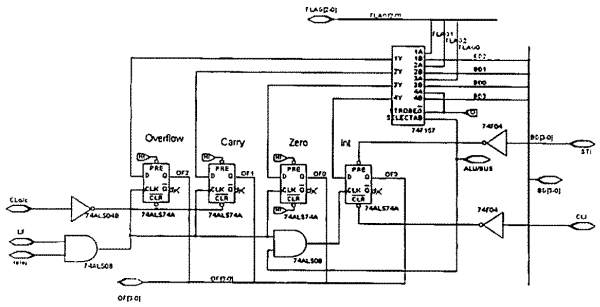


figura 3 Registro de Estado

6.- SELECCIÓN DE DIRECCIONES.

La memoria es única, y para acceder a su contenido es necesario proporcionar la dirección del dato que se quiere leer o escribir. Existen tres fuentes que pueden proporcionar dicha dirección, por un lado el Contador de Programa (PC), registro encargado de llevar la cuenta de las instrucciones del programa que se está ejecutando, el Puntero de la Pila (SP), registro que apunta a la primera dirección libre de una zona de memoria a la que se denomina Pila y que sirve para guardar las direcciones de retorno cuando se ha ejecutado una instrucción de llamada a subrutina (CALL) o de interrupción (INT), por último y teniendo en cuenta que se ha dotado al procesador de un direccionamiento indirecto a registro de 16 bits, utilizaremos el registro HL para almacenar temporalmente dicha dirección. La existencia de estas tres fuentes para la dirección, nos obliga a introducir un multiplexor 4 a 1 de 16 bits.

El Contador de programa, está formado por dos contadores de 8bits encadenados, con precarga y borrado. El registro HL esta formado por dos registros de 8 bits formados por biestables tipo D, iguales que los del Banco de Registros. El puntero de la Pila, es un contador de 16 bits ascendente/descendente con precarga, para evitar problemas debidos al encadenado de contadores a través de la patilla RC0 que poseen para tal efecto y que produciría valores no deseados, por ejemplo, RC0 se encuentra normalmente a nivel alto, cuando el contador está configurado en modo ascendente y sus salidas alcanzan el valor FFh, aquella pasa a nivel bajo, de forma que cuando el siguiente flanco de subida haga que las salidas pasen a 00h la salida RC0 vuelve a cambiar a nivel alto produciendo un flanco de subida en el segundo contador, algo similar, pero al revés ocurre cuando el funcionamiento es de descendente. Para evitar estos problemas hemos tenido que diseñar nuestro propio RC0.

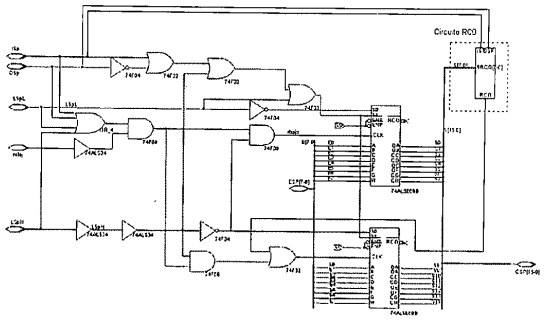


figura 4 Puntero de la Pila

7.- UNIDAD DE CONTROL.

La Unidad de Control que hemos diseñado, es lo suficientemente grande para descartar la implementación cableada. Puesto que hemos optado por una U.C. microprogramada partiremos de la información que nos proporciona los registros de instrucción (RI) y de estado (RE) y debemos generar todas las señales necesarias para el funcionamiento de la ruta de datos, teniendo en cuenta que hemos utilizado dispositivos activos por flanco de subida.La información procedente de los registros la llevamos hasta un Decodificador de Microinstrucciones (DM), que es el que se encargará de traducir el código de operación de cada instrucción en la dirección de inicio del microprograma correspondiente, para mayor simplicidad hemos hecho que las direcciones relacionadas con los saltos condicionales y la interrupción, sólo difieran en un bit. La dirección generada por DM deberá cargarse en el Contador de Microprograma (CM) e incrementarse con cada ciclo de reloj. Puesto que la ruta de datos funciona con flancos de subida, haremos que el CM funcione mediante flancos de bajada así las señales se generarán un semiperiodo antes de que se produzca el flanco por el cual son activas pudiéndose asumir sin dificultad los retardos de los componentes. La finalización de la ejecución de cada microprograma debe iniciar un nuevo "fetch" por lo que dentro de aquél debe estar incluida su dirección, además

existen muchas instrucciones que poseen fragmentos idénticos por lo que sería absurdo repetirlos y lo que haremos será que todas ellas compartan las mismas direcciones.
Puesto que disponemos de 33 señales de control más 6 internas de la propia U.C. los microprogramas se encuentran en cinco memorias EEPROM de 512 bytes.

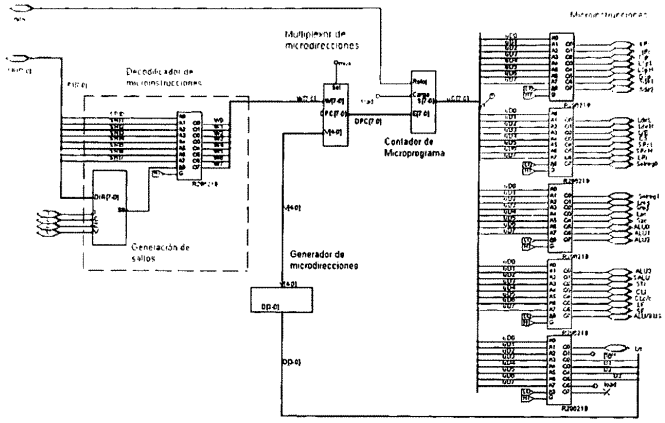


figura 5 Unidad de Control

Cuando se arranca el sistema todos los registros están a cero, por lo que en estas condiciones la U.C. carga el PC con la dirección que se encuentre en las posiciones 0000h y 0001h de la Memoria Principal, iniciando una fase de búsqueda, esto quiere decir que cuando se escriba un programa debemos tener la precaución de colocar en estas direcciones la dirección de inicio.

Teniendo en cuenta que tenemos 16 interrupciones vectorizadas, las 32 primeras posiciones de memoria deben estar reservadas. Si durante el programa no se modifica el contenido del vector 0, podremos utilizar la interrupción cero como "reset" software. Como ejemplo veamos un fragmento de la ejecución de un programa que incluye la instrucción LDAX y STA dir.

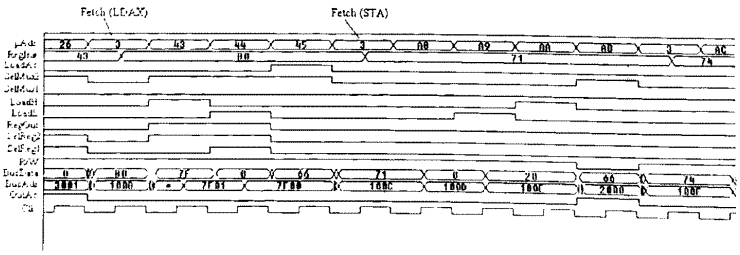


figura 6 Cronograma de las instrucciones LDAX y STA 2000

8.- REFERENCIAS.

[1] IEEE-CS Educational Actives Board. *The 1983 IEEE Computer Society model program in Computer Science and Engineering.* The Institute of Electrical and Electronic Engineers, Inc. 1983.

[2] M. Cutler and R.R. Eckert. "A microprogrammed Computer Simulator", *IEEE Trans. Educ., vol 30 pp 135-141. Aug 1987.*

[3] M. Cutler and R.R. Eckert. "Microprogrammed Computer Simulator Tools", *IEEE Trans. Educ., vol 33 pp 212-220. May 1990.*

[4] J.J. Devore and D.S. Hardin, "A computer design for introducing hardware and software concepts.", *IEEE Trans. Educ., vol E-30 nº 4 pp 219-226. May 1987.*

[5] L. Koneva and J. Denev, "EASY/VI – a new instructional computer", *ACM SIGCSE Bull. vol 22 nº 2 pp 55-58. June 1990.*

[6] B. L. Barnett, "A visual simulator for a simple machine and assembly language", *Proc. 26 th ACM SIGCSE Tech. Symp. Computer Science Education pp 233-278. 1997*

[7] P.P. Silvester, "Introducing computer structure by machine simulation", *IEEE Trans. Educ., vol 33 nº 4 pp 326-332.Nov 1990.*

[8] William D. Henderson, "Animated Models for Teaching aspects of Computer Systems Organization",*IEEE Trans. Educ., vol 37 pp 247-256. Aug 1994.*

[9] M. R. Smith, "A microprogramable microprocessor simulator and development system", *IEEE Trans. Educ., vol E-27 pp 93-100. May 1984.*

[10] R. Yen and Y. Kim, "Development and implementation of an educational simulator software package for specific microprogramming architecture", *IEEE Trans. Educ., vol E-29 pp 1-11. Feb 1986.*

[11] R. E. Purvis, R. D. Yoho, and B. Lamont, "MIME: An educational microprogrammable minicomputer emulator", *IEEE Trans. Educ., vol E-24 pp 257-261. Nov. 1981.*

[12] M. Tsuchiyama, "Development of an educational computer simulator", *Electronics and Communications in Japan Part 3, vol 74 nº 3 pp 25-35.*

[13] Douglas V. Hall. Teaching Design Methodology and "Industrial Strenght" EDA Tools in First-Term Freshman Digital Logic Course". *IEEE Trans. Educ.*, vol. 41 pp 45-49. Feb. 1998

[14] G. Puvvada and M.A. Breuer, "Teaching computer hardware design using commercial CAD tools". *IEEE Trans. Educ.*, vol. 36 pp 158-162. Feb. 1993

[15] MicroSim Pspice AD. Reference Manual v.7.1.1997

[16] Texas Instruments. Logic Selection Guide and DataBook. 1997

[17] Cypress Memory DataBook. 1997

[18] Stalling W. "Computer Organization and Architecture". *Prentice Hall.*1993

TÉCNICAS PEDAGÓGICAS Y ESTUDIO DE RESULTADOS PARA LA ENSEÑANZA DE MS-DOS Y LENGUAJE ENSAMBLADOR EN PRIMER CURSO DE I.T. INFORMÁTICA GESTIÓN / SISTEMAS.

Antonio J. de Vicente[1], José Gallego[2], Juan I. Pérez[3],

Agustín Martínez[4], Sara García[5]

Departamento de Automática. Universidad de Alcalá
[1]*e-mail: avicente@aut.alcala.es*
[2]*e-mail: jgallego@aut.alcala.es*
[3]*e-mail: nacho@aut.alcala.es*
[4]*e-mail: hellin@aut.alcala.es*
[5]*e-mail: sara@aut.alcala.es*

RESUMEN: En el presente artículo se recoge la experiencia docente en la enseñanza de *MS-DOS* y lenguaje ensamblador a alumnos de primer curso de las ingenierías técnicas de Informática de Gestión y de Sistemas, los aspectos pedagógicos y un estudio estadístico detallado de los resultados obtenidos.

1.- INTRODUCCIÓN.

Hoy en día los lenguajes de alto nivel, orientados a objetos o lenguajes visuales, separan cada vez más al programador del hardware real de la máquina. Con dichos lenguajes, los desarrollos son mucho más rápidos, y más baratos de diseñar y mantener, facilitando también el prototipado de cara a los clientes de las futuras aplicaciones.

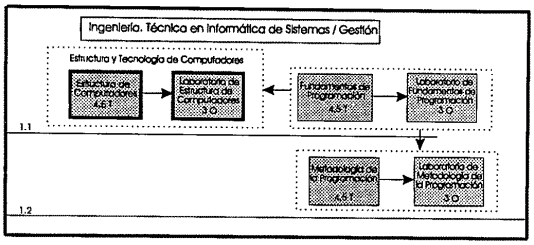
Los lenguajes visuales de programación y orientados a objetos tienen como característica principal la portabilidad de los programas de unas máquinas a otras, ya que ocultan o simplifican las características propias del hardware sobre el que se ejecuta el software final.

Cualquier lenguaje de alto nivel siempre necesitará compiladores, empleará librerías, etc. que requieren del lenguaje ensamblador. Además, existen aplicaciones que no se pueden programar empleando lenguajes de alto nivel, como por ejemplo las diseñadas para ser empleadas en sistemas microcontrolados. En estos sistemas se debe tener una visión muy cercana al hardware sobre el que se ejecutará la aplicación final, por lo que se tiene que acudir a lenguajes que permitan manejar los elementos hardware del sistema de forma directa. Aquí subyace el interés del lenguaje ensamblador.

El lenguaje ensamblador, además, proporciona al usuario / programador una visión clara de las particularidades del hardware sobre el que se ejecutará la aplicación. Esta característica hace que los programas desarrollados en lenguaje ensamblador no sean portables. Es la cercanía al hardware lo que hace que se utilice en la enseñanza de arquitectura.

La asignatura sobre la que se ha realizado el estudio, *Laboratorio de Estructura de Computadores*, es una asignatura obligatoria de primer curso de Informática. Los alumnos llegan a ella con pocos o ningún conocimiento sobre hardware de ordenadores o sobre lenguajes de programación, lo que complica aún más la enseñanza.

El plan de estudios actual no consigue solventar esta situación, pues como se muestra en el siguiente gráfico, la relación de asignaturas de *Programación* y de *Estructura de Computadores,* se imparten a la vez que dicho laboratorio, e incluso dándose el caso, como es el de *Metodología de Programación* de desarrollarse en el siguiente cuatrimestre.



2.- TEMARIO DE LA ASIGNATURA.

La asignatura de *Laboratorio de Estructura de Computadores* (B.O.E. 24-5-95) es una asignatura de primer curso y primer semestre.

El objetivo de la asignatura es dar a los alumnos unos conocimientos básicos sobre la estructura de un computador, mediante la explicación de un sistema operativo sencillo, como es el *MS-DOS*, basado en órdenes, y la programación en lenguaje ensamblador de un microprocesador comercial, el *i80x86,* (que pueden tener incluso en su casa).

Cada una de estos módulos se desarrolla en:

a) Conceptos de sistema operativo.

El sistema operativo empleado es el *MS-DOS*, pues introduce de forma sencilla las funciones de un sistema operativo. No obstante, se hace referencia a los nuevos sistemas operativos como Windows 98, Windows 2.000 y Linux, para enriquecer su formación en sistemas operativos.

Se le da al alumno una visión de cómo almacenar la información en archivos y directorios, cómo manejarse por medio de las órdenes de *MS-DOS* y cómo programar archivos de proceso por lotes que faciliten y mejoren las limitaciones de algunas órdenes.

b) Conceptos de arquitectura mediante el empleo del lenguaje ensamblador.

Es la parte más difícil de impartir, ya que los alumnos carecen en su mayoría de conocimientos de arquitectura de computadores por un lado, y por otro de lenguajes de programación.

Mediante el lenguaje ensamblador se consiguen los siguientes objetivos:

- se muestra el concepto de unidad aritmético-lógica y se enfatiza el sistema de representación de complemento a dos, puesto que el i80x86 opera matemáticamente en dicho formato;
- se les muestra también la estructura del i80x86, los registros que lo forman y las características de cada uno de ellos, registros generales, de segmento, punteros de pila, contador de programa, etc;
- se introduce el concepto de segmentación de memoria y como influye en los diferentes modos de direccionamiento a memoria;
- finalmente se les imparte un repertorio básico de instrucciones, el concepto de interrupciones software y algunas funciones con los que desarrollar las prácticas de la asignatura.

Al finalizar, el alumno habrá comprobado de forma práctica alguna de las características de una arquitectura *von Neumann*: unidad aritmético-lógica, sistemas de memoria, entrada salida, etc.

3.- PEDAGOGÍA EMPLEADA.

Durante los cursos **96-97, 97-98** y **98-99**, el método empleado fue la presentación mediante transparencias de cada uno de los temas y la explicación de todos aquellos conceptos necesarios para realizar con éxito cada una de las prácticas de laboratorio. A continuación, se realizaba una breve descripción de la práctica, y alguna indicación en cuanto a cómo utilizar los conceptos explicados para la realización de la misma.

Los alumnos, en grupos de dos, realizaban las prácticas, y documentaban el proceso con una memoria, que entregaban al profesor, y que era devuelta con indicaciones sobre errores, posibles mejoras, etc, de forma que le fuera útil para terminar de fijar los conocimientos, y además con vistas al examen.

En el curso **99-00** el método anterior se completó con una serie de actividades previas a cada práctica, que guiaban al alumno en la realización de las mismas y facilitaban la comprensión de la complejidad teórica de dichas prácticas, trasladando parte de los contenidos conceptuales al proceso de realización de actividades. De esta forma el alumno, en lugar de recibir y poner en práctica todos los conceptos en bloque, los iba descubriendo por sí mismo (de una forma tutelada) con lo que se le exigía una actitud más activa. Redundando en un mejor aprendizaje y asimilación.

4.- MÉTODOS DE EVALUACIÓN EMPLEADOS.

A lo largo de los últimos cuatro años, nos hemos planteado cómo evaluar una asignatura de laboratorio, qué peso dar a cada uno de los módulos. Es claro, que si a los alumnos se les exige la entrega de prácticas, ese trabajo debería ser tenido en cuenta en la calificación final de la asignatura.

Durante los cursos ***96-97 y 97-98***, el método de evaluación consistía en que los alumnos podían obtener un punto de la calificación final mediante la entrega y superación de las prácticas del laboratorio, pero finalmente, existía un examen, que debido al gran número de alumnos existentes en un curso de primero, debía ser por escrito, aportando así los nueve puntos restantes de la calificación final. No obstante, aunque el método parezca *duro*, se verá en el capítulo 5 que los resultados fueron buenos.

Durante el curso ***98-99***, nos planteamos darle un mayor valor a las prácticas de laboratorio, pudiendo obtener hasta tres puntos de la calificación final de la asignatura, aportando los siete puntos restantes un examen por escrito. Como se verá en el capítulo 5 los resultados fueron desastrosos.

Finalmente en el curso ***99-00***, se optó por que la evaluación fuese eminentemente práctica. Los alumnos podían obtener el 100 % de la nota final mediante la entrega y superación de las prácticas. Entre todas las prácticas, había algunas que debían ser superadas obligatoriamente, con calificación de *apta* o *no apta*. El resto de las prácticas contribuían a la calificación final valorándose la calidad del desarrollo de las mismas. La forma de superación de cada una de las prácticas era oral, en unos plazos fijados desde el comienzo del curso, y de manera individual por cada uno de los componentes del grupo de prácticas. Para aquellos alumnos que no superasen las prácticas, existía la posibilidad de un examen final en el laboratorio, para superar la asignatura. Como se en el capítulo 5, este método proporcionó muy buenos resultados académicos.

5.- ESTUDIO ESTADÍSTICO DE LOS RESULTADOS ACADÉMICOS.

Las tablas siguientes muestran los estudios estadísticos, ANOVA, test de Kruskal-Wallis y LSD para los resultados de las titulaciones de I.T. Informática de Gestión y de Sistemas respectivamente.

a) I.T. Informática de Gestión.

I.T. Informática de Gestión							
					Intervalo de confianza al 95%		
Curso	Pres./ No pres.	Media	Desviación típica	Coeficiente de simetría	Coeficiente de curtosis	Media	Desviación típica
96-97	**56** / 64	5,6	1,71	1,607	-0,811	[5,2 ; 6,1]	[1,44 ; 2,10]
97-98	**74** / 104	6,5	2,13	***-2,413***	0,65269	[6,0 ; 7,0]	[1,83 ; 2,54]
98-99	**58** / 134	4,5	2,44	0,349	-1,557	[3,9 ; 5,2]	[2,06 ; 2,99]
99-00	**138** / 176	6,2	1,10	***-11,692***	***21,058***	[6,0 ; 6,4]	[0,99 ; 1,25]

A continuación veamos los resultados al aplicar diferentes test:

Analisis de la varianza (ANOVA) al 95%					
Fuente	Suma de cuadrados	Df	Media de cuadrados	F-ratio	p-valor
Entre grupos	169,169	3	56,3897	17,88	0
Dentro de grupos	1018,47	323	3,15315		
Total (correlación)	1187,64	326			

Como el p-valor < 0,05 se puede afirmar con el 95% de confianza que existe una diferencia significativa entre las medias de los cuatro años.

Test de Kruskal-Wallis		
Curso	**Tamaño muestra**	**Rango medias**
96-97	56	143,7
97-98	74	197,5
98-99	59	105,6
99-00	138	179,2
Estadística del test 38,36 p-valor 2,36E-8		

Como p-valor < 0,05 se puede asegurar que existen diferencias entre las medias.

Test de Fisher de la menor diferencia entre medias (LSD) al 95%			
	Tamaño población	**Media**	**Grupos Homogéneos**
Curso 96-97	56	5,6	X
Curso 97-98	74	6,5	X
Curso 98-99	59	4,4	X
Curso 99-00	138	6,2	X
Contraste	**Diferencia**	**+/- límites**	
96-97 contra 97-98	**-0,9**	0,62	
96-97 contra 98-99	**1,2**	0,65	
96-97 contra 99-00	**-0,5**	0,55	
97-98 contra 98-99	**2,1**	0,61	
97-98 contra 99-00	0,3	0,50	
98-99 contra 99-00	**-1,7**	0,54	

A la vista de los resultados anteriores se puede ver que realmente sí existen diferencias entre las medias de los diferentes cursos.

b) I.T. Informática de Sistemas.

I.T. Informática de Sistemas							
					Intervalo de confianza al 95%		
Curso	**Tamaño población**	**Media**	**Desviación típica**	**Coeficiente de simetría**	**Coeficiente de curtosis**	**Media**	**Desviación típica**
96-97	**56** / 59	5,2	1,92	0,498	-0,377	[4,7 ; 5,8]	[1,61 ; 2,35]
97-98	**79** / 97	6,7	1,71	0,311	-1,063	[6,3 ; 7,1]	[1,48 ; 2,03]
98-99	**87** / 142	5,2	2,37	-0,97	-1,928	[4,6 ; 5,7]	[2,06 ; 2,79]
99-00	**143** / 171	6,4	0,98	***-6,857***	***16,005***	[6,2 ; 6,5]	[0,88 ; 1,11]

Análisis de la varianza (ANOVA) al 95%					
Fuente	**Suma de cuadrados**	**Df**	**Media de cuadrados**	**F-ratio**	**p-valor**
Entre grupos	150,129	3	50,0431	17,15	0
Dentro de grupos	1053,23	361	2,91753		
Total (correlación)	1203,36	364			

Como el p-valor < 0,05 se puede afirmar con el 95% de confianza que existe una diferencia significativa entre las medias de los cuatro años.

Test de Kruskal-Wallis		
Curso	Tamaño muestra	Rango medias
96-97	56	134,045
97-98	79	216,089
98-99	87	151,839
99-00	143	202,85
Estadística del test 32,7391 p-valor 3,65E-7		

Como p-valor < 0,05 se puede asegurar que existen diferencias entre las medias.

Test de Fisher de la menor diferencia entre medias (LSD) al 95%			
	Tam.Poblac	Media	Grupos Homogéneos
Curso 96-97	56	5,2	X
Curso 97-98	79	6,7	X
Curso 98-99	87	5,2	X
Curso 99-00	143	6,4	X
Contraste	Diferencia	+/- límites	
96-97 frente a 97-98	**-1,46**	0,59	
96-97 frente a 98-99	0,053	0,57	
96-97 frente a 99-00	**-1,14**	0,53	
97-98 frente a 98-99	**1,51**	0,52	
97-98 frente a 99-00	0,32	0,47	
98-99 frente a 99-00	**-1,20**	0,45	

A la vista de los resultados anteriores se puede ver que realmente sí que existe diferencia de medias entre diferentes cursos.

6.- CONCLUSIONES.

La asignatura de *Laboratorio de Estructura de Computadores* es de gran complejidad, ya que el alumno debe enfrentarse a ella sin conocimientos de hardware ni de programación y que deben ser impartidos a la par que la asignatura. De esta forma los profesores se encuentran con la difícil situación de explicar unos conceptos que deberían ser previos a la impartición de la asignatura, junto con los propios de la misma. La razón se encuentra en en el plan de estudios, puesto que dichos conceptos se imparten al final de asignaturas que comparten temporalidad con la de *Laboratorio de Estructura de Computadores*.

El estudio estadístico anterior pone en evidencia que el mejor de los métodos empleados para impartir la asignatura de *Laboratorio de Estructura de Computadores* ha sido el seguido durante el curso ***99-00***, puesto que aunque no existe una diferencia significativa en las medias con el curso 97-98, en comparación, en el año 99-00 el número de alumnos que superan las asignatura, se ha incrementado considerablemente.

Finalmente, debido a las actividades desarrolladas previa a las prácticas, a los profesores que hemos impartido la asignatura nos ha quedado patente el aumento del interés de los alumnos, y como consecuencia una mejora de las calificaciones.

7.- BIBLIOGRAFÍA.

F. Segovia Olmo y J. Beltrán Llera. El aula inteligente. Nuevo horizonte educativo. Editorial Espasa. 1998

F. Remiro Domínguez y A. Martín García. Microprocesadores: el 8088 / 86 Editorial Akal-Bibloteca tecnológica. 1993.

J. Beltrán de Heredia . Lenguaje ensamblador de los 80x86. Editorial Anaya-Multiemdia. 1996.

K. Jamsa. DOS manual de bolsillo. Ed Osborne-McGraw-Hill. 1.989

APLICACIONES MULTIPLATAFORMA PARA EL APOYO A LA DOCENCIA UNIVERSITARIA DE LA INFORMÁTICA[1].

Benjamín Sahelices Fernández, Q. Isaac Moro Sancho, Juán José Rodríguez Diez

Departamento de Informática. Universidad de Valladolid.
e-mail: {benja,isaac, juanjo}@infor.uva.es

RESUMEN: en este artículo se presenta un proyecto basado en el desarrollo de un repertorio de aplicaciones informáticas multiplataforma enfocadas sobre aspectos puntuales de asignaturas correspondientes a Ingenierías Técnicas en Informática. Para facilitar su utilización se acompañan de explicaciones teóricas y ejemplos insertados en documentos de tipo hipertexto accesibles a traves del protocolo HTTP. El objetivo es facilitar la comprensión de ciertos aspectos de diferentes asignaturas que tradicionalmente suponen una mayor dificultad para el alumno.

1.- INTRODUCCIÓN.

El principal objetivo de este proyecto es la creación de un conjunto de herramientas para mejorar el aprovechamiento del trabajo personal del alumno en aquellas asignaturas específicas de la enseñanza Universitaria de la Informática. La orientación seguida es la de complementar la información recibida en la sesión de aula. Para ello se le proporciona al alumno información básica que le debe servir como recordatorio de lo explicado en el aula y le permita utilizar las diferentes herramientas con el objeto de interactuar con ellas y visualizar sus principales características.

Una fórmula comúnmente aceptada para conseguir estos objetivos es la utilización de interfases gráficos que permitan la animación y la interacción del usuario (alumno), a los que se añaden los correspondientes contenidos teóricos adecuadamente estructurados, y en su caso, un mecanismo de autoevaluación a través de la generación de forma automática de nuevos ejemplos y casos que puedan ser resueltos de forma guiada por el usuario.

Un aspecto clave del proyecto es facilitar el acceso a sus contenidos a través de Internet, lo que permite la extensión de la tarea educativa a un ámbito que excede el de los límites del aula-biblioteca-laboratorio-edificio. Además, ahora se podría hablar de laboratorio virtual y de prácticas virtuales. Así, en una práctica virtual se podría realizar la manipulación de elementos que o bien por su elevado coste económico, o por su elevada complejidad requerirían unos

[1]El trabajo expuesto en este artículo ha sido financiado por la Consejería de Educación y Cultura de la Junta de Castilla y León a través del proyecto de investigación VA36/99 conjuntamente con el Departamento de Informática de la Universidad de Valladolid. Localización actual del proyecto: http://odra.dcs.fi.uva.es

conocimientos previos no necesariamente poseídos por el estudiante (por ejemplo conocimientos en microelectrónica digital), o bien simplemente la imposibilidad de efectuar experiencias que involucren elementos conceptuales (como pueden ser fuentes de información, entropías, canales de transmisión, ...).

2.- METODOLOGÍA.

La gran mayoría de los componentes de este proyecto son módulos de simulación sobre temas puntuales y especialmente conflictivos o confusos para el alumno [6] [7]. Los módulos deben contener una completa ayuda así como los fundamentos teóricos indispensables para que el alumno revise los conceptos necesarios para su utilización. Estos módulos deben ser altamente interactivos para permitir al alumno el diseño de sus propios experimentos y la constatación de las consecuencias de sus decisiones. El apoyo visual es considerado fundamental; el objetivo es la visualización paso a paso de procedimientos de cierta complejidad, con explicación de cada uno de los pasos.

Para facilitar el acceso del alumno, este proyecto toma la forma de página web [5]. Las diferentes aplicaciones se han desarrollado en lenguaje de programación independiente de la plataforma (Java [2]), lo cuál permite que, además de un acceso remoto, se pueda asegurar la ejecución bajo cualquier tipo de computador. La información del servidor web se organiza en áreas y dentro de cada área se ha desarrollado un número variable de subproyectos.

Las áreas incluidas son sistemas digitales, sistemas operativos y estructura de ordenadores, teoría de la información, tutoriales de programación, y estructuras de datos.

Dado el carácter genérico de cada una de las áreas, únicamente se desarrollan subproyectos puntuales para cada una. Cada subproyecto es enfocado sobre un aspecto concreto desarrollando tanto la teoría como ejemplos prácticos sobre el mismo. El número de posibles subproyectos es enorme lo cuál conlleva la siempre incompletitud del trabajo desarrollado. Sin embargo también permite que este proyecto crezca y mejore de forma contínua. De un modo general cada uno de los subproyectos desarrollados tiene los siguientes componentes:

- Apartados teóricos que permiten un repaso de los conceptos básicos. Estos apartados están organizados con criterios didácticos de forma que se realice un repaso general y ordenado de la teoría. En cada apartado puede aparecer una pequeña aplicación preconfigurada para mostrar gráficamente lo explicado.

- Herramienta completa de simulación que permita al alumno configurar sus propios ejemplos y experimentar con ellos. Esta herramienta permite al alumno modificar los modelos y comprobar las consecuencias de dichas modificaciones.

- Generación automática de problemas que el alumno debe resolver. Posteriormente la aplicación indica la corrección de las soluciones. No se pretende evaluar al alumno, únicamente que éste practique y se asegure de la correcta comprensión del tema.

- Bibliografía clásica y enlaces a temas relacionados. Naturalmente, dado que una de las premisas impuestas va a ser la accesibilidad vía red, sería muy deseable abrir la posibilidad de uso de otros recursos disponibles en Internet, como pueden ser artículos, documentos, u otras herramientas.

Hasta este momento se han completado los siguientes subproyectos. simplificación de funciones lógicas para el *área de sistemas digitales*. planificación de procesos, gestión de memoria virtual, gestión de memoria cache y unidad aritmético-lógica entera para el *área de sistemas operativos y estructura de ordenadores*. codificación de la información en el *área de teoría de la información*. tutoriales de Java y de OpenGL en el *área de tutoriales de programación*. montículos y tablas de dispersión para el *área de estructuras de datos*.

Dado que estas herramientas van a estar a la disposición del público en general, y especialmente a los alumnos de las asignaturas de Informática en un nivel universitario, resulta muy importante conocer la opinión del usuario. Para ello se están desarrollando encuestas que serán insertadas en la página web. En ellas el alumno expondrá sus opiniones y valoración general del proyecto así como la valoración particular de cada subproyecto. Es esta la mejor forma de conocer la utilidad real del trabajo realizado.

3.- SISTEMAS OPERATIVOS Y ESTRUCTURA DE ORDENADORES

A modo de ejemplo se va a explicar en este apartado el trabajo realizado en las *áreas de sistemas operativos y estructura de ordenadores*. Estas dos áreas se asocian con las asignaturas de sistemas operativos y ampliación de sistemas operativos así como las de estructura y tecnología de computadores I y II de las carreras de Ingeniería Técnica Informática de Sistemas y de Gestión [3] [4]. El objetivo de estas asignaturas es el de proporcionar la base de conocimiento acerca de lo que es un computador, del funcionamiento de sus principales componentes y del funcionamiento básico y servicios ofrecidos por el sistema operativo [1]. Es por ello que el conjunto de subproyectos que se han desarrollado pretenden ayudar a conseguir los objetivos de las asignaturas, no tratándose en ningún caso de prácticas de tipo profesional, es decir, aquellas que permiten al alumno adquirir habilidad en el manejo de herramientas o lenguajes de programación con el objeto de aplicar dichas habilidades en el desarrollo de otros temas. Hasta este momento se han completado cuatro subproyectos que se detallan a continuación.

a) Memorias caché.

El objetivo fundamental de este subproyecto es explicar el funcionamiento básico de las memorias caché en su interacción con la memoria principal y la CPU. Por lo tanto está orientado especialmente a la descripción de las diferentes funciones de mapa ya que es éste el punto inicial que sirve al alumno para comprender la forma de almacenamiento y la utilidad de este tipo de memorias. También se describen de forma completa diferentes políticas de reemplazo así como de extracción y actualización de la memoria principal.

El subproyecto está dividido en diferentes apartados teóricos. En cada uno se explica un tema de los mencionados previamente. Para completar la explicación de cada tema, se incluye un ejemplo gráfico. El ejemplo es un programa de simulación preconfigurado que permite examinar cada una de las etapas por las que atraviesa el acceso a la memoria cache. En cada etapa se proporciona una breve explicación de lo que el alumno está viendo, así como los resultados de dicha etapa. Por ejemplo, es posible ver los valores de las etiquetas asociadas a cada bloque o línea caché, los valores de los contadores LRU para el reemplazo, los bloques libres o usados, entre otras cosas. Finalmente existe un programa de simulación completamente configurable por el alumno, el cuál podrá elegir el tamaño de la memoria caché, el de la memoria principal, la política de lectura, la de escritura, la de reemplazo, la función de mapa, el tamaño de bloque y el modo de generación de las direcciones desde la CPU.

b) Unidad aritmético-lógica entera.

Este subproyecto tiene como objetivo el estudio a nivel de puerta lógica de las operaciones básicas con datos de tipo entero en una unidad aritmético-lógica (ALU) sencilla. Las operaciones estudiadas son las de un sólo operando y las sumas. En el primer grupo se analizan las operaciones de carga, puesta a cero, complemento, incremento, decremento, desplazamiento y rotación. En el segundo grupo son abordados diferentes tipos de sumadores, desde los más sencillos de tipo serie o paralelo, hasta aquellos que permiten una mayor velocidad de suma como los de llevadas anticipadas (CLA), selección de llevadas y ahorro de llevadas. Todo ello debe permitir al alumno comprender de forma más concreta el funcionamiento interno de las ALU.

Para cada operación se realiza una descripción teórica, apoyada con gráficos y tablas, de su funcionamiento a nivel de puerta lógica. Esta explicación debe servir como refresco de la exposición del tema realizada en el aula y como introducción al manejo de los módulos de simulación. Hay dos módulos, uno dedicado a las operaciones con un solo operando y otro para las sumas. Ambos representan gráficamente y a nivel de puerta lógica el circuito a estudiar, sus señales de control y los valores binarios en cada instante de tiempo. Permiten la realización de las operaciones paso a paso y proporcionan una explicación de lo que representa cada paso. Permiten al alumno estudiar cómo se propagan las señales binarias por los circuitos, distinguir todas las etapas intermedias y experimentar con diferentes valores.

c) Planificadores de CPU

Este subproyecto tiene como objetivo la gestión que el sistema operativo realiza de la CPU como un recurso más del sistema informático. Se centra en el estudio de algoritmos de planificación a corto plazo, tanto de tipo monocola como multicola. El estudio de estos algoritmos permite al alumno entender el tratamiento interno que el sistema operativo realiza sobre los procesos, y por lo tanto ayuda a comprender la idea de la multiprogramación. Este concepto es básico para el estudio de la programación concurrente y, por extensión, la paralela. Se trata de temas que tradicionalmente resultan de difícil comprensión en las asignaturas de introducción a los sistemas operativos.

El subproyecto realiza una introducción teórica a los conceptos básicos de procesos, estados y cambios de contexto, así como de los tipos de planificación. A continuación describe los diferentes algoritmos de planificación junto con algunas técnicas de evaluación de prestaciones. Los algoritmos descritos son el FCFS, Round-Robin, SJF, SRT, HRRN, cola múltiples, colas múltiples con realimentación y colas múltiples con prioridades. Finalmente hay dos módulos que permiten al alumno configurar el número de procesos junto con la generación estadística o manual de sus instantes de llegada y ráfagas de CPU y E/S. Una vez seleccionado el algoritmo, se realiza una ejecución paso a paso que permite visualizar el estado de las colas en cada instante, la selección del siguiente proceso o la evolución entre las diferentes colas entre otras cosas.

d) Paginación bajo demanda.

La gestión de memoria virtual mediante paginación bajo demanda es el ejemplo clásico que sirve para explicar la gestión de memoria en los sistemas operativos. Uno de los aspectos que dificultan su comprensión es que está intimamente relacionada con el hardware de gestión de direcciones de la CPU. Se trata por lo tanto de un tema en el cuál es preciso abordar conceptos de hardware y de software para realizar una explicación completa.

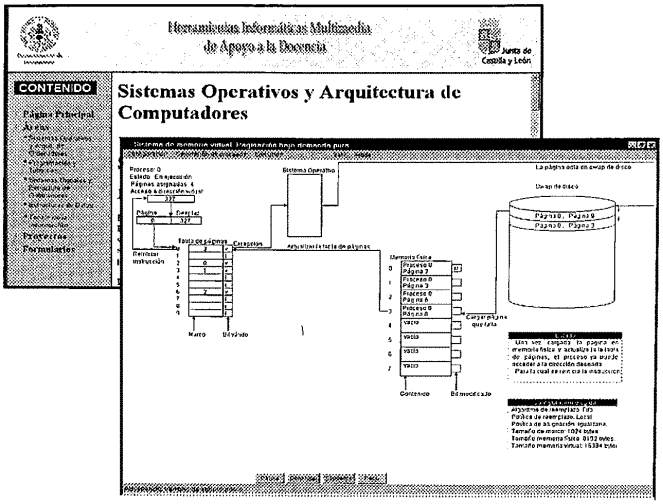


Ilustración 1 Módulo de paginación bajo demanda.

El subproyecto realiza un repaso teórico sobre los conceptos básicos de gestión de memoria, incluyendo algunos ejemplos reales, como la gestión en el sistema operativo Linux. Su herramienta de simulación (ver ilustración 1) permite configurar el algoritmo y la política de reemplazo en la gestión de la memoria, el tamaño de página y de marco y los tamaños de memoria física y virtual. Permite también generar de forma aleatoria o directa las direcciones generadas por los diferentes procesos que intervendrán en la simulación. Ésta es realizada paso a paso con visualización de los dispositivos hardware involucrados en cada uno (CPU, tabla de páginas, memoria física, área de intercambio) y con explicación de lo ocurrido en cada paso. El alumno visualiza la evolución de la memoria física, el proceso propietario de cada una de las páginas, el estado de cada tabla de páginas de cada proceso, etc.

4.- CONCLUSIONES

Las conclusiones a las que se pueda llegar en estos momentos han de ser forzosamente provisionales, ya que el proyecto aún sigue en marcha. Tal vez uno de los puntos que más convenga destacar es que el proyecto no dará lugar a un producto cerrado, sino que siempre va a poder ser actualizado, ya sea por la agregación de nuevos subproyectos en las áreas existentes, o la incorporación de áreas nuevas con sus respectivos subproyectos. El hecho de que todos ellos se encuentren centralizados en una única página web facilita el acceso del alumno y el mantenimiento tanto del software como de la documentación.

En el aspecto didáctico este proyecto trata de fomentar el trabajo personal del alumno haciéndolo un poco más atractivo. Para conseguirlo se ha pretendido que el alumno sea parte activa, además de receptiva de información, mediante aplicaciones de simulación altamente interactivas. Actualmente el número total de subproyectos es de diez, esperando que al final del trabajo haya subido hasta la veintena. Se consegurá así tratar un número representativo de temas y será en ese momento en el que se podrá establecer hasta qué punto se ha facilitado al alumno la comprensión de las materias tratadas.

BIBLIOGRAFÍA

The Joint Task Force on Computing Curricula, "Computing Curricula 2001", IEEE Computer Society. Association for Computing Machinery, marzo 2000.

J. Bishop, "JAVA. Fundamentos de Programación. Segunda Edición.", Addison-Wesley, 1999.

A.Silberschatz, P.B.Galvin, "Operating System Concepts. Fourth Edition.", Addison-Wesley, 1994.

W.Stallings, "Organización y Arquitectura de Computadores. Diseño para Optimizar Prestaciones. Cuarta Edición.", Prentice-Hall, 1996.

I.S.Graham, "HTML Sourcebook: A Complete Guide to HTML 3.0", John Wiley & Sons, 1996.

W.Pierson, S.H.Rodger, "Web-based Animation of Data Structrures Using JAWAA", Twenty-ninth SIGCSE Technical Symposium on Computer Science Education, p. 267-271, 1998.

S.H.Rodger, "Integrating Animations into Courses", ACM SIGCSE/SIGCUE Conference on Integrating Technology in Computer Science Education, Barcelona, Spain, p. 72-74, 1996

DESARROLLO DE PROYECTOS HARDWARE DE VISIÓN Y PROCESAMIENTO DE SEÑAL DIGITAL SOBRE FPGAs

Sergio A. Cuenca Asensi, Jose M. Guerrero Romero, Emilio Tendero Esteve, Abel Gonzalez Onrubia

Universidad de Alicante, Escuela Politécnica Superior, Dpt. Tecnología Informática y Computación.
e-mail: sergio@dtic.ua.es

RESUMEN: En el presente trabajo se exponen una serie de herramientas y metodologías para la implementación en hardware de algoritmos relacionados con visión y procesamiento digital de la señal. Este tipo de prácticas resultan altamente motivadoras tanto en las asignaturas de visión y procesamiento de señal como en las relacionadas con el diseño de arquitecturas avanzadas, diseño VLSI o lenguajes de descripción hardware. Se proponen los dispositivos reconfigurables (FPGAs) como herramienta principal de implementación. Gracias a su gran versatilidad y facilidad de uso, esta tecnología esta desplazando cada vez más a los tradicionales ASIC, tanto en la industria como en los temarios de las asignaturas relacionadas.

1.- INTRODUCCIÓN.

Las tecnologías reconfigurables (principalmente FPGAs y CPLDs) se están convirtiendo en una alternativa real a las implementaciones tradicionales de algoritmos computacionalmente intensivos. En algunos sistemas las FPGAs pueden sustituir eficientemente a microprocesadores de propósito general o procesadores DSP [1], en otros la lógica programable trabaja junto con estos a modo de coprocesador liberándolos de ciertas tareas masivamente paralelas para que puedan atender otras tareas de control [2]. Por otro lado la tendencia a la baja del precio de estos dispositivos así como la aparición de nuevas familias de dispositivos de bajo coste (p. ej: serie SpartanII de Xilinx), ha convertido a las FPGAs en una seria alternativa al tradicional diseño de circuitos full-custom y semi-custom.

Los estudios de informática no son ajenos a esta tendencia de la industria, y en los últimos años se están incorporando estas tecnologías a las prácticas de muchas asignaturas [3] relacionadas con arquitecturas avanzadas, diseño de circuitos, visión o procesamiento digital de la señal. Otro factor que juega a favor es el hecho de que las herramientas de trabajo y las metodologías son cada vez más parecidas a las utilizadas en el diseño de ASIC, por lo que las asignaturas de diseño VLSI y lenguajes de descripción hardware (HDL) también empiezan a incorporarlas a sus temarios [4].

En este trabajo se describen una serie de herramientas y metodologías para el diseño de prácticas y pequeños proyectos hardware relacionados con visión y procesamiento de señal. La implementación de estos proyectos, mediante lógica reconfigurable, generalmente queda vedada a la mayor parte de los alumnos, ya que supone el empleo de costosas tarjetas tipo PCI, cuyo uso suele reservarse a la investigación (por ejemplo Aristotle de Mirotech [5], o RC1000-PP de Embedded Solutions [6], con precios por encima de los 2.500$). Gracias a la utilización de las herramientas que proponemos, este tipo de proyectos se pueden realizar utilizando tarjetas de bajo coste lo que supone una alternativa realmente asequible, máxime pensando en un laboratorio de 10 a 20 puestos. En nuestro caso disponemos de la placa XS40 de la compañía Xess [7] (<100$), que actualmente se utiliza en la asignatura de Diseño de Circuitos Asistido por Computador en los estudios de informática de la Universidad de Alicante.

Con el desarrollo de estas herramientas se pretende pues, aumentar las posibilidades de esta plataforma en varios sentidos. Primero, suministrando un interfaz amigable que permita de forma sencilla y transparente llevar a cabo las tareas de configuración y comunicación. En segundo facilitando el control, la depuración y la verificación de los algoritmos desde el host. Y por último proporcionando una librería de controladores hardware que realice la comunicación entre los tres elementos básicos de todo el sistema, esto es, el host, la memoria de la tarjeta y el dispositivo reconfigurable. De esta forma el alumno podrá concentrarse en el diseño del núcleo del algoritmo, ahorrará tiempo en el desarrollo y se ampliarán notablemente las posibilidades de éxito de los proyectos.

2.- EL ENTORNO DE DESARROLLO.

La placa XS40 está basada en la FPGA XC4010 de Xilinx [8] y el microcontrolador 8031 de Intel. Dispone además de una pequeña memoria SRAM de 32Kx8, accesible tanto desde la FPGA como desde el micro. Su arquitectura es muy simple como muestra la figura 1; la mayoría de los pines de los puertos P1 y P3 del micro están conectados a la FPGA y pueden ser utilizados como entradas/salidas de propósito general. La placa incluye un oscilador de 12MHz conectado directamente a una entrada de la FPGA que controla a su vez la señal de reloj que recibirá la patilla XTAL1 del micro. Además dispone de un display de 7 segmentos y un conector VGA accesibles directamente desde la FPGA. La mayor parte de las patillas de la FPGA están accesibles mediante un par de arrays de pines en la parte inferior de la placa, por lo que resulta relativamente sencillo la conexión de circuitería externa.

La comunicación con el host y la configuración de la placa se realiza a través del puerto paralelo. Desde el host se pueden enviar datos utilizando los ocho bits del registro de datos (D0..D7) y cuatro bits del registro de control (C0..C3) de este puerto. La comunicación desde la placa se realizará utilizando únicamente cuatro bits del registro de estado (S3..S6).

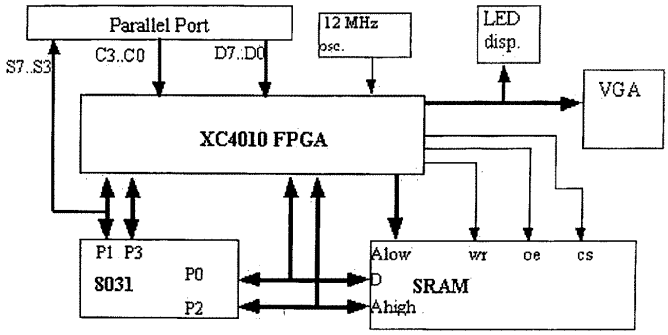


fig 1. Arquitectura de la placa XS40.

3.- DESARROLLO DE UN INTERFAZ DE PROTOTIPADO .

El objetivo principal de este interfaz es el prototipado rápido y la verificación funcional de los algoritmos mapeados en el hardware. Para conseguir esto las distintas tareas se reparten como se detalla a continuación. El host se encarga, por un lado, de las tareas de control y configuración del hardware, y por otro de la adquisición de los datos (imágenes y/o señales) y su visualización para la verificación funcional del procesamiento. Estas imágenes y/o señales pueden estar almacenadas en la memoria del host como ficheros de distintos formatos (bmp, tiff, pgm, wav,...). La FPGA, por su parte, se encarga del procesamiento del algoritmo utilizando los datos que le va transfiriendo el host o los datos almacenados previamente en la memoria SRAM de la placa. Conforme se van procesando estos, se envían de vuelta los resultados o bien se almacenan en la SRAM para su posterior volcado y visualización en el host. Esta metodología de trabajo permite al alumno concentrarse en el diseño y mapeado del algoritmo dejando para una etapa posterior opcional la incorporación de circuitería externa que permita la adquisición de los datos.

En esta línea de trabajo, la utilización del puerto paralelo para la comunicación con la placa supone una importante limitación a la hora de implementar aplicaciones con cierto grado de complejidad y en las que se necesite un trasiego masivo de datos entre el host y la FPGA. Esta limitación podría subsanarse utilizando el modo mejorado del puerto paralelo (EPP), sin embargo las líneas de datos D0 y D1 (fig.1) están conectadas a sendos inversores, lo que hace imposible su utilización en modo bidireccional. Es por tanto imprescindible desarrollar un interfaz a medida que permita la transferencia eficiente de datos en ambos sentidos. Este interfaz constará de una parte software, que implementa el protocolo de comunicación correspondiente al host, y una parte hardware que implementa la parte del protocolo correspondiente a la XS40.

a) Software

Se han diseñado una serie de funciones empaquetadas en una librería dinámica (DLL) que permite, mediante simples llamadas desde un programa C, realizar las principales tareas de comunicación bidireccional con la placa:

- test de los componentes.
- configuración de la FPGA.
- Control del procesamiento de la FPGA.
- Escritura/Lectura en la memoria SRAM de la XS40.
- Transferencia de bloques de datos hacia/desde la XS40.

Mediante esta librería el usuario puede crear un entorno amigable de comunicación e intercambio de datos con la placa.

b) Hardware

En este caso se trata de una librería de controladores hardware escrita en lenguaje VHDL, que permite fácilmente incorporar a los algoritmos los protocolos de comunicación y otras funcionalidades adicionales. Estos módulos ocupan muy pocos recursos de la FPGA (entre el 5% y 15% dependiendo de su complejidad), por lo que la funcionalidad de esta apenas se ve mermada.

Existen dos tipos de controladores (MC):

- Los "básicos" que implementan el protocolo mínimo de comunicación que permite controlar el procesamiento de la FPGA y la escritura y lectura de la memoria.
- Los de tipo "scan" que añaden a las funcionalidades básicas la posibilidad de controlar de forma transparente los accesos a la memoria SRAM siguiendo patrones típicos de los procesamientos de imagen y señal (acceso por filas, columnas, ventanas 3x3, etc..). Estos controladores permiten además trabajar con diferentes tamaños y resoluciones de imágenes y/o señales.

Utilizando esta librería, el usuario incorpora al núcleo del algoritmo o módulo de procesamiento (MP) el controlador apropiado y crea un circuito ("XS40 Core") totalmente preparado para comunicarse con el host (fig.2). Esta conexión entre el MC y el MP se lleva a cabo mediante una serie de líneas de control (control "residual") que especifican el tipo de operación a realizar: procesamiento, espera, lectura, escritura, etc... Este esquema de trabajo soporta cualquier tipo de arquitectura para la implementación del MP (multiciclo, segmentada, etc...).

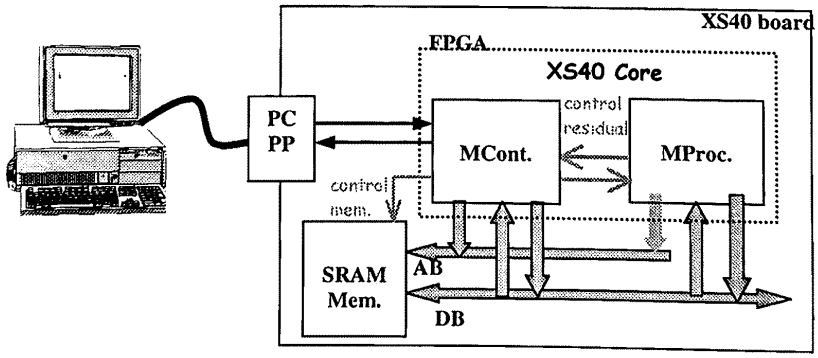


fig. 2. Esquema de trabajo.

4.- METODOLOGÍA Y FLUJO DE DISEÑO

En la figura 3 se puede apreciar el flujo de diseño a seguir para el desarrollo de un proyecto. Por un lado contamos con el software Foundation de Xilinx, que consta de una serie de herramientas que permiten realizar el ciclo completo de diseño de la FPGA. Incorpora, además de la típica captura de esquemáticos, varios de los lenguajes de descripción hardware más extendidos (VHDL, Verilog y Abel). Utilizando cualquiera de estos se lleva a cabo la entrada del circuito que implementará el algoritmo. A continuación se incorpora uno de los controladores (especificado en VHDL) y se llevan a cabo las simulaciones funcionales y temporales del circuito resultante. Una vez depurado el diseño se pasa a la fase de implementación (colocación, encaminamiento, etc...) que produce un fichero de configuración para la FPGA. Utilizando el entorno desarrollado mediante la librería software, se configura y verifica el circuito y por último se realiza una verificación funcional del algoritmo mediante el envío de imágenes y/o señales desde el host y la posterior visualización de los resultados.

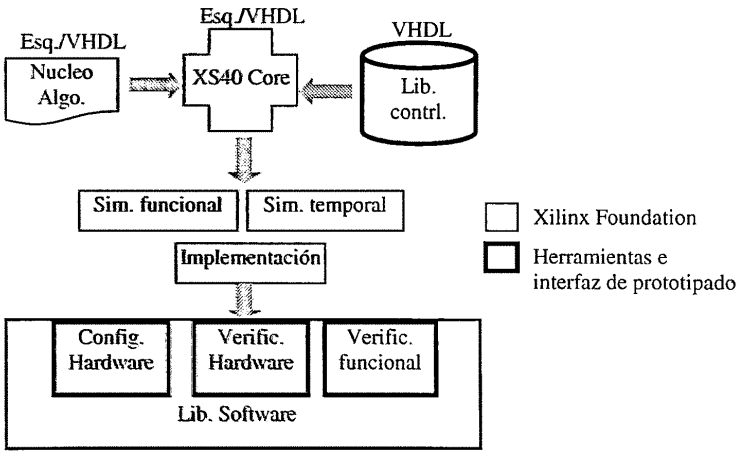


fig 3. Flujo de diseño.

5.- IPXS: ENTORNO PARA EL DESARROLLO DE PROYECTOS DE PROCESAMIENTO DE IMAGEN.

Siguiendo esta filosofía de trabajo y utilizando las librerías de prototipado, se ha desarrollado un entorno para el desarrollo de proyectos de procesamiento de imagen (IPXS). Utilizando este entorno visual (fig. 4) el usuario puede configurar la FPGA y comprobar su funcionamiento mediante un sencillo interfaz gráfico de generación de señales del puerto paralelo. Del mismo modo, puede abrir ficheros de imagen, visualizarlos, cambiar la resolución de los pixels y enviar la imagen a la memoria SRAM de la XS40 para ser procesada por la FPGA. El procesamiento se inicia desde el mismo entorno y una vez terminado este automáticamente se lee la imagen resultado y se visualiza. Existe además la posibilidad de comprobar la funcionalidad del algoritmo repitiendo localmente el procesamiento en el host mediante un programa C y comparando los resultados.

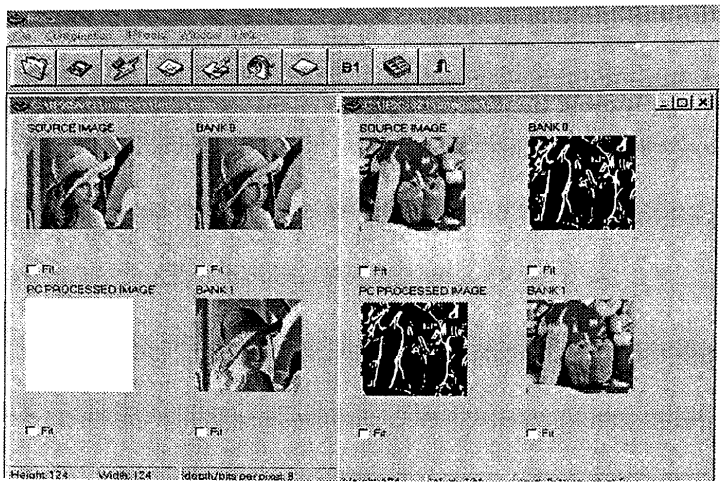


fig 4. Entorno IPXS

Al mismo tiempo se ha desarrollado una librería de controladores que realizan de forma transparente los accesos a la imagen siguiendo patrones típicos de visión: acceso por filas, acceso mediante ventanas 2x2, 3x3,4x4, 5x5, etc... (fig. 5). Pueden además, soportar distintas resoluciones de los pixels (1bit, 2bits, 4bits, ...). Para este tipo de proyectos se considera la memoria SRAM (32Kx8) dividida en dos bancos, B0 y B1 de 16Kx8, con lo que es posible trabajar con imágenes de hasta 124x124 pixels en 256 niveles de gris. Ambos bancos se utilizan indistintamente para almacenar la imagen original o el resultado del procesamiento, pudiéndose realizar procesamientos iterativos simplemente intercambiando los papeles.

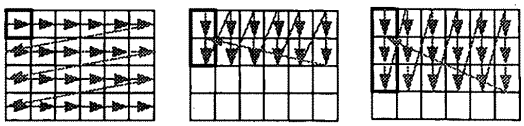


fig 5. Patrones de acceso: por filas, ventana 2x2 y 3x3

En la figura 4 se puede apreciar el resultado de los procesamientos mediante las operaciones NOT y SOBEL , mostrándose en cada caso el contenido de ambos bancos. En el último se muestra además el resultado del procesamiento local.

El entorno y la librería de controladores se encuentran actualmente disponibles en la dirección:
www.dtic.ua.es/dtic.local/asignaturas/DCAC/project/IPXS21.htm

6.- CONCLUSIONES

Se ha presentado una serie de herramientas y métodos de trabajo para facilitar la labor de los estudiantes en el desarrollado de prácticas y proyectos de visión y procesamiento de imagen basados en dispositivos programables (FPGAs). Gracias a estas herramientas se hace posible la utilización de una tarjeta de bajo coste para este tipo de proyectos, facilitando el acceso de los alumnos a prácticas más atractivas y motivadoras. Por último se presenta un entorno desarrollado sobre la base de esta filosofía para el prototipado de circuitos de procesamiento de imagen.

7.- AGRADECIMIENTOS

Este proyecto ha sido parcialmente financiado por la empresa Xess Corp.

8.- REFERENCIAS

[1] Goslin, G. R. "Using Xilinx FPGAs to Design Custom digital Signal Processing Devices". Proceedings of the DSP^X 1995.
[2] Nassif S. C. & Capson D.W. "A Flexible DSP-based Network for Real-Time Co-operative Windowing Applications". Real-Time Imaging. pp 283-293. 1997.
[3] Gómez Pulido, J. A. et. al " Actualización, Investigación y Desarrollo en el Entorno Docente de Arquitectura de Computadores en la Universidad de Extremadura". IV Jornadas sobre la enseñanza universitaria de la informática, Andorra 1998.
[4] Albadalejo, J. et al "Uso del VHDL en la enseñanza de Arquitectura de Computadores". IV Jornadas sobre la enseñanza universitaria de la informática, Andorra 1998.
[5] http://www.mirotech.com
[6] http://www.embeddedsol.com
[7] http://www.xess.com
[8] "The Programmable logic data book". Xilinx Inc. 1998.

NNOS: UN MICROKERNEL DISTRIBUIDO PARA DOCENCIA E INVESTIGACIÓN.

Enric Gibert[1], Pere Creus[1], Miquel Nicolau[1]

[1]*Enginyeria i Arquitectura La Salle - Universitat Ramon Lull:*
{enric,perecb,miquelm}@salleURL.edu

RESUMEN: el Grupo de Sistemas Operativos de *Enginyeria i Arquitectura La Salle* está diseñando e implementando un *microkernel* distribuido llamado NNOS como herramienta de docencia e investigación. Este proyecto se encuentra en una fase inicial, si bien ya hay partes completamente operativas. El marco actual nos hace ser optimistas y pensar que a finales de este año dispondremos de una versión local totalmente estable y a finales del curso académico 2001-2002, un prototipo distribuido totalmente funcional.

1.- INTRODUCCIÓN

El Grupo de Sistemas Operativos pertenece a la sección de Arquitectura de Computadores del Departamento de Informática de *Enginyeria i Arquitectura La Salle*. Este grupo está formado por profesores, proyectistas de primer y segundo ciclo y estudiantes colaboradores. Los integrantes son responsables de las áreas de docencia e investigación relacionadas con los sistemas operativos.

Dentro del área docente, el grupo es responsable actualmente de dos asignaturas: *Sistemas Operativos* y *Sistemas y Entornos Distribuidos*. La primera es una asignatura que se imparte en tercer curso de Ingeniería Técnica en Informática de Sistemas. Las clases teóricas se complementan con cuatro prácticas: una en la que los alumnos se familiarizan con las llamadas a sistema de *Unix* y las otras tres se enfocan en la implementación de un pequeño micronúcleo llamado *Quark*. La asignatura de *Sistemas y Entornos Distribuidos* es una asignatura similar a la anterior que se imparte en tercer curso de Ingeniería Técnica Multimedia. Los fundamentos teóricos son similares a los de *Sistemas Operativos*, perfilando la asignatura de manera distinta para adecuarla a esta titulación. Ambas asignaturas tienen 3 horas semanales de clase y 9 créditos.

El proyecto *Quark* también se enmarca dentro de la área docente del Grupo de Sistemas Operativos dado que se usa como práctica para los alumnos de *Sistemas Operativos*. *Quark* es el diseño de los componentes básicos de un micronúcleo que los alumnos deben implementar. Los componentes que lo forman son: un pequeño gestor de interrupciones; un gestor de procesos,

encargado de ofrecer las abstracciones de proceso y *thread* e implementar un planificador preemtivo basado en prioridades; un gestor de entrada/salida; y finalmente, el mecanismo de comunicación entre procesos (IPC), que será un mecanismo de paso de mensajes basado en puertos. La práctica se divide en tres entregas debido a su carga, en las que se van implementando cada uno de los módulos.

La segunda área de la cual es responsable el Grupo de Sistemas Operativos es la de investigación. En estos momentos se trabaja en dos proyectos distintos: el proyecto NNOS y el proyecto OSEK. Del primero hablaremos en más detalle puesto que es el objetivo de esta ponencia. El proyecto OSEK, por otro lado, se basa en el diseño e implementación de un sistema operativo basado en el estándar OSEK/VDX. OSEK (*Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug*) [1] es una especificación de varias empresas (entre ellas BMW, Renault, Volkswagen y Volvo) de un sistema operativo de tiempo real para la automoción. Evidentemente, este diseño se puede extrapolar a cualquier entorno de control.

2.- NECESIDADES ACTUALES Y OBJETIVOS

Antes de iniciar el proyecto NNOS, se detectaron una serie de necesidades en el grupo las cuales dieron paso a este proyecto. En la área de la docencia falta un entorno de desarrollo completo y potente para que el alumno pueda tratar otras partes del sistema operativo. Con *Quark*, los alumnos no pueden trabajar con partes importantes de un sistema operativo como la gestión de memoria, el cargador, etc. debido a que empiezan el sistema desde cero y lo desarrollan bajo 80x86 en modo real. También interesa un entorno como este para las prácticas de futuras asignaturas avanzadas de sistemas operativos que se pretenden ofrecer en segundo ciclo. Además, este entorno se podría utilizar para investigación, puesto que si se dispone de un sistema operativo flexible y distribuido, puede ser fácilmente adaptado para temas de investigación más avanzados.

Dadas estas necesidades, se emprendió el proyecto NNOS con los siguientes objetivos:

1. Disponer de una herramienta de docencia adecuada para asignaturas introductorias y avanzadas de sistemas operativos. La herramienta también debería ser útil para la investigación.

2. La herramienta debe ser flexible facilitando la inclusión de nuevos módulos o políticas por parte de los alumnos e investigadores.

3. Todos los componentes del sistema deben ser fácilmente mantenibles, sacrificando, si fuera necesario, el rendimiento. Esta mantenibilidad implica que el código sea lo más claro posible y que todas las funciones y estructuras utilizadas se documenten exhaustivamente.

4. El sistema ha de ser portable a otras plataformas.

3.- VISIÓN GLOBAL DE NNOS

NNOS es un sistema operativo multiproceso, *multithread* y distribuido, basado en una arquitectura micronúcleo. Está formado por un núcleo o *kernel*, que es el encargado de proporcionar los mecanismos básicos necesarios, y un conjunto de procesos servidores, que se ejecutan en modalidad usuario y que proporcionan parte de la funcionalidad del sistema

operativo [2]. Dado que el núcleo de NNOS no implementa políticas, será necesario que sean los servidores los que las definan e implementen, lo que implica una gran flexibilidad.

Dentro del núcleo podemos distinguir varios módulos diferenciados como son el gestor de comunicación entre procesos (IPC), el gestor de procesos (PH), el gestor de interrupciones (IH), el gestor de memoria (MEM), entre otros. Cada uno de estos módulos, sin embargo, está dividido en dos capas:

- **Parte dependiente del Hardware o HDL**: es la parte del núcleo del sistema operativo que interacciona con la máquina y que, por lo tanto, hay que modificar dependiendo de la plataforma sobre la cual se ejecute NNOS.

- **Parte independiente del Hardware o HIL**: es la parte del núcleo que utiliza las abstracciones que ofrece el HDL para realizar operaciones de más alto nivel.

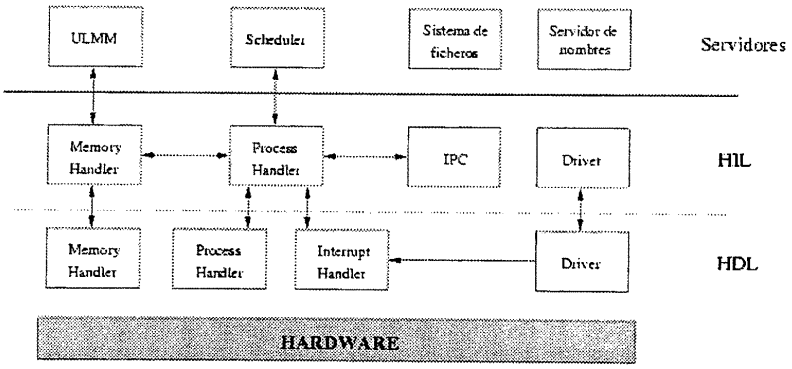


Figura 1. Modelo del Sistema Operativo NNOS.

NNOS responde a las siglas de *No Name Operating System*. Aunque en esta primera versión el sistema operativo no es aún distribuido, todas las estructuras subyacentes están pensadas para que, en un futuro, se puedan utilizar de forma distribuida. Lo que da a NNOS la apariencia de sistema operativo distribuido es la transparencia en las llamadas a sistema. Existe un único conjunto global de llamadas a sistema común para todas las máquinas que responden todas las peticiones del sistema, tanto locales como las remotas.

4.- MÓDULOS DEL MICROKERNEL

A continuación presentamos, con un poco más de detalle, los componentes que forman el *microkernel*.

a) Gestor de interrupciones

El gestor de interrupciones es el encargado de ejecutar la rutina de servicio de interrupción (RSI) adecuada según el tipo de interrupción que se haya producido. Este módulo ofrece servicios a nivel de *kernel* para asociar y desasociar rutinas RSI a interrupciones.

b) Gestor de procesos

NNOS ofrece las abstracciones de tarea y *thread* a nivel de *kernel*. Una tarea es un contenedor de recursos (espacio de memoria y puertos) donde se ejecutan uno o más *threads*. Sin embargo, NNOS no ofrece ninguna política de planificación a nivel de *kernel*, sinó que éstas se implementarán mediante uno o más procesos servidores planificadores (llamados planificadores de alto nivel o simplemente planificadores).

Para conseguir esto, NNOS divide el ciclo de planificación en un número determinado de intervalos de tiempo fijo llamados *slots* que se repiten cíclicamente. Estos *slots* se dividen en unidades de ejecución más pequeñas (*thread slots*) y tienen una cola de *threads* asociada con las hebras que deberán ejecutarse durante el tiempo que dura el *slot*. Los planificadores se registrarán en uno o más de estos *slots* teniendo en cuenta que un *slot* sólo podrá pertenecer a un único planificador. En dichos *slots* el planificador decidirá qué política utilizará (y por tanto, qué *threads* deberán ejecutarse a continuación) actualizando su cola correspondiente.

El mecanismo implementado internamente se basa en ceder el procesador al primer *thread* de la lista correspondiente al *slot* actual durante los *k thread slots* asignados por el planificador a dicha hebra. Una vez que se haya agotado este tiempo o que el *thread* en ejecución se bloquee, se cede el procesador al siguiente. Cuando la cola queda vacía o se produce un acontecimiento interesante para el planificador (p.e. se despierta un *thread*), se cede el procesador al *thread* del planificador para que llene de nuevo la cola o la modifique.

c) Gestor de *capabilities*

Todos los recursos (tareas, puertos, regiones de memoria...) de NNOS se identifican con *capabilities* [3]. Una *capability* está compuesta por el nombre o identificador del recurso, los permisos sobre éste y finalmente una firma digital para protegerlas ya que residen en espacio de usuario. Cuando una tarea de usuario crea un recurso, el *kernel* le devuelve una *capability* con todos los permisos activos que lo identifica. A partir de ésta, la tarea puede derivar otras *capabilities* que hagan referencia al mismo recurso con distintos permisos.

El módulo de *capabilities* es la parte del *kernel* encargada de crear, comprobar y destruir las *capabilities* de los recursos. Este módulo es usado por los demás componentes del *kernel*. Por ejemplo, cuando un usuario crea una tarea, realiza una llamada a sistema que es servida por el gestor de procesos y éste acaba llamando al gestor de *capabilities* para crear una *capability*. Cuando un usuario haga una llamada a sistema para suspender una tarea especificando la

capability de ésta, el gestor de procesos pedirá al gestor de *capabilities* que la compruebe y verifique que tiene permisos para dicha acción.

d) Gestor de comunicación entre procesos (IPC)

El mecanismo de comunicación entre procesos que ofrece el *kernel* se basa en el paso de mensajes a través de puertos. Existen dos tipos distintos de puertos: los puertos normales y los puertos de *send/receive*. Los primeros son puertos en los que se pueden dejar mensajes (llamada a sistema *send*) o recojerlos (llamada a sistema *receive*), mientras que los segundos no pueden ser accedidos con un *send* sinó con una operación *send/receive* que combina ambas acciones dentro del *kernel*. Este segundo tipo de puertos es similar al mecanismo de llamada remota a procedimientos (RPC) [4] y son necesarios en una arquitectura *microkernel* puesto que la mayoría de servicios son ofrecidos por procesos de usuario. La comunicación es totalmente síncrona (estilo *rendez-vous*) y los mensajes son de tamaño variable.

El gestor de comunicación entre procesos también ofrece eventos como un segundo mecanismo de comunicación. Un evento es un acontecimiento que altera temporalmente el flujo de ejecución de un *thread* concreto. NNOS ofrece dos tipos de eventos: síncronos y asíncronos. Los primeros suceden en un instante concreto (como por ejemplo la división por cero, un fallo de página...) y la rutina asociada a ellos se ejecuta en el contexto del *thread* que lo ha producido. Los eventos asíncronos, por su parte, no se producen en instantes concretos y su significado lo definen las dos tareas involucradas (vendrían a ser como los *signals SIGUSR1* y *SIGUSR2* de Unix). La ejecución de la rutina asociada se realiza en el contexto de cualquier *thread* de la tarea receptora.

e) Gestor de memoria

El gestor de memoria de NNOS ofrece las funcionalidades básicas de gestión de memoria que se complementan con funcionalidades y políticas más complejas a través de gestores externos llamados ULMM (*User Level Memory Manager*). Las dos abstracciones básicas definidas por el *kernel* son las regiones de memoria (*mem_regions*, o secciones contiguas del espacio lógico de una tarea) y las páginas de memoria (*mem_pages*). NNOS ofrece llamadas para reservar espacio lógico para una tarea (crear una *mem_region* que se identificará con una *capability*), reservar páginas físicas y mapear o desmapear páginas.

Los gestores externos serán los encargados de aplicar las funcionalidades para compartir memoria y ofrecer memoria virtual. Para ello, se ha diseñado un ULMM por defecto que define la abstracción de objeto de memoria (*memory_object*) o conjunto de páginas físicas. Este proceso ofrece el servicio de asociar una *mem_region* a un *mem_object*, lo que implica la asociación de páginas lógicas a físicas. Si dos tareas distintas se mapean el mismo *memory_object* compartirán memoria.

f) Servidores a nivel de usuario

También se ha pensado inicialmente en dos procesos más de usuario (a parte del planificador de alto nivel y del gestor de memoria ULMM) necesarios para el funcionamiento básico de NNOS en su primera versión. Estos procesos son: un servidor de nombres, para poder obtener fácilmente referencias a recursos; y un servidor de ficheros y cargador para poder ejecutar procesos dinámicamente.

5.- CAPACIDADES DISTRIBUIDAS

Uno de los primeros objetivos de NNOS es que sirva como base para un sistema operativo distribuido. De hecho, se añade dicha funcionalidad permitiendo que las *capabilities* no sólo hagan referencia a recursos locales, sinó que pueden identificar recursos remotos de manera transparente al usuario.

Así pues, cuando se hace referencia a un recurso remoto (por ejemplo se hace un *send* a un puerto de una tarea remota), el gestor de *capabilities* ha de poder detectar la ubicación de este recurso y encaminar la petición a su destino de manera totalmente transparente al usuario. Lo mismo sucede con el resto de recursos identificados con *capabilities* como las tareas, regiones de memoria, ...

6.- ESTADO ACTUAL DEL PROYECTO

El proyecto NNOS se encuentra en un estado inicial, si bien empieza a tener partes totalmente operativas. Actualmente tenemos implementado el gestor de interrupciones, el gestor de procesos, el gestor de IPC (si bien se está rediseñando este), parte del gestor de *capabilities* y el gestor de memoria para arquitecturas 80x86 en modo protegido [5]. Estos módulos han sido implementados por proyectistas de primer ciclo. En fase de desarrollo tenemos el servidor de ficheros y cargador, el servidor de nombres, el diseño de la fase de arranque del sistema y algún que otro *device driver*. Finalmente, estamos diseñando el mecanismo DIPC (*Distributed IPC*) para tener un sistema operativo totalmente distribuido y un gestor de memoria ULMM que ofrezca memoria compartida bajo arquitecturas multicomputador.

La planificación que tenemos para este proyecto incluye la finalización total del prototipo local (sin ningún mecanismo distribuido) para finales de año y tener un prototipo distribuido para finales del curso académico siguiente.

REFERENCIAS

[1] *OSEK/VDX. Operating System specification 2.1*. Disponible en http://www.osek-vdx.org/.

[2] William Stallings. Operating Systems: internals and design principles. Prentice Hall, 1998.

[3] A. S. Tanenbaum, S. J. Mullender, R. Van Renesse. *Using Sparse Capabilities in a Distributed Operating System*. Proc. Of the 6th Int. Conference on Distributed Computing Systems, IEEE Press, 1986.

[4] Pradeep K. Sinha. *Distributed Operating Systems. Concepts and Design*. IEEE Computer Society Press, 1996.

[5] Pentium Pro Family Developer's Manual. Volume 3: Operating System Writer's Guide. Intel Corporation, 1995.

SIMULADOR DE UN COMPUTADOR SEGMENTADO CON ESPECULACIÓN *HARDWARE*.

Juan Carlos Martínez y Pedro López

Departamento de Informática de Sistemas y Computadores.
Escuela Universitaria y Facultad de Informática de Valencia.
Universidad Politécnica de Valencia.
e-mail: jc@gap.upv.es

RESUMEN: En el presente artículo se presenta un simulador de un computador DLX con especulación *hardware*. El simulador permite la ejecución ciclo a ciclo de pequeños programas, visualizando el estado de la unidad de instrucción segmentada y el avance de las instrucciones por la misma.

1.- INTRODUCCIÓN.

Las materias propias de "Arquitectura de Computadores" estudian, entre otras, las técnicas empleadas en los modernos procesadores para conseguir altas prestaciones. Así, es habitual encontrar el estudio de las técnicas de segmentación, planificación dinámica de instrucciones, predicción de saltos, especulación, procesadores superescalares, etc, en los programas de dichas asignaturas [2].

Para la comprensión de estas técnicas es conveniente la resolución de ejercicios en los que se realiza un seguimiento ciclo a ciclo (o paso a paso) del estado de los distintos componentes del procesador. La realización de estos ejercicios se ve facilitada mediante el empleo de simuladores que permitan ejecutar programas sencillos, aplicando las técnicas que se están estudiando. De esta manera, el estudiante puede validar los ejercicios efectuados, al tiempo que alcanzar una mejor comprensión de dichas técnicas. A diferencia de otros simuladores, cuyo objetivo es favorecer el aprendizaje del lenguaje máquina, en el caso que nos ocupa hay que hacer un especial hincapié en la utilización de un modelo detallado del sistema a simular, junto con una buena interfaz con el usuario. Así pues, el simulador debe ofrecer, además de los resultados de la ejecución del programa (tiempo transcurrido, estado de los registros y memoria), la posibilidad de simular con detalle y visualizar el avance de las instrucciones a lo largo de la unidad de instrucción segmentada, la reserva de estaciones con un algoritmo de planificación dinámica de instrucciones, etc,...

En este artículo se presenta un simulador de un computador segmentado con planificación dinámica de instrucciones y especulación (*xdlxvsim*), desarrollado en el lenguaje de programación C bajo el sistema operativo Linux y entorno gráfico X-Window. El resto del artículo está organizado como sigue. La sección 2 describe la arquitectura de la máquina simulada. La sección 3 describe el interfaz con el usuario. Finalmente, se presentan algunas conclusiones.

2. ARQUITECTURA DE LA MÁQUINA SIMULADA.

Xdlxvsim es un simulador de un computador segmentado con gestión dinámica de instrucciones y especulación. La arquitectura de la máquina simulada se basa en el computador DLX descrito por Hennessy-Patterson en [1]. El procesador simulado tiene un ciclo de instrucción segmentado, aplicando técnicas de planificación dinámica de instrucciones para permitir alterar el orden de lanzamiento a ejecución de las instrucciones, y predicción dinámica de instrucciones de salto basadas en un *branch target buffer* (*BTB*) de 1 bit y especulación hardware, siguiendo el algoritmo propuesto en [1]. Sólo los saltos que se predicen como tomados se almacenan en la tabla *BTB*.

La ruta de datos consta de dos buses de datos internos independientes (Figura 1), uno para instrucciones enteras y el otro para las instrucciones en coma flotante. Existe un único *reorder buffer* para ambos buses para conservar el orden de ejecución de las instrucciones. En la Figura 2 se puede observar la parte de la ruta de datos dedicada a las operaciones en coma flotante. Como podemos comprobar se dispone de una unidad de ejecución diferente según el tipo de operación. Cada operador tiene asociado un conjunto de estaciones reserva. Nótese que algunas instrucciones (cargas, almacenamientos o transferencia de datos) pueden necesitar utilizar el bus de enteros y el de coma flotante.

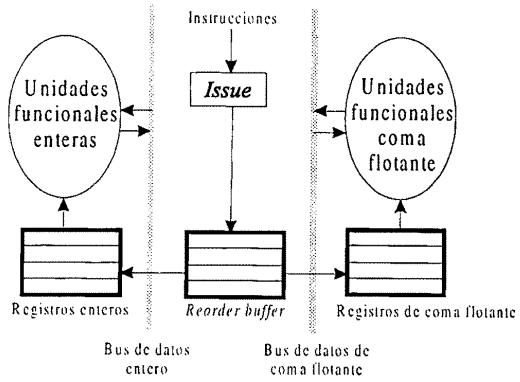


Figura 1. Ruta de datos del procesador simulado.

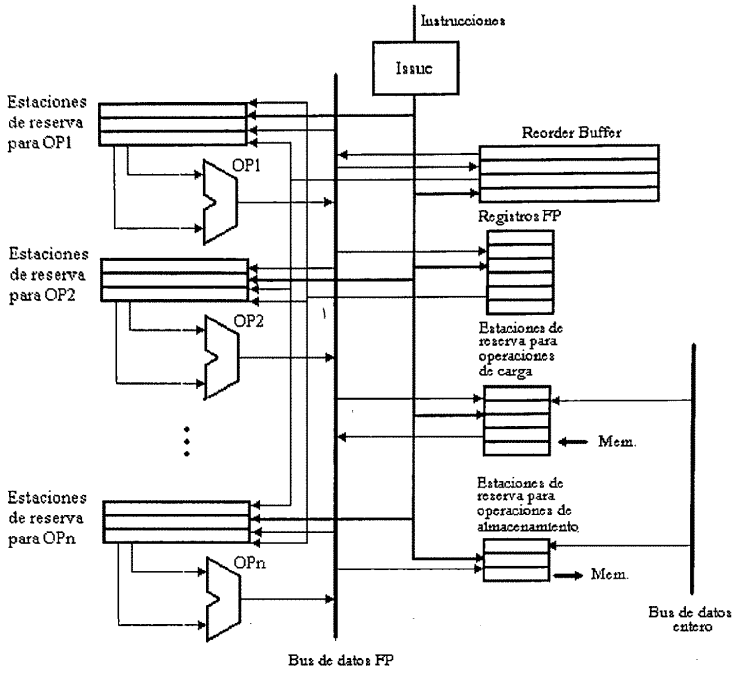


Figura 2 Unidad de coma flotante.

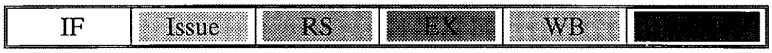


Figura 3: Segmentación del procesador dimulado.

El ciclo de instrucción está segmentado en seis fases (Figura 3). La fase *IF* realiza la búsqueda en memoria de la siguiente instrucción a ejecutar y calcula la dirección de la siguiente instrucción. En esta fase se realiza la predicción de saltos consultando el *BTB*.

La fase *Issue* decodifica la instrucción y la lanza según el algoritmo de planificación dinámica de instrucciones utilizado y resuelve las posibles dependencias de datos. Lo primero que se hace es decodificar la instrucción y extraer los operandos fuentes y destino, o la dirección destino de salto, en su caso. Después se busca una posición libre en el *reorder buffer* y una estación de reserva libre en el operador correspondiente, introduciendo en dichas tablas los datos necesarios. Si se dispone de todos los operandos y hay un operador libre la instrucción es pasada directamente a la fase *EX*. En otro caso la instrucción pasa a la fase *RS*.

La fase *RS* se limita a comprobar si la instrucción tiene los operandos fuentes disponibles y un operador libre para poder pasar la instrucción a la fase *EX*.

La fase *EX* realiza la ejecución de la instrucción y calcula el resultado de la operación. Esta fase es multiciclo, es decir, dependiendo de la operación a realizar esta puede durar varios ciclos. Por otro lado, varios operadores pueden trabajar en paralelo.

Una instrucción llega a la fase *WB* cuando ha terminado de calcular el resultado en la fase *EX* volcándolo sobre el bus de datos. Al tener buses separados para enteros y coma flotante, pueden entrar una instrucción entera y una en coma flotante a la vez a esta fase. Para diferenciarlas las denominaremos *WBI* (enteros) y *WBF* (coma flotante), respectivamente. En esta fase, la instrucción que toma el bus vuelca no solo el resultado de su operación en el bus sino también un identificador de la instrucción para que todas las estaciones de reserva lean el dato si es un operando fuente de alguna operación en espera. El *reorder buffer* almacenará el dato que se vuelca en el bus.

La última fase es la fase *Commit* que es la encargada de actualizar el estado de la máquina. Las instrucciones ejecutadas llegan en orden a esta fase. En esta fase, las operaciones que depositan su resultado en un registro actualizan el banco de registros y las instrucciones de almacenamiento lanzan la escritura en memoria. Las instrucciones de salto comprueban si se predijo correctamente el salto y en caso de error se anulan todas las instrucciones que se haya lanzado. En esta fase se comprueban también las excepciones que pudo provocar la instrucción en cualquier fase de la segmentación.

4.- INTERFAZ CON EL USUARIO.

En este punto vamos a presentar el simulador utilizando un ejemplo. Para ello utilizaremos un programa que suma un escalar a un vector almacenado en la memoria, cuyo código es el siguiente:

```
tamanyo:    .word 256

start:      lw r1, tamanyo(r0)      ; r1=límite
            add r2,r0,r0            ; r2=0 sera el indice
            ld f0, a(r0)            ; f0=a

bucle:      ld f2, x(r2)            ; f2=X[i]
            addd f8, f2, f0         ; f8= a + X[i]
            sd z(r2), f8            ; Z[i]= a+ X[i]
            add r2, r2,8            ; i++
            slt r3, r2, r1          ; compara si es el último elemento
            bnez r3, bucle          ; bucle
            trap #0                 ; fin de programa.
```

La aplicación procura minimizar al máximo el espacio en pantalla. La ventana principal es una simple barra de herramientas (Figura 4).

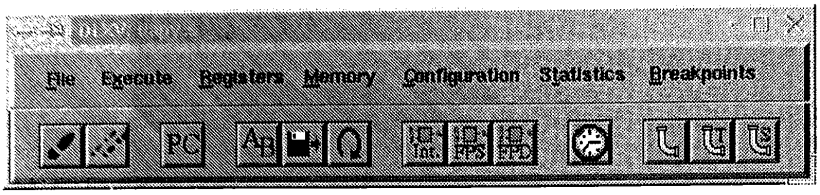


Figura 4: Ventana principal del programa.

Mediante los menús podemos acceder a todas las funciones del programa, y algunas de ellas han sido añadidas a la barra de herramientas en forma de botones con dibujos intuitivos. Las funciones del programa son las siguientes. Mediante el menú *File* se puede cargar un programa, abrir un editor para modificarlo, recargar el programa cargado si lo hemos modificado con el editor o salir del programa. Al cargar o recargar se ensambla automáticamente el programa. El menú *Execute* permite ejecutar el programa ciclo a ciclo, un cierto número de ciclos o hasta que termine el programa o al alcanzar un punto de ruptura. También en este menú se puede inicializar la máquina y las estadísticas. El menú *Registers* permite abrir acceder a los registros del procesador, enteros, de coma flotante de simple o doble precisión, así como a los registros internos del procesador. En el menú *Memory* podemos consultar o cambiar zonas de memoria. El menú *Configuration* sirve para configurar el simulador. Se puede configurar el tamaño de memoria, el número de estaciones de reserva, el número de operadores, sus latencias, el tamaño del *reorder buffer*, etc. También permite configurar el editor a utilizar para escribir el código fuente de los programas. Con el menú *Statistics* se accede a las estadísticas del programa ejecutado y con el menú *breakpoints* se pueden configurar puntos de ruptura para el programa.

Los tres botones de la derecha muestran el estado de las etapas de la segmentación. En el caso que nos ocupa, el botón es el ubicado al final cuya "S" hace referencia a la especulación (ver Figura 5). El botón con la forma de un reloj abre la ventana *Clock Cycles Diagram* (ver Figura 6) con un esquema de la ejecución de las instrucciones, indicando en qué fase de ejecución se encuentran en cada ciclo.

Vamos a pasar ahora a visualizar la simulación del programa. En la figura 6 se puede observar los primeros ciclos del programa. Obsérvese la ejecución de las instrucciones fuera de orden en los operadores y la terminación en orden (etapa *Commit*). Nótese que en el ciclo 9 se efectúa incorrectamente la predicción como "no tomado" (el salto no está todavía en la tabla BTB), cancelándose en el ciclo 17 las instrucciones lanzadas a ejecución incorrectamente. En la figura 5 se observa el estado de la unidad de instrucción segmentada en el ciclo 7.

El simulador también permite visualizar el estado de todas las tablas de la unidad de instrucción (estaciones de reserva, reorder buffer, BTB, etc). La Figura 7 muestra capturas de pantalla con esta información. Finalmente, la Figura 8 muestra un ejemplo de las estadísticas de ejecución que ofrece la aplicación (en este caso, las instrucciones ejecutadas clasificadas por códigos de operación).

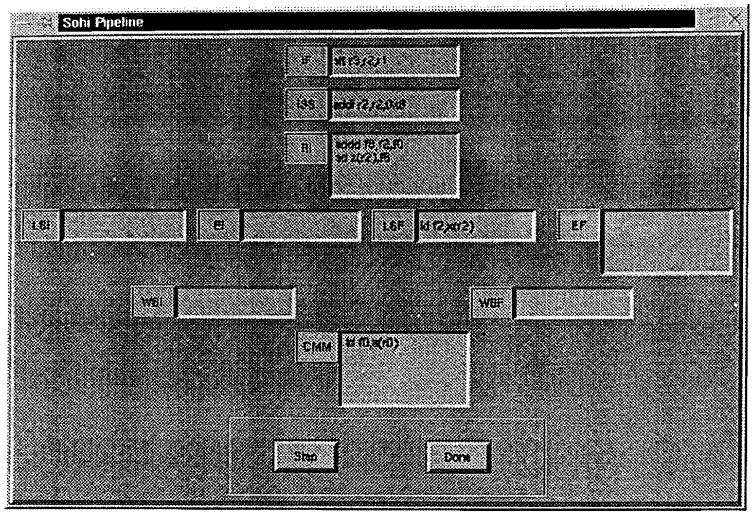


Figura 5: Avance de las instrucciones en la unidad de instrucción.

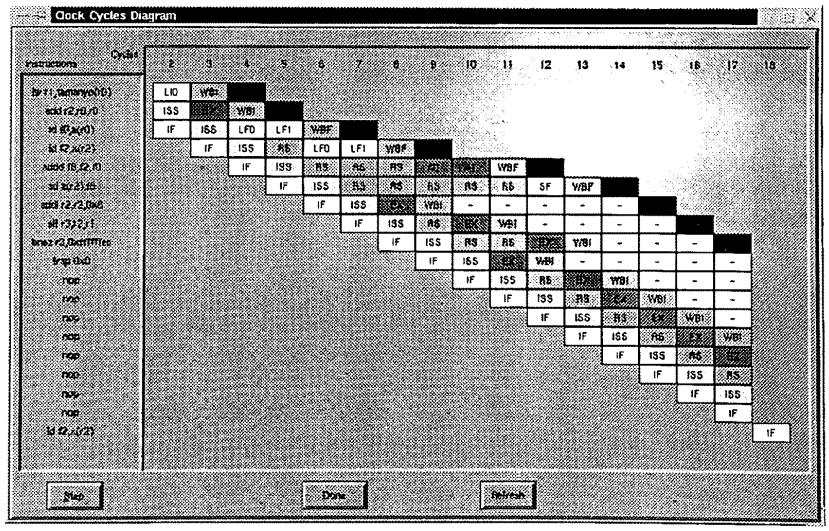


Figura 6: Esquema de ejecución de una iteración del programa.

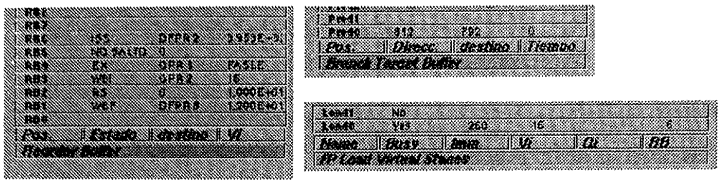


Figura 7: Estructuras de datos de la unidad de instrucción.

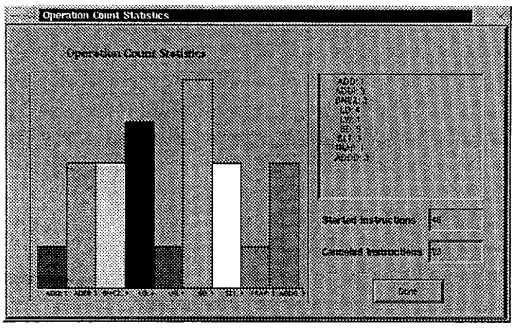


Figura 8: Ejemplo de estadísticas de ejecución.

5.- CONCLUSIONES

En este artículo se ha presentado un simulador de un computador con planificación dinámica de instrucciones, predicción de saltos y ejecución especulativa. El simulador es capaz de ejecutar programas escritos en lenguaje ensamblador del computador *DLX* y ofrece la posibilidad de visualizar el avance de las instrucciones por las etapas de la unidad de instrucción, el diagrama de ejecución ciclo a ciclo y el contenido de todas las estructuras de datos. El simulador se ha desarrollado como una herramienta de apoyo a la docencia en la materia de "Arquitectura de Computadores", permitiendo a los estudiantes la realización y ejecución de pequeños programas, alcanzando una mejor compresión del funcionamiento de las técnicas de especulación hardware.

BIBLIOGRAFÍA

[1] J. Hennessy, D.A. Paterson, *Computer Architecture: A quantitative approach.* Morgan Kauffman, 1996.

[2] http://dlxv.disca.upv.es, Web de las asignaturas *Arquitectura de Computadores I y II* Facultad de Informática. Univ. Politécnica de Valencia.

PRÁCTICAS DE ENSAMBLADOR

José Flich[1], Jorge Real[1], Juan Carlos Cano[1] y Julio Sahuquillo[1]

[1]*Departamento de Informática de Sistemas y Computadores*
E. U. I.–Universidad Politécnica de Valencia–
e_mail: {jflich, jorge, jucano, jsahuqui}@disca.upv.es

RESUMEN: En la asignatura de microprocesadores y Periféricos (MYP) se imparten prácticas de ensamblador. Estas prácticas estaban diseñadas para que el alumno solamente aprendiese el lenguaje. Para ello el alumno implementaba programas bajo un entorno MS-DOS. A su vez, el alumno, en la parte teórica, aprendía conceptos de microprocesadores utilizándose el 8086 de Intel como referencia.

En el curso actual (99-00), se ha modificado el temario y las prácticas de la asignatura con el fin de motivar al alumno ofreciendo ejemplos prácticos en los que el lenguaje ensamblador es de utilidad. Además, la parte teórica se ha basado en el procesador Pentium.

En este artículo se describen los cambios realizados en la asignatura tanto en la parte teórica como en la práctica.

1.- INTRODUCCIÓN.

En la asignatura de Microprocesadores y Periféricos (MYP) de la Escuela Universitaria de Informática de la Universidad Politécnica de Valencia se han desarrollado en los últimos años prácticas del lenguaje ensamblador. Esta asignatura es optativa y se enmarca en el tercer curso de la titulación de Ingeniería Técnica en Informática de Gestión (ITIG).

En esta asignatura se imparte docencia de dos temas claramente diferenciados. En una primera parte se imparten conceptos de los microprocesadores mientras que en una segunda parte se imparten conceptos de periféricos del computador. Ambas partes tienen la misma duración de 7 semanas, tanto en las clases teóricas como en las clases prácticas. La asignatura tiene asignados 3 créditos de teoría y 3 de prácticas.

En los cursos anteriores, en la parte de microprocesadores se utilizaba como base el microprocesador 8086. Se profundizaba en conceptos de organización y arquitectura del juego de instrucciones del procesador 8086.

En las sesiones de prácticas correspondientes a la parte de microprocesadores, se profundizaba en el conocimiento del lenguaje ensamblador. Las prácticas ejercitaban la declaración de variables en ensamblador, los modos de direccionamiento, las instrucciones de salto, las instrucciones de cadena, etc. Estas prácticas se realizaban sobre el sistema operativo MS-DOS.

El código fuente era introducido por los alumnos con el editor *edit* de MS-DOS y se utilizaba el *TASM*, *TLINK* y *TD* de Borland para ensamblar, enlazar y depurar el código.

Las practicas en ensamblador son importantes para el alumno, pues ayudan a comprender el funcionamiento interno del computador, así como a conocer mejor la arquitectura y organización del microprocesador en cuestión. Además, el lenguaje ensamblador es fundamental en el diseño de compiladores. Igualmente, el ensamblador es de gran valor en aquellas aplicaciones con requerimientos de tiempo críticos. En estas aplicaciones, el código implementado en ensamblador puede facilitar el cumplimiento de requisitos temporales de la aplicación. Ejemplos de tales aplicaciones son los de toma de datos, captura y procesado de imágenes, control de procesos, etc. Por último, el conocimiento del lenguaje ensamblador es un buen medio para conocer mejor el funcionamiento de un lenguaje de alto nivel.

No obstante, para el alumno, el aprendizaje de dicho lenguaje siempre supone un pequeño rechazo, pues piensa que el lenguaje ensamblador no es útil en la actualidad, ya que casi cualquier programa puede ser implementado íntegramente con un lenguaje de alto nivel. Además, el alumno estaba doblemente desmotivado por el entorno de trabajo utilizado en las prácticas, ya que se trabajaba con el sistema operativo MS-DOS, estando más familiarizado con sistemas operativos con un entorno gráfico.

En el curso actual (99-00), se ha cambiado la orientación de la docencia (teoría y prácticas) de la parte de microprocesadores de la asignatura MYP para darle un enfoque más atractivo y actual al alumno. En primer lugar, se ha optado por el cambio de la materia impartida dándole un enfoque más novedoso. Por ello, se ha modificado el temario de la asignatura utilizando el microprocesador Pentium como ejemplo. En lo referente a las prácticas, se han desarrollado nuevas prácticas que motiven más al alumno y que de verdad muestren los beneficios del lenguaje ensamblador al mismo tiempo de que sirvan para que el alumno aprenda el lenguaje ensamblador. Para ello, las prácticas se han desarrollado bajo el entorno MS-Windows (el alumno trabaja con aplicaciones Windows desarrolladas con el lenguaje ensamblador) y se han diseñado para que demuestren tales ventajas.

En el presente artículo, se detalla la organización del nuevo material tanto teórico como práctico de la parte de microprocesadores, haciendo hincapié en la experiencia personal de tales cambios.

2. TEMARIO DE LA ASIGNATURA Y BIBLIOGRAFÍA.

En la Tabla 1 se muestra el temario actual de la asignatura y la duración de cada tema. Los tres primeros temas corresponden a microprocesadores.

Tema	Título	Semanas
1	Introducción a los Microprocesadores	2
2	Arquitectura del Juego de Instrucciones del Pentium	3
3	Estructura/Organización del Pentium	2
4	Dispositivos de Entrada	2
5	Interfaces Paralelo y Serie	3
6	Dispositivos de Salida	2
-	Presentación de trabajos	1

Tabla 1 Temario de la asignatura MYP.

En un primer tema se introducen los microprocesadores y se describen de forma general las características comunes y particulares de los microprocesadores actuales. Se tratan temas como unidades funcionales, segmentación, prebúsqueda de instrucciones, ejecución dinámica, etc. También se describe la arquitectura del juego de instrucciones típica en los microprocesadores actuales. En este primer tema se ofrece un enfoque global, para que el alumno tenga conciencia de que además del microprocesador que se detalla en los siguientes temas (Pentium) también existe una gran variedad de microprocesadores.

En los temas 2 y 3 se describe con todo detalle un ejemplo práctico: el Pentium. Se describe tanto la arquitectura de su juego de instrucciones como su organización. Hay que resaltar que estos dos temas están diseñados para que el alumno conozca la evolución de los microprocesadores 80x86 de la familia Intel. Para ello, los temas hacen hincapié en cada apartado en su ámbito de aplicación (en qué microprocesador está implementado).

En estos temas se describen conceptos como: modos de direccionamiento, modos de operación, registros y juego de instrucciones. En la parte de ensamblador también se describen: etiquetas y tipos de datos, operadores, atributos, definición de segmentos, etc. Los conceptos más avanzados de direccionamiento a memoria (modos de operación) se describen de forma más generalista. La bibliografía asociada a estos temas se muestra en la Tabla 2.

Tema	Bibliografía
Tema 1	*Advanced Microprocessors*, Daniel Tabak. McGraw-Hill. 1991. Computer Organization And Architecture, William Stallings. Prentice Hall
Tema 2	The Art of Assembly Language Programming, Randall Hyde, 1996. Manual del Turbo Assembler, Borland
Tema 3	*Pentium System Architecture*. Tom Shanley. Pc System Architecture Series. Ed. Addison Wesley. (2nd edition). 1998 *Pentium Pro and Pentium II System Architecture*. Tom Shanley. Pc System Architecture Series. Ed. Addison Wesley. (2nd edition). 1997.

Tabla 2 Bibliografía Asociada

3. PRÁCTICAS DE LABORATORIO.

a) Entorno de trabajo.

Las prácticas de laboratorio se han desarrollado en un entorno gráfico (MS-Windows). El alumno utiliza/modifica el código en ensamblador de una aplicación Windows, sirviéndose para ello del entorno de desarrollo integrado de Borland C++ 4.52. Todo el código a modificar y/o examinar en las prácticas está integrado en un proyecto con diferentes módulos. Para realizar una buena depuración de la aplicación, el alumno utiliza el TurboDebugger (TD) de Borland, integrado en el propio BC++ 4.52. En la Ilustrción 1 podemos ver el entorno de trabajo con el Borland C++ 4.52 y el proyecto de la aplicación.

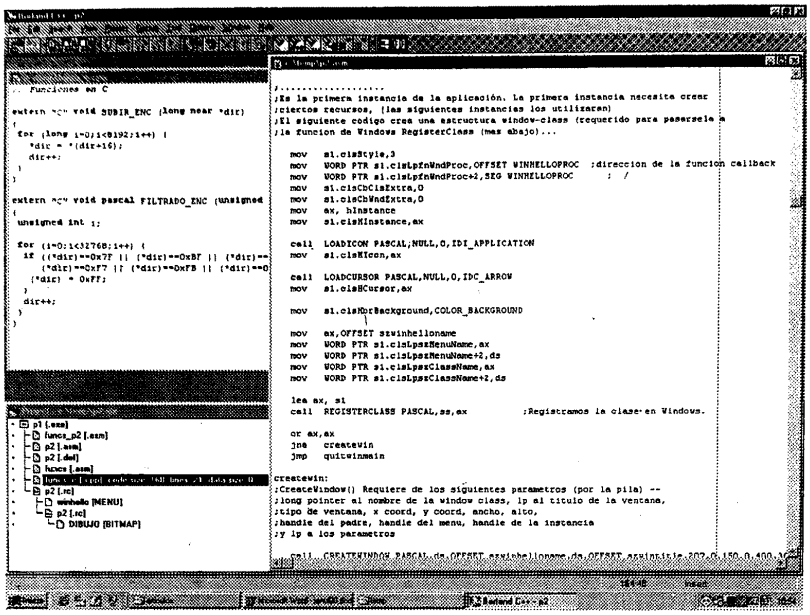


Ilustración 1 Entorno de trabajo y proyecto de la aplicación.

La estructura principal de la aplicación se facilita al alumno, dejando ciertos procedimientos para que el alumno codifique el código requerido en cada práctica.

b) Las prácticas de ensamblador.

Las prácticas de ensamblador son las siguientes:

Práctica	Semanas
Introducción a una aplicación Windows en ensamblador	2
Modos de direccionamiento	1
Instrucciones de cadena	1
Instrucciones varias	2
Recuperación	1

Tabla 2 Prácticas de microprocesadores (1,5 créditos).

En la primera práctica el alumno se familiariza con el entorno de trabajo y con la aplicación Windows en ensamblador (que será la aplicación que utilizará en el resto de prácticas). Para

ello, el alumno repasa todo el código de la aplicación. Dicho código está agrupado y comentado por bloques.

A su vez, el alumno empieza a relacionarse con los proyectos del Borland C++ 4.52 y aún más importante, se relaciona con el Turbo Debugger de Borland. Para ello, se proponen ejercicios de ejecución paso a paso, viendo cómo cambia el estado del procesador (registros, bits de estado, variables, etc.). Además también se ejercita en la búsqueda automática de código (prólogo, epílogo, llamadas a procedimientos) con el Turbo Debugger. Por último, y debido a la naturaleza del API de Windows, aprenden en la práctica el mecanismo de paso de parámetros y variables locales utilizando el convenio PASCAL. Debido a la relativa complejidad del código ensamblador y a que es la primera toma de contacto, está práctica se realiza en dos sesiones (4 horas en total). En la
Ilustración 1 Entorno de trabajo y proyecto de la aplicación.
se puede ver un fragmento del código de la aplicación.

En la segunda práctica se hace hincapié en los modos de direccionamiento del procesador. Esta práctica es imprescindible debido a la dificultad que los alumnos tienen en ejercitar y entender los modos de direccionamiento de los procesadores Intel. El buen conocimiento de los modos de direccionamiento es imprescindible para diseñar eficientemente código en ensamblador. Por lo tanto, esta práctica va a servir para que en las siguientes prácticas el alumno pueda utilizar los modos de direccionamiento de forma correcta. En la práctica acceden a estructuras de datos anidadas (vectores de registros, registros anidados, etc) utilizando diversos modos de direccionamiento.

En la práctica 3 ya se ejercita con un ejemplo concreto de mejora de prestaciones utilizando el lenguaje ensamblador. En concreto se pide realizar diferentes operaciones sobre una imagen en blanco y negro de 512×512 puntos. Se les facilita una función en C que desplaza una fila hacia arriba toda la imagen (*scroll*). Con la instrucción `cpuid` del juego de instrucciones del Pentium el alumno obtiene en ciclos de procesador el tiempo de ejecución de la función en C, para posteriormente implementar la función equivalente en lenguaje ensamblador. Esto lo realiza primeramente con instrucciones de transferencia (`mov`) con diferentes tamaños de operandos (*byte*, *word* y *double word*) y tomando los tiempos de ejecución con la instrucción `cpuid`. El alumno, también implementa la función con instrucciones de cadena y prefijos de repetición, tomando de nuevo tiempos, y viendo las mejoras que se obtienen en cada caso.

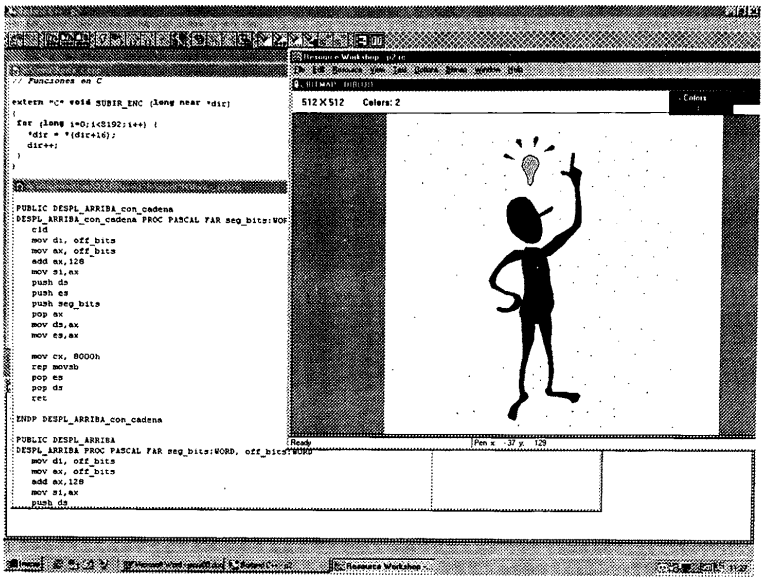


Ilustración 2 Funciones de scroll e imagen tratada.

En la misma práctica, el alumno debe identificar por qué la función en C resulta más lenta. Esto lo hace viendo directamente el código en ensamblador generado a partir de la función en C. El alumno identifica posibles deficiencias del código. En la Ilustración 2 vemos la imagen tratada y el código en C y ensamblador que implementan la función de *scroll*.

En la misma práctica se motiva al alumno para que ejercite otras instrucciones de cadena (`stos` y `lods`) y que identifique posibles pérdidas de prestaciones al utilizar estas instrucciones sin prefijos de repetición. Esto se realiza implementando una función de filtrado de la imagen (`xor` a cada byte del mapa de bits).

Por último en la práctica 4 se conciencia al alumno de que un código en ensamblador mal diseñado puede repercutir en un mayor tiempo de ejecución con respecto a un código equivalente en C. Para ello, el alumno debe realizar un filtrado a la imagen eliminando los *pixels* que aparezcan aislados en un byte del mapa de bits (un byte posee 8 *pixels*). El alumno incurre en la implementación del algoritmo con dos bucles anidados (el más externo para tomar bytes del mapa de bits y el más interno para contar el numero de *pixels* activos). El alumno aprende a mejorar su código hasta que su versión mejora el tiempo de ejecución del equivalente en C.

Para la realización de estas prácticas en MS-Windows se ha utilizado la siguiente referencia bibliografica:

Windows Assembly Language and Systems Programming,
Barry Kauler, R&D Books.

4. CONCLUSIONES.

Como conclusiones, la parte teórica y práctica de microprocesadores de la asignatura MYP se ha actualizado. Se imparte el tema de los microprocesadores utilizando un microprocesador actual como es el Pentium, cuando antes se impartían conceptos del 8086. El resultado ha sido una mayor motivación en los alumnos en las clases de teoría, debido a que se estudia un microprocesador actual.

Las prácticas se han actualizado a un entorno gráfico y se han diseñado nuevas prácticas que muestren los beneficios del lenguaje ensamblador.

Por tanto, utilizar un entorno más atractivo y un procesador más actual, ha ayudado a estimular al alumno en el estudio de un lenguaje tan importante como el ensamblador.

METODOLOGÍA PARA LA ENSEÑANZA DEL PROCESADOR *MULTICICLO*

Juan Luis Posadas[1], Antonio Robles[1]

[1]*Departamento de Informática de Sistemas y Computadores (DISCA)*
Escuela Universitaria de Informática, Universidad Politécnica de Valencia
E-mail: { jposadas, arobles } @disca.upv.es

RESUMEN: El presente trabajo ilustra una metodología para la enseñanza del procesador *multiciclo* a alumnos de primer curso de los estudios de Informática. Dicha metodología consiste en el diseño estructural de una Ruta de Datos basada en la utilización de buses y lógica triestado para, posteriormente, determinar las fases necesarias en la ejecución de las instrucciones en dicha Ruta de Datos. Asimismo, se diseña el circuito de control cableado que permite la ejecución de cada instrucción en varios ciclos de reloj. Mediante el uso de herramientas de desarrollo de presentaciones animadas, se describen un conjunto de animaciones didácticas que presentan de forma sencilla y visual la ejecución en varios ciclos de reloj de cada una de las instrucciones. El objetivo de dicho material es facilitar la exposición y comprensión del tema en el aula así como el estudio posterior que el alumno realiza. Esta metodología se aplica en la asignatura de Estructura de Computadores 1 de las titulaciones de Informática de la UPV, dentro del marco del Proyecto de Innovación Educativa nº 135 [1].

INTRODUCCIÓN

En los actuales planes de estudios para la obtención de los títulos de ITIS, ITIG e II en la Universidad Politécnica de Valencia (UPV), el estudio de la organización del computador y de sus principios básicos de funcionamiento es competencia de tres asignaturas de carácter troncal/obligatorio (tabla 1).

Asignatura	Curso	Cuatrimestre
Fundamentos de Computadores (FCO)	1º	A
Estructura de Computadores 1 (EC1)	1º	B
Estructura de Computadores 2 (EC2)	2º	A

Tabla1: Asignaturas troncales/obligatorias relativas a la organización del computador

En las asignaturas de Estructura de Computadores [2] se aborda el estudio de la organización del computador a través de las Unidades Funcionales que lo integran. En particular, en la asignatura de Estructura de Computadores 1 [3] se estudia la Unidad Funcional conocida como procesador o CPU (del inglés Central Processing Unit). La asignatura se divide en dos bloques: (1) estudio de la estructura e implementación del procesador monociclo, **multiciclo**, microprogramado y segmentado [4], diferenciando la Ruta de Datos y la Unidad de Control, y (2) el estudio de un elemento específico del procesador como es la Unidad Aritmético-Lógica. En la asignatura de Estructura de Computadores 2 [5] se estudian las Unidades Funcionales de Memoria y de Entrada/Salida.

En este trabajo se ilustra la experiencia de innovación docente llevada a cabo para facilitar la enseñanza del procesador multiciclo a los alumnos de primer curso de los estudios de Informática. Dicha experiencia se basa en una revisión metodológica de contenidos, apoyada en el empleo de herramientas tecnológicas de presentación en el aula. En particular, se propone la adaptación de la conocida Ruta de Datos de Patterson&Henessy [8] a una estructura de tres buses. El objetivo es que el alumno se familiarice con: (1) el concepto de bus como alternativa al empleo de multiplexores; (2) el empleo de recursos con lógica triestado; (3) el lenguaje de transferencia entre registros y (4) una estructura interna del procesador más cercana a lo que será la estructura de interconexión del procesador con memoria y dispositivos de E/S. Todo ello habrá de beneficiar el posterior proceso de aprendizaje de las técnicas de diseño basado en microprocesador.

El trabajo está organizado como sigue: en la sección 2 se pone de manifiesto la contribución de las nuevas tecnologías a la mejora de la enseñanza; en las secciones 3, 4 y 5 se describen los elementos de innovación que afectan a diferentes contenidos del tema de ejecución multiciclo; finalmente, en la sección 6 se exponen algunas conclusiones de la experiencia.

NUEVAS HERRAMIENTAS DE APOYO A LAS EXPOSICIONES EN EL AULA.

En los últimos años se ha venido haciendo un esfuerzo de acondicionamiento de las aulas tanto en la Facultad como en la Escuela de Informática (FI y EUI respectivamente) de la UPV. Este hecho ha posibilitado la incorporación y aplicación de nuevos recursos y tecnologías en las exposiciones que el profesorado realiza en las aulas de dichas instituciones.

Hasta el curso 1997-1998 en las aulas sólo se disponía de la tradicional e insustituible pizarra y del también tradicional proyector de transparencias. Estos recursos, aplicados en los métodos tradicionales de enseñanza, permiten abarcar cualquier tipo de exposición pero, sin embargo, pueden ampliarse con nuevos elementos para facilitar la comprensión de determinados temas. Durante el curso 1998-1999 el profesorado dispuso en todas las aulas por vez primera de un proyector o cañón de vídeo en color, con control a distancia y pantalla desplegable automáticamente, ordenador y conexión en red.

Estas herramientas han significado un apoyo importante en las exposiciones en el aula de gran número de asignaturas [6], dado que han permitido mostrar a los alumnos fotos, videos, simulaciones, animaciones y prácticas como refuerzo, ampliación o simplemente como medio de facilitar la comprensión de conceptos que sin el uso de estas herramientas el alumno se vería obligado a abordar en un plano más teórico y abstracto.

En este marco se ha tratado de introducir la técnica de ejecución multiciclo, presentada en este artículo, a los alumnos de primer curso (actualmente también se introducen otras técnicas como

la segmentación [4]). Para ello se ha confeccionado un material didáctico que incorpora animaciones en color orientadas a facilitar la comprensión del funcionamiento de un procesador multiciclo, con las que se permite realizar un seguimiento visual de las fases o ciclos necesarios en la ejecución de las instrucciones. Para el desarrollo de dicho material se ha empleado PowerPoint de paquete Office de MS para W98.

El uso de estas nuevas técnicas ha sido aceptado satisfactoriamente por los alumnos quienes apuestan por esta forma de comprensión visual y a la vez entretenida (al menos más amena que las tradicionales y conocidas clases magistrales). Sin embargo, no hay que perder de vista que las nuevas técnicas de presentación en el aula deben ser un complemento de los medios tradicionales como la pizarra y no un sustitutivo de los mismos. Un uso abusivo de los nuevos medios puede agotar al alumno y llegar a desmotivarle.

Con las nuevas técnicas, los alumnos también obtienen el nuevo material autoinstructivo accesible vía INTERNET que les facilita la tarea del estudio [9]. Todos los alumnos de la FI y EUI disponen en las aulas informáticas de acceso libre a Internet, cuenta personalizada de correo electrónico, impresoras, etc., por lo que pueden libremente obtener todo el material utilizado en las clases accediendo a las páginas WEB de las asignaturas en cuestión.

DISEÑO ESTRUCTURAL DE LA RUTA DE DATOS

Cuando se aborda el estudio del procesador multiciclo, el alumno ya conoce del tema anterior la estructura general de un procesador *monociclo* donde cada instrucción se ejecuta en un único ciclo de reloj. La estructura de dicho procesador se obtiene partiendo de una simplificación de la Arquitectura del Juego de Instrucciones del procesador MIPS R2000 [7] [8].

El tema comienza motivando el interés de introducir el procesador multiciclo como forma de reducir recursos en la Ruta de Datos, así como de reducción del tiempo medio de ejecución de las instrucciones. El estudio de dicho procesador multiciclo se plantea como un modo más realista de funcionamiento del procesador, donde lo habitual es que cada instrucción se ejecute en un cierto número de ciclos de reloj, dependiendo de su grado de complejidad. Con ello se busca un incremento en la velocidad de procesamiento y un mayor aprovechamiento de los recursos *hardware*. Seguidamente, se realiza y se muestra visualmente la construcción de una Ruta de Datos multiciclo basada en multiplexores a partir de la Ruta de Datos monociclo (fig.1). A continuación, se le hace ver al alumno la necesidad de emplear más de un ciclo de reloj para ejecutar las instrucciones en la nueva Ruta de Datos. En ella, pese a la reducción del número de recursos, es posible que una misma instrucción haga uso repetido de éstos a lo largo de los diferentes ciclos de reloj en que se ejecuta. Una vez el alumno ha asimilado el nuevo concepto de Ruta de Datos multiciclo, se propone una transformación de la misma a base de sustituir los multiplexores por buses y de dotar de capacidad de lógica triestado a los recursos conectados a los mismos. La nueva Ruta de Datos multiciclo (fig.2) se corresponde con la clásica estructura de tres buses en la que, con objeto de garantizar una completa equivalencia a nivel de fases de ejecución con la Ruta de Datos basada en multiplexores, se introducen un par de conmutadores (enlaces de bus) para unir y/o dividir buses en un momento dado. El diseño de la Ruta de Datos se completa identificando las señales de control necesarias para indicar el camino que han de seguir los datos y las operaciones que han de realizarse con ellos (lecturas, escrituras, etc.).

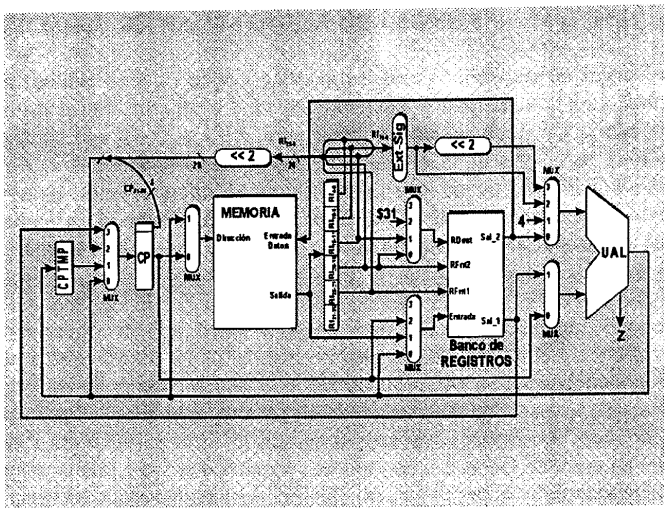


Fig.1: R.D. basada en multiplexores

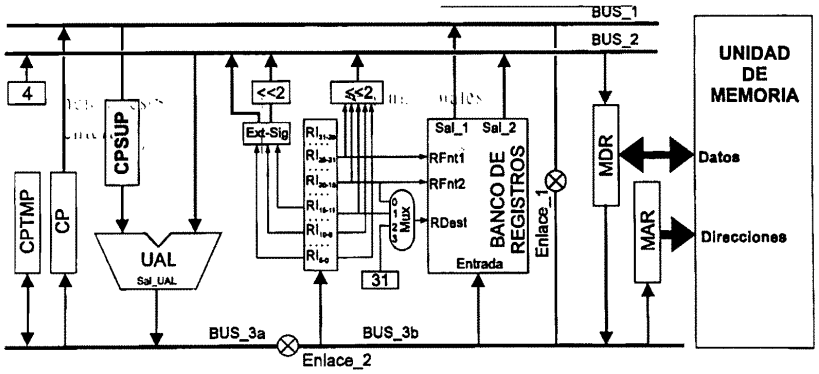


Fig.2: R.D. basada en buses con lógica triestado

FASES EN LA EJECUCIÓN DE LAS INSTRUCCIONES

Tras la construcción de la Ruta de Datos y la identificación de líneas de control, y como paso previo a la descomposición de la ejecución de las instrucciones en fases, se establecen los costes temporales asociados a las operaciones realizadas con los diferentes recursos de la Ruta de Datos (UAL, Banco de Registros, Memoria, etc.), así como la duración del ciclo de reloj (fig.3). El alumno puede observar cómo influye el retardo de cada componente en la definición de las fases de ejecución.

Para cada fase de ejecución de cada instrucción se especifican las operaciones que en ella tienen lugar, empleando para ello el lenguaje de transferencia entre registros, así como la activación de las correspondientes líneas de control.

A tal fin, han sido desarrolladas una serie de animaciones que presentan de forma visual y sencilla la ejecución de las distintas instrucciones en varios ciclos de reloj.

Realizando un estudio detallado, se obtiene y se presenta de forma animada, a partir de la ruta de datos basada en buses con lógica triestado, la división en fases de la ejecución de cada una de las instrucciones.

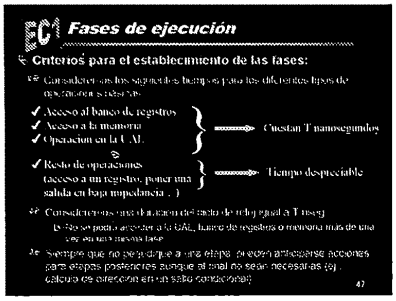


Fig.3: criterios temporales

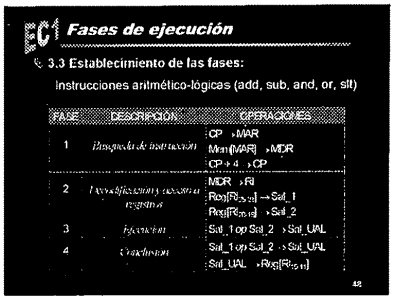


Fig.4: fases ejecución aritmético-lógicas

En la figura 4 pueden observarse las cuatro fases en las que queda dividida la ejecución de las instrucciones aritmético-lógicas, las operaciones realizadas en cada una de ellas y los recursos utilizados. Esta división se obtiene y se presenta de forma animada (fig.5, 6, 7 y 8) ajustándose cada fase a la duración del ciclo de reloj.

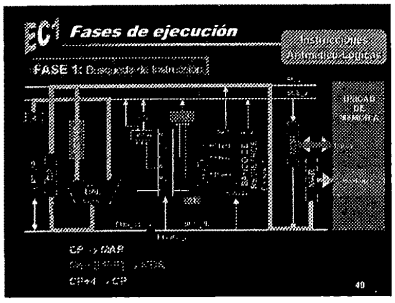


Fig.5: fase 1 aritmético-lógicas

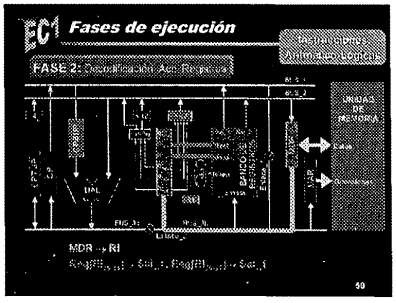


Fig.6: fase 2 aritmético-lógicas

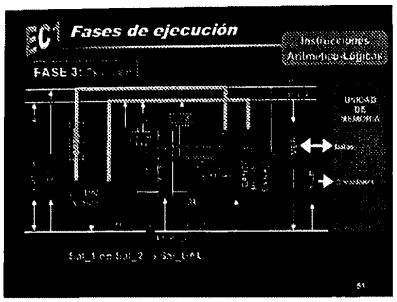


Fig.7: fase 3 aritmético-lógicas

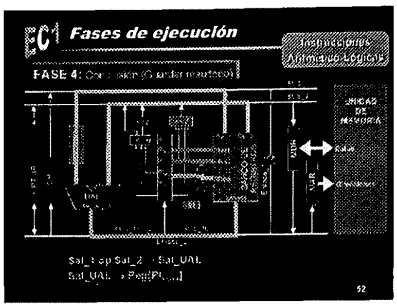


Fig.8: fase 4 aritmético-lógicas

UNIDAD DE CONTROL MULTICICLO

Una vez establecida la división en fases de cada una de las instrucciones soportadas, se aborda la implementación cableada del circuito Unidad de Control mediante dos métodos: el método del contador de fases y el método de la tabla de estados. En temas posteriores se realiza la implementación del mismo circuito de forma microprogramada.

Independientemente del método empleado, previamente se realiza una identificación de las señales de control necesarias y activación de las mismas en cada una de las fases de ejecución de las instrucciones. Mediante la proyección de las animaciones desarrolladas resulta fácil identificar en cada etapa las señales requeridas.

CONCLUSIONES

La experiencia docente llevada a cabo vuelve a poner de manifiesto [5] la idoneidad del empleo de materiales didácticos que incluyan presentaciones animadas, en el ámbito de la estructura del computador; siempre que éstos se utilicen como complemento de los métodos tradicionales, y no como sustitutivo de los mismos.

La facilidad para la comprensión visual de los conceptos, inherente a este tipo de material didáctico, permite un mayor aprovechamiento de cada sesión liberando tiempo para una mayor profundización y discusión en el aula de los conceptos.

Uno de los aspectos positivos que los alumnos destacan del material es la posibilidad de recrear una y otra vez la exposición del profesor en el aula; al disponer del mismo en la *web* de la asignatura [9].

REFERENCIAS

[1] A. Pont y M.L. Rico. Plan de innovación docente en Arquitectura y Estructura de Computadores. III Jornadas Univ. sobre Innovación Educativa en las Enseñanzas Técnicas.
[2] A. Robles y J. Sahuquillo. Integración teórico-práctica en la enseñanza de Estructura de Computadores: un enfoque metodológico. II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías. ICE (UPM). 1996
[3] J.A. Gil, L. Peñalver, A. Pont y A. Robles. Estructura de Computadores (Vol.1). SPUPV-1999

[4] J.L. Posadas, A. Robles. Innovación Educativa en la Enseñanza del Procesador Segmentado. JENUI 99
[5] A. Pont, J.A. Gil, M. Martínez, J. Molero, A. Robles y A. Rodas. Estructura de Computadores (Vol. 2). SPUPV-1996
[6] J. Real, J.A. Gil y A. Robles. Uso de nuevos medios de apoyo a las exposiciones en el aula. III Jornadas de Enseñanza Universitaria de Informática. 1997
[7] J.L. Hennessy, D.A. Patterson. Arquitectura de Computadores, un enfoque cuantitativo. New York. Mc. Graw Hill. 1993
[8] D.A. Patterson, J.L. Hennessy. Organización y Diseño de Comput. Mc. Graw Hill. 1995
[9] WEB de la asignatura con animaciones disponibles: http://poseidon.disca.upv.es/ec1/index.html

Win8086: Una herramienta para el análisis de la frecuencia de utilización de las instrucciones 8086

José L. Poza Luján[1], Álvaro Doménech Pujol[2], Pedro López Rodríguez[3]

[1]*Escuela Universitaria de Informática. U. P. de València*
e-mail: jopolu@doctor.upv.es
[2]*Escola Universitària d'Informàtica. U. P. de València*
e-mail :alvaro@disca.upv.es
[3] *Facultad de Informática. U. P. de Valencia*
e-mail: plopez@disca.upv.es

Resumen: En el presente artículo se presenta una herramienta desarrollada para la obtención y posterior análisis de la frecuencia de utilización de las instrucciones Intel 8086. Esta herramienta permite generar las trazas en sistemas basados en DOS y Windows, y permite analizarlas gráficamente de forma cómoda.

1.- INTRODUCCIÓN.

a) Motivación

En las asignaturas de arquitectura de los computadores, sean genéricas o específicas del PC, es interesante disponer de una herramienta que permita estudiar el uso que los programas hacen del juego de instrucciones de la familia Intel. Si con esta herramienta se pueden monitorizar tanto los programas propios del usuario, escritos en un lenguaje de alto nivel, como los programas ya compilados disponibles, su ámbito de aplicación se extiende hacia las asignaturas de compiladores o de algorítmica.

b) Herramientas actuales

Actualmente existen métodos de monitorización en los que predomina el aislamiento sobre el sistema a analizar [Egg 90].

También existen herramientas software que monitorizan al propio sistema en el que se ejecutan, una de ellas es la herramienta Etch para sistemas Intel basados en Windows32 [Rom98], en la que se aprovechan las facilidades de Windows NT para la monitorización. Las herramientas que funcionan bajo Windows NT tienen la dificultad de no encontrar fácilmente este sistema entre el alumnado.

También se puede encontrar otra herramienta, IDTrace [Pier 94] que no monitoriza, sino simula la actuación del sistema. Estas herramientas tienen el problema de que el alumno/a no interactúa directamente en el sistema, esa interactividad es muy importante en el proceso de enseñanza-aprendizaje.
Otro de los mayores problemas con los que nos encontramos, es el que está en los archivos de trazas generados, en ocasiones el tamaño de los archivos los hace inviables para su análisis, por lo que se debe reducir su tamaño en gran medida [JoHa 92].

b) La herramienta Win8086

La herramienta desarrollada se divide en dos bloques principales, la generación de trazas, y el análisis de las mismas.

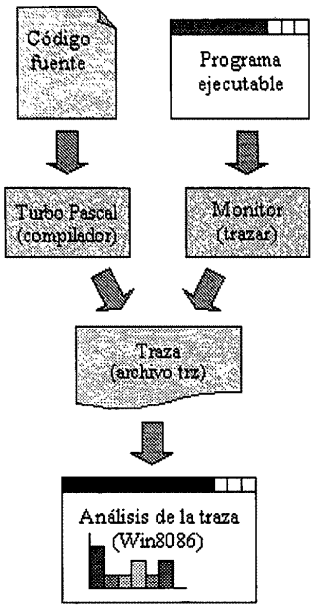


Ilustración 1 Esquema de bloques y operación

Una cuenta de Win8086 es un estadillo en el que se contabilizan el número total de instrucciones procesadas, las frecuencias de los códigos de operación ejecutados, la frecuencia de cada modo de direccionamiento aplicado, la distribución de direcciones accedidas, la tasa de aciertos de las instrucciones de salto condicional y la distribución de distancias cubiertas por estas últimas. Hay dos métodos de obtener una traza. El primero se aplica a casos en que se dispone de un código fuente en alto nivel y consiste en la inserción de llamadas a procedimientos de monitorización en el texto del programa. El segundo es la ejecución de un programa ejecutable activando previamente el sistema de monitorización.

El análisis de las trazas generadas se hace por medio de la aplicación *Win8086* que proporciona vistas de informes o de gráficos. Todo el proceso de trabajo se puede ver en la Ilustración 1.

2.- GENERACIÓN DE TRAZAS.

El esquema empleado para la monitorización se puede apreciar en la siguiente secuencia. En la Ilustración 2 se aprecia la ejecución normal de una instrucción, desde el punto de vista del programador y sin entrar en los detalles de cómo la UCP procesa la misma.

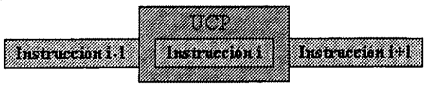


Ilustración 2 Secuencia habitual de ejecución

En la Ilustración 3 se aprecia el método de monitorización empleado: modificando un bit de la palabra de estado del procesador, éste provocará la interrupción de traza en cada ciclo de instrucción. Previamente se habrá instalado en la tabla de vectores la rutina de servicio asociada, que se encarga de actualizar los contadores necesarios.

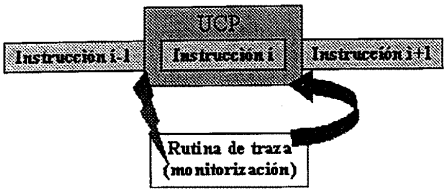


Ilustración 3 Ejecución monitorizada

La rutina de servicio activada accede a la instrucción a través de la pila, como se puede ver en la Ilustración 4.

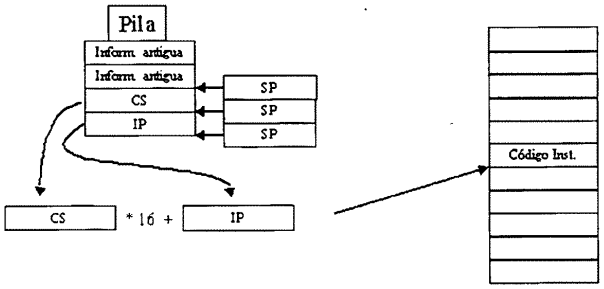


Ilustración 4 Acceso a la información de la instrucción procesada

a) Monitorización de código fuente.

La monitorización del código fuente se realiza por medio de la importación de una unidad de Turbo Pascal: *trazas*.
Esta unidad proporciona los siguientes servicios:

Servicios de tiempo. Para poder obtener los tiempos empleados en la monitorización.
Servicios de traza. Permiten configurar, comenzar y concluir la traza.
Servicios de resultados. Permiten guardar los resultados en un archivo de texto ASCII que será leído por la herramienta de análisis.
La referencia a las funciones de los servicios se hace insertando directamente en el código. Considérese el siguiente ejemplo en que el programa principal llama al procedimiento *whetstone*.

```
Begin
      { Lee número de iteraciones de whetstone }
      write('Número de iteraciones...');
      Readln(iteraciones);
      {Llamada al procedimiento}
      whetstone(iteraciones);
end;
```

Para analizar la ejecución de *whetstone* bastará con insertar referencias a los servicios de traza definidos en *trazas* antes y después de la llamada:

```
uses trazas;
```

```
...
begin
      { Lee número de iteraciones de whetstone }
      write('Número de iteraciones...');
      Readln(iteraciones);
      hola;
      pontraza;
      {Llamada al procedimiento}
      whetstone(iteraciones);
      quitatraza;
      adios;
      guardatraza('whet.trz','Traza de whetstone');
end.
```

El procedimiento *hola*, se encarga de insertar en la tabla de vectores de interrupción las correspondientes funciones de monitorizado

Los procedimientos *pontraza* y *quitatraza*, determinan el comienzo y finalización del monitorizado. Y el procedimiento *adios*, se encarga de restaurar la tabla de vectores de interrupción y dejar el sistema con la misma configuración que antes de la monitorización.

La combinación en la inserción de los procedimientos en el código, puede dar lugar a monitorizar exclusivamente zonas de código determinadas.

La modificación del código se puede realizar directamente en el editor de pascal que corresponda o bien por medio de una ventana de edición de la aplicación Win8086 y llamando al compilador en línea instalado, de una forma transparente al usuario/a.

b) Monitorización de programas ya compilados

Para la monitorización de programas ejecutables, se proporciona un sencillo programa *trazar* cuya labor es activar los procedimientos de monitorización y a continuación lanzar el programa a monitorizar.

Esta llamada se puede realizar desde una sesión en línea de comandos de DOS, la configuración de la llamada se debe hacer por medio de los parámetros del programa, como habitualmente se hace en cualquier aplicación de consola:

C:\>trazar /s2 /m100000,20000 c:\command.com

En la línea de comandos anterior se monitoriza una sesión de DOS, con rutina de monitorización en ensamblador (s2) y con intervalos de análisis de memoria de la dirección 100000 decimal a la posición 20000 decimal (/m).

También desde una ventana de la aplicación *Win8086* en la que se escoge la aplicación a monitorizar, con los parámetros de monitorización configurables en la correspondiente caja de diálogo.

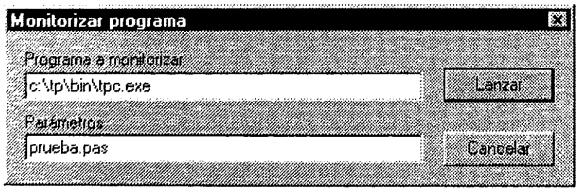


Ilustración 5 Ventana de monitorización de ejecutables.

3. ANÁLISIS

Una vez generada la traza, por cualquiera de los dos caminos posibles, se crea un archivo en el que está toda la información obtenida de la ejecución.

Cabe destacar que el programa puede abrir varias trazas diferentes al mismo tiempo, y varios gráficos a la vez, tanto de una misma traza como de varias.

Esto último es muy útil desde el punto de vista pedagógico, ya que facilita la comparación de resultados.

El archivo es de texto (ASCII), con la información dividida en líneas, para facilitar la forma de acceso al mismo y la interpretación por otro tipo de programas que no sea el propuesto en este proyecto.
Debido al preproceso de la monitorización, el archivo no ocupa más de un Kbyte, por lo que no plantea los problemas clásicos de los archivos de trazas [JoHa 92].
Ese archivo es abierto y procesado por la aplicación Win8086. La información que del mismo se obtiene es la siguiente:

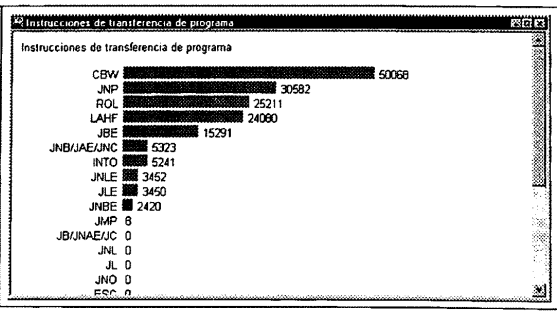
- Instrucciones procesadas.
- Bifurcaciones realizadas.
- Direcciones de memoria accedidas.
- Modos de direccionamiento empleados.

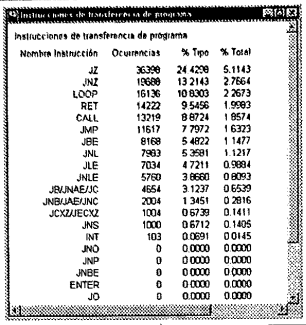
A continuación se expone un ejemplo de cada uno de los análisis.

a) Instrucciones procesadas

Las instrucciones procesadas, se pueden visualizar en modo gráfico de barras o listado de instrucciones. Se permite ordenar o filtrar las mismas por medio de la misma clasificación que Intel realiza [Intel 97]:

- Transferencia de datos.
- Operaciones aritméticas.
- Manejo de bits.
- Manejo de cadenas.
- Transferencia de programa.
- Control del procesador.
- Coma flotante.
- MMX.

Gráficos de instrucciones		Muestra las instrucciones que se han procesado, con un gráfico de barras

Listado de instrucciones			Muestra un listado de instrucciones que se muestran con los tantos por cien comparativos de las ocurrencias de las instrucciones.

Instrucciones de transferencia de programa

Nombre Instrucción	Ocurrencias	% Tipo	% Total
JZ	36398	24.4208	5.1143
JNZ	19688	13.2143	2.7664
LOOP	16136	10.8303	2.2673
RET	14222	9.5456	1.9983
CALL	13219	8.8724	1.8574
JMP	11617	7.7972	1.6323
JBE	8168	5.4822	1.1477
JNL	7883	5.3691	1.1217
JLE	7034	4.7211	0.9884
JNLE	5760	3.8660	0.8093
JBJNAE/JC	4654	3.1237	0.6539
JNBJAE/JNC	2004	1.3451	0.2816
JCXZ/JECXZ	1004	0.6739	0.1411
JNS	1000	0.6712	0.1405
INT	103	0.0691	0.0145
JNO	0	0.0000	0.0000
JNP	0	0.0000	0.0000
JNBE	0	0.0000	0.0000
ENTER	0	0.0000	0.0000
JO	0	0.0000	0.0000

b) Bifurcaciones

Las bifurcaciones se pueden analizar por medio de un gráfico de barras en los que se pueden ver cuántas instrucciones y tanto por cien han ocurrido, y cuántas de cada sentido (adelante o atrás), han resultado exitosas y cuántas fallidas.

La precisión del gráfico puede variarse como se ve a continuación:

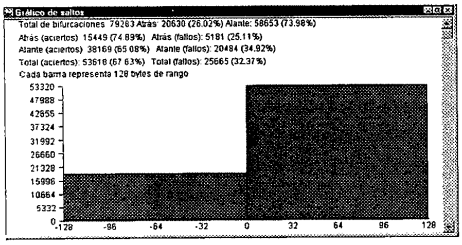


Ilustración 6 Rango de representación = 2

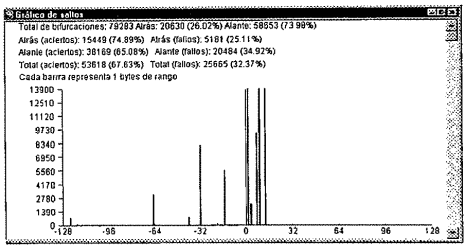


Ilustración 7 Rango de representación = 256

c) Rango de direcciones de memoria

Las direcciones de memoria accedidas por la aplicación, se pueden monitorizar por rangos y especificando la precisión del mismo, en la visualización se puede ver el número de accesos en cada rango.

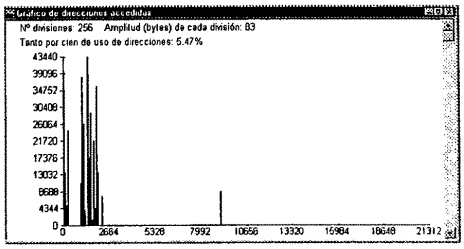


Ilustración 8 Rango de representación = 512

d) Modos de direccionamiento empleados

Se pueden visualizar los modos de direccionamiento empleados, en un gráfico de barras en los que se muestran los mismos, clasificados bajo el criterio de Intel.

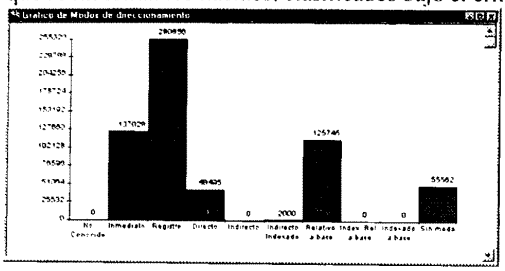


Ilustración 9 Gráfico de modos de direccionamiento.

4.- CONCLUSIONES.

Con esta aplicación, se ha facilitado el estudio de la arquitectura de un sistema, del compilador empleado y el comportamiento de los mismos ante un programa. Aparte, se tiene desarrollada una herramienta fácil de usar y que muestra los resultados de forma agradable.

Con ella se puede analizar fácilmente, a coste casi nulo, el comportamiento del juego de instrucciones de la familia de procesadores Intel 80x86.

Este juego crece para cada nueva versión. Sin embargo, desde el primitivo 8086, el empleo del juego de instrucciones es escaso. Es decir, el 90% de las instrucciones empleadas, suponen tan solo, el 10% de instrucciones disponibles.

Así mismo, por medio de la aplicación se justifica que el estudio de mejoras en los saltos, el empleo de la memoria caché, y la optimización en los modos de direccionamiento suponen un buen avance en el rendimiento del sistema.

Por medio de la herramienta, el alumno/a puede crear por sí mismo/a la monitorización, y llevar a cabo el análisis, con las ventajas que ello implica no sólo en el aprendizaje del comportamiento del procesador, sino en la implicación del alumnado en la práctica, al disponer de una herramienta fácil de emplear.

5.- REFERENCIAS.

[Rom 98] ***Ted Romer y otros.*** "Instrumentation and optimization of Win32/Intel Executables Using Etch" University of Washington 1998.

[Pier 94] ***Jim Pierce and Trevor Mudge***, *"IDtrace—A Tracing Tool for i486 Simulation"*.Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS), Enero 1994

[JoHa 92] **Eric E. Johnson y Jiheng Ha**, "PDATS. Lossless Address Trace Compression For Reducing File Size And Access Time".

[Egg 90] **S. Eggers, D. Keppel, E. Koldinger y H. Levy**. "Thecniques for Efficient Inline Tracing on a Shared-Memory Multiprocessor" Proc. 1990 ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems,Bouldeer, CO May 1990, pp. 37-47

[Intel 97] "Intel architecture software developer's manual volume #2 (instruction set reference)" *Intel Corporation, 1997 order #243191-001*

[Dom 98] **Domenech Pujol, Álvaro, Real Sáez, Jorge Vicente**. "Organización del PC: del hardware a la aplicación". SPUPV 1998.

[Poza 99] **José Luis Poza, Álvaro Doménech, Pedro López**. Obtención de trazas. Estudio, y desarrollo de una herramienta de análisis. P .F .C. DISCA – UPV, 1999.

USO DEL LENGUAJE ENSAMBLADOR PARA PRÁCTICAS DE Aritmética

Julio Sahuquillo[1], Juan Carlos Cano[1], José Flich[1] y Salvador Petit[1].

[1]*Departamento de Informática de Sistemas y Computadores*
E. U. I.–Universidad Politécnica de Valencia–
e_mail: { jsahuqui, jucano, jflich, spetit} @ disca.upv.es

RESUMEN: La Unidad Aritmético Lógica (ALU) junto con la unidad de control (UC), son las unidades funcionales más relevantes del procesador.

El estudio de la UC es con diferencia el más abordado por los autores del área de conocimiento, desplazando a un segundo plano el estudio de la aritmética ha sido siempre un tema marginal. Lo cual queda patente tanto en clases magistrales como en las prácticas de laboratorio.

Este trabajo presenta un nuevo enfoque en las prácticas de laboratorio de la ALU, enfatizando el aspecto práctico sobre el teórico de cara a una formación más sólida del futuro Ingeniero en Informática.

1.- INTRODUCCIÓN.

La aritmética del computador se estudia en las asignaturas de Estructuras de Computador, dentro de los planes de estudio de Ingeniería Informática e Ingeniería Técnica en Informática. El hecho de que el tema se encuentre ligado con la lógica digital, ha provocado que las prácticas de laboratorio se realicen (en caso de que se realicen) mediante simuladores de circuitos digitales.

En líneas generales, las prácticas consisten en la implementación de circuitos aritméticos (sumadores y restadores). En opinión de los autores, este tipo de prácticas poco aportan a la formación del futuro ingeniero, debido a que los conceptos teóricos ya se han estudiado en clase, y en el laboratorio los alumnos simplemente implementan, utilizando el simulador, el mismo circuito (mismos componentes y conexiones) que el estudiado en clases magistrales.

El principal concepto en los temas de sumadores es la composición de circuitos. Para esto, se utilizan circuitos pequeños para la composición de circuitos más grandes con análogo funcionamiento. Por ejemplo, la realización de un circuito sumador convencional o CPA (siglas de Carry Propagated Adder) de 16 bits a partir de CPAs menores (de 8 ó 4 bits). La comprensión del funcionamiento de dichos circuitos es relativamente sencilla, y fácilmente asimilable por el alumno en clases magistrales; aún así, en muchas universidades se insiste en realizar prácticas de laboratorio, desde nuestro punto de vista poco o nada pragmáticas.

En este trabajo se propone utilizar el lenguaje ensamblador de cara a realizar y poner en práctica la composición de circuitos. Por tanto, se presenta un enfoque práctico para acercar al alumno hacía esta temática amen de formarle de cara al futuro ejercicio profesional. Debido a que en nuestra Universidad los alumnos estudian el MIPS R2000 en el primer cuatrimestre, en este trabajo se utiliza el citado lenguaje de cara a abordar la implementación práctica.

2. PRESENTACIÓN DEL PROBLEMA: EJEMPLO

Como ejemplo del enfoque propuesto, en este trabajo se aborda la implementación mediante lenguaje ensamblador de la realización de operaciones aditivas de dos números de 64 bits (o múltiplos de 32) utilizando la ALU de un procesador real de 32 bits.

Las CPUs tradicionalmente proporcionaban el valor de los indicadores de resultado en un registro especial. Estos indicadores se han estudiado en clase. Se pide, como ejercicio complementario que el alumno obtenga los indicadores de Z(Zero), N (Negative), C(Carry) y O(Overflow), mediante el lenguaje ensamblador.

El alumno debería intentar obtener él solo la solución que pasa por sumar independientemente la parte baja (bits del 31 al 0) de la parte alta (bits del 63 al 32). Como se aprecia en la Figura 1, se necesita tener en cuenta el acarreo del bit de mayor peso de la parte baja para sumárselo al resultado de la suma de la parte alta.

Aunque este estudio se realiza mediante el MIPS R2000, el ejercicio se puede particularizar para cualquier lenguaje ensamblador conocido por el alumno.

Solución tradicional

La Figura 1 muestra el diagrama de bloques de un CPA de 64 bits implementado a partir de dos CPAs de 32 bits conectados en serie. Como se aprecia el acarreo saliente (C_{31}) del CPA que suma los 32 bits de menor peso se conecta con la entrada de acarreo del CPA que suma los bits de mayor peso.

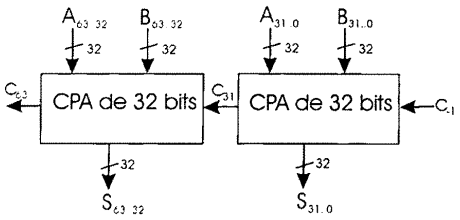


Figura 1. Diagrama de bloques de un CPA de 64 bits a partir de CPAs de 32 bits.

El esquema es sencillo y fácil de entender por lo que creemos que para su comprensión no es necesario montar el circuito en el laboratorio ya sea físicamente o mediante un simulador.

Solución propuesta: objetivos perseguidos

Teniendo en mente el problema descrito, el ejercicio persigue que se implemente la solución mediante el uso del lenguaje ensamblador con miras a que el alumno:

1. Madure en el aspecto práctico, utilizando la funcionalidad de la ALU para realizar operaciones con números más grandes (mayor número de bits) que los circuitos que en ella se integran.

2. Perciba y aprecie la importancia del lenguaje ensamblador, ya que minimiza la distancia entre la teoría y práctica.

Que duda cabe, que los aspectos citados avivan y despiertan, más si cabe, el interés del alumno por la materia.

3. DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

El problema o práctica a realizar se basa en implementar un programa que nos permita sumar dos números A y B de 64 bits, con una ALU de sólo 32, teniendo en cuenta que:

- Los datos a sumar se almacenarán inicialmente en memoria a partir de la dirección 0x10000000.
- Se supone que primero se almacena el dato A y a continuación el dato B.
- Se desea almacenar el resultado de la operación aritmética a continuación del dato B.

4. RESOLUCIÓN

Definición de los datos en memoria y cargado en registros

La ubicación de los datos en memoria debe ser el mostrado en la

Tabla 1.

Dirección	Contenido
0x1000 0000	$A_{31..0}$
0x1000 0004	$A_{63..32}$
0x1000 0008	$B_{31..0}$
0x1000 000C	$B_{63..32}$

Tabla 1. Ubicación de los datos en memoria.

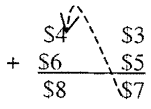
Para esto, las directivas de datos asociadas son:

```
    .data 0x10000000
A:      .word A31..0, A63..32     # Reservamos espacio para el sumando A
B:      .word B31..0, B63..32     # Reservamos espacio para el sumando B
R:      .space 8                  # Reservamos espacio para el resultado
```

Para poder operar con la ALU, hay que cargar los datos en los registros. Colocaremos la parte baja de A en $3 y en $4 la parte alta. B lo almacenaremos de forma que su parte baja esté en $5 y la parte alta en $6. Por último, el resultado lo obtendremos en $7 y $8 (parte baja y alta respectivamente). Una posible solución es:

```
__start:  la $4,A          # Cargamos la dirección de A para poder
          lw $3, 0($4)     # cargar primero la parte baja y luego
          lw $4, 4($4)     # cargar la parte alta.
          la $6,B          # Hacemos lo mismo con el segundo sumando
          lw $5, 0($6)
          lw $6, 4($6)
```

Esquemáticamente, se desea realizar:



Obtención de la suma

La suma de dos dobles palabras (números de 64 bits) se puede dividir en cuatro partes. En la primera, se sumarían las dos palabras de menor peso. A continuación se calcularía el acarreo de esta suma. Luego sólo nos queda sumarle este acarreo a la parte alta de A para finalizar sumando este resultado a la parte alta de B. Veamos el código:

```
add $7,$3,$5      # Hacemos la suma de las palabras de menos peso
sltu $8,$7,$3     # Ponemos en $8 el acarreo
                  # Hay acarreo si el resultado es menor que un sumando
add $8,$8,$4      # Sumamos la parte alta de A con el acarreo
add $8,$8,$6      # Sumamos el resultado anterior con la parte alta de B,
```

Obtención de los indicadores de resultado

Indicador de Cero (Z)

El indicador de cero se activa si el resultado de la operación es cero.

```
or $9,$8,$7   # En $9 tendremos distinto de 0 si hay algún 1 en la parte
              # alta del resultado $8 o en la parte baja $7
```

```
sltiu $9,$9,1   # Ponemos $9 a 1 si el resultado anterior es menor que 1
                # (es igual a 0). En caso contrario, pondremos un 0.
                # Por tanto $9=Z
```

Indicador de signo (N)

Este indicador se utiliza cuando se trabaja con números con signo, y se activa cuando el resultado es negativo. El valor del indicador es igual al valor del bit más significativo del resultado.

```
srl $9,$8,31    # Desplazamos el registro que contiene la palabra de mayor peso
                # 31 posición para chequear el valor de su bit de mayor peso: 63
                # El bit correspondiente es el de menor peso del registro $9
                # Por tanto $9=N
```

Indicador de Desbordamiento (Overflow)

El desbordamiento nos indica que el resultado de una operación necesita más bits de los que se dispone para almacenarlo. Internamente, se implementa con la XOR del bit, de acarreo del último y penúltimo bit. Sin embargo se puede calcular de otra forma (en una operación aditiva): si sumamos dos números de distinto signo no puede haber desbordamiento, sino que sólo se produce desbordamiento cuando se suman dos números positivos y se obtiene uno negativo o cuando se suman dos negativos y se obtiene un resultado positivo. Por tanto, calcularemos el indicador para sumas de números de igual signo.

```
xor $9,$4,$6    # En el bit más significativo del resultado obtenemos un 1 si los
                # sumandos tienen distinto signo y 0 si tienen el mismo signo.
srl $9,$9,31    # Colocamos el bit que nos interesa en la posición de menos peso
bne $9,$0,fin   # Si tienen signos distintos, $9=1 (no hay desbordamiento)
xor $9,$8,$6    # Si tienen el mismo signo verificamos si el resultado tiene
                # el mismo signo que los sumandos ($9=1)
srl $9,$9,31    #Colocamos el bit que nos interesa en el lugar de menos peso
fin:            # $9=O
```

Indicador de acarreo (C)

Este indicador se activa cuando al sumar los dos bits de mayor peso se genera acarreo ("nos llevamos uno"). La interpretación de su valor varía en función de si se trabaja con números con signo o números sin signo.
Cuando se trabaja con números sin signo un valor igual a uno indica que se ha producido desbordamiento.
Al realizar la suma de números sin signo se activa cuando el resultado sea menor que cualquiera de los sumandos, el código correspondiente es:

```
sltu $9,$8,$6
bne $9,$0,fin   # Hay acarreo ($9=1) si el resultado es menor que un sumando
bne $8,$6, fin  # Si el resultado es mayor que un sumando (salta) no
                #hay acarreo ($9=0)
sltu $9,$7,$5   #AQUÍ LLEGAMOS SI SON IGUALES
                # se comprueba las partes menos significativas y si
                # la del resultado es menor, habrá acarreo. En caso contrario, no.
fin:            # C=$9
```

Programa completo

Con todo lo expuesto anteriormente, el programa completo resultaría:

```
        .data 0x10000000
A:      .word A31..0, A63..32    # Reservamos espacio para el sumando A
B:      .word B31..0, B63..32    # Reservamos espacio para el sumando B
R:      .space 8                 # Reservamos espacio para el resultado
        .text
__start: la $4,A                 # Cargamos la dirección de A para poder
        lw $3, 0($4)             # cargar primero la parte baja y luego
        lw $4, 4($4)             # cargar la parte alta.
        la $6,B                  # Hacemos lo mismo con el segundo sumando
        lw $5, 0($6)
        lw $6, 4($6)
        add $7,$3,$5             # Hacemos la suma de las palabras de menor peso
        sltu $8,$7,$3            # Hay acarreo si el resultado es < que un sumando
        add $8,$8,$4             # Sumamos la parte alta de A con el acarreo
        add $8,$8,$6             # Sumamos también la parte alta de B.
        la $10, R                # Cargamos la dirección del resultado para poder
        sw $7, 0($10)            # almacenar primero la parte baja y
        sw $8, 4($10)            # en la dirección siguiente la parte alta del resultado
```

Generalización para k>2

Hasta ahora se ha visto cómo sumar dos números de 64 bits (2 palabras de 32 bits) con un procesador que sólo puede trabajar con 32 bits. Sin embargo, lo estudiado se puede extender para cualquier numero de k*32 bits. A continuación, se presenta el código para sumar dos números para cualquier valor de k. Con las siguiente restricciones:

Los números a sumar se almacenan en A y B en formato *little endian*, separando cada palabra por comas. Ej: A: .word 5678,1234.

En la línea (2) debemos sustituir 4*k por el resultado de multiplicar 4 por el número de bytes que ocupan los sumandos. Ej: .space 8.

En la línea (3) tenemos que cambiar k por el número de palabras (de 32 bits) de los sumandos. Ej: K: .word 2.

Cada registro se utiliza con los fines especificados en la Tabla2.

Registro	Contenido
$3	Dirección de la palabra de A que estamos operando.
$4	Dirección de la palabra de B que estamos operando.
$5	Dirección de la palabra donde hay que almacenar el resultado de la palabra en curso.
$6	Número de palabras que nos quedan por operar.
$7	Registro temporal donde llevamos la suma de la palabra en curso y el acarreo.
$8	Registro temporal usado para leer de memoria de los sumandos.

Tabla 2. Uso de los registros.

```
# Suma.S
# Calcula la suma de dos números de k*32 bits
        .data
   A:   .word A1,A2...Ak # Reservamos memoria para el primer sumando
   B:   .word B1,B2...Bk # Reservamos memoria para el segundo sumando
   R:   .space 4*k       # (2) Reservamos memoria para el resultado
   K:   .word k          # (3) Le indicamos al programa el tamaño (en palabras)
                         # de cada número
        .text
__start :la  $3,A        # Cargamos el puntero al primer sumando
        la  $4,B         # Cargamos el puntero al segundo sumando
        la  $5,R         # Cargamos el puntero al resultado
        la  $6,K
        lw  $6,0 ($6)    # Cargamos el numero de palabras que hay que operar
        or  $7,$0,$0     # Ponemos a cero el acarreo, puesto que en la primera
                         # suma no nos llevamos nada de la suma anterior.
bucle:  lw  $8,0 ($3)    # Leemos la palabra a operar del primer sumando
        add $7,$7,$8     # Sumamos la palabra del primer sumando al acarreo
                         # anterior
        lw  $8,0 ($4)    # Cargamos la palabra del segundo sumando
        add $7,$7,$8     # Sumamos el segundo sumando a la suma parcial
                         # anterior
        sw  $7,0 ($5)    # Almacenamos el resultado obtenido en memoria
        sltu $7,$7,$8    # Calculamos el acarreo para la siguiente palabra
        addi $3,$3,4     # Apuntamos a la siguiente
        addi $4,$4,4     # palabra a operar tanto de los dos sumandos como del
        addi $5,$5,4     # resultado.
        addi $6,$6,-1    # Ahora ya nos queda una palabra menos por operar
        bne  $6,$0,bucle # Si aún quedan palabras por sumar, volvemos para
                         # sumar la siguiente palabra.
```

5.- CONCLUSIONES

En este trabajo se ha presentado un nuevo enfoque de realizar las prácticas de circuitos aritméticos, considerando el lenguaje ensamblador como herramienta.

En opinión de los autores este nuevo enfoque motivará a los alumnos no sólo en el estudio del lenguaje ensamblador sino a consolidar sus conocimientos de arquitectura de computadores. Además este acercamiento entre la teoría y la práctica, incentiva al alumno en la utilización de este lenguaje. Por último señalar, que la mayor parte de los alumnos disponen de PCs lo que les habilita para continuar el trabajo en sus casas y potenciar, más si cabe, el "aprender haciendo".

ENSEÑANZA DE SISTEMAS OPERATIVOS CON NACHOS: UNA EXPERIENCIA ESPAÑOLA

José Miguel Santos Espino[1], Carmelo Rubén García Rodríguez[2]

[1] *Departamento de Informática y Sistemas, Universidad de Las Palmas de Gran Canaria*
correos: {jomis,rgarcia}@dis.ulpgc.es

RESUMEN: En este documento se describe la experiencia de uso del sistema operativo didáctico Nachos en la enseñanza de la materia de Sistemas Operativos, dentro de las ingenierías informáticas de la Universidad de Las Palmas de Gran Canaria. Se describen los criterios que han motivado la elección de esta herramienta y se presentan los resultados tras los dos primeros años de implantación.

1.- PRESENTACIÓN

Los autores de este artículo somos responsables docentes de la materia de Sistemas Operativos en las titulaciones informáticas de la Universidad de Las Palmas de Gran Canaria. Con motivo de la reciente implantación de los nuevos planes de estudios (1998), tuvimos que enfrentarnos a una drástica reducción de créditos en la asignatura correspondiente. Hasta ese momento, la asignatura tenía asignada una carga de 15 créditos, 9 teóricos y 6 prácticos. Las prácticas adiestraban a los alumnos en programación de sistemas bajo UNIX, concurrencia mediante IPC e implementación de módulos de sistema operativo, utilizando un *software* de confección propia.

En las nuevas ingenierías, la materia de Sistemas Operativos se convierte en una única asignatura cuatrimestral, con la mitad de créditos (4'5 teóricos, 3 prácticos). Afrontamos el reto de mantener unos objetivos de formación similares a la situación anterior, pero sin que los estudiantes resulten sobrecargados de trabajo. Con este fin hemos explorado distintas alternativas en la realización de las prácticas y hemos encontrado la mejor opción en un sistema operativo didáctico llamado Nachos.

2. LAS NUEVAS PRÁCTICAS

a) Objetivos y enfoque metodológico

Consideramos que el estudiante de esta asignatura de Sistemas Operativos debe adquirir un conocimiento general sobre estas dos materias:

- Estructura, funcionamiento y diseño de los sistemas operativos
- Fundamentos de la concurrencia y la sincronización entre procesos

Esos son, por tanto, los conocimientos que deben reforzar las actividades prácticas. La cuestión es cual es el mejor enfoque para conseguir este objetivo. Las diversas propuestas de actividades practicas en la enseñanza de Sistemas Operativos suelen encajar en uno de estos tres modelos:

1. Simulación de algoritmos y políticas
2. Implementación de módulos de un sistema operativo
3. Programación de sistemas (llamadas al sistema UNIX, etc.)

Nosotros nos decantamos por el segundo enfoque, puesto que es el único que abarca los tres ambitos de la materia: estructura, funcionamiento y diseño del sistema operativo. Las simulaciones no ejercitan el conocimiento de la estructura, y la programación de sistemas sólo aproxima al alumno a los servicios del sistema operativo, a su interfaz. El gran inconveniente del segundo enfoque es la mayor dificultad para el estudiante.

b) ¿Qué herramienta elegir?

Una vez asumido el enfoque de implementación de módulos, hemos de decidir cuál es la herramienta más apropiada. Desde luego, hay que proporcionar al estudiante un sistema operativo básico con el cual trabajar. Así el alumno no tiene que malgastar tiempo en construir un *software* desde cero. Por otra parte, el estudiante se acostumbra a manejar código ajeno.

Se puede emplear como sistema de trabajo un sistema operativo real, como Linux o Mach. En este caso, el estudiante ha de consumir mucho tiempo estudiando un código de gran tamaño y repleto de optimizaciones que lo hacen ilegible. A esto se añaden las dificultades prácticas que entraña la modificación el sistema, ya que hay que reinstalar el sistema operativo, rearrancar el equipo, etc., por no hablar de la enorme dificultad de depuración.

Para reducir estos costes, desde hace décadas se han desarrollado sistemas operativos *didácticos*, cuya arquitectura es deliberadamente simple, están bien documentados y son relativamente fáciles de modificar. Un ejemplo clásico es Minix [1], que corre sobre hardware PC-Intel. Aunque Minix es más llevadero que un núcleo real, sigue siendo demasiado voluminoso para nuestros estudiantes y no resuelve el problema de la modificación y depuración. Este problema radica en que se trata de un sistema que se instala sobre el hardware nativo.

Otros sistemas didácticos se basan en la *emulación* de una máquina virtual. El sistema operativo administra un hardware virtual más sencillo, con lo cual la arquitectura del software se simplifica. Además, si la máquina virtual corre como programa de usuario sobre el sistema operativo nativo, la depuración y modificación del sistema emulado es tan simple. Este enfoque ha sido empleado satisfactoriamente en España [2], [3]. Y es éste el tipo de sistemas que nosotros buscamos.

2.- EL SISTEMA OPERATIVO NACHOS

De entre toda esta gama de alternativas, hemos optado por el sistema didáctico Nachos, descrito en [4] y [5]. Se trata de un núcleo desarrollado a lo largo del año 1992 por W. Christopher, S. Procter y T.E. Anderson para la docencia de Sistemas Operativos en la Universidad de Berkeley, EUA. El código fuente y la documentación original de Berkeley están disponibles públicamente en Internet. Desde su nacimiento, el uso de Nachos se ha extendido por decenas de universidades norteamericanas. También se emplea en varias universidades iberoamericanas,

desde Guatemala hasta Chile. No nos consta ninguna experiencia docente con Nachos en España.

a) Características de Nachos

Nachos es un pequeño núcleo escrito en C++, que suministra servicios básicos de manejo de procesos, memoria, archivos, entrada/salida y redes. Los servicios de Nachos se han simplificado al máximo, consiguiendo un núcleo completamente funcional en unas pocas páginas de código en C++. El núcleo Nachos se ejecuta como un proceso UNIX y administra los recursos de una máquina virtual basada en un procesador MIPS.

El código fuente de Nachos es bastante legible. Abundante en comentarios, organiza los servicios en clases de C++ con interfaces muy claras y predecibles. En cuanto al lenguaje, prescinde de características como la sobrecarga, excepciones, *templates*, etc. en aras de la simplicidad. Gracias a ello sólo es necesario un conocimiento mínimo de C++ para poder entender el código fuente. El volumen del código es bastante pequeño: los estudiantes han de manejar realmente apenas 2500 líneas de código (incluyendo comentarios). El emulador ocupa otras 2500 líneas, pero no hay que conocer su implementación.

La emulación de la máquina virtual simula el repertorio completo del procesador MIPS (salvo instrucciones de coma flotante). Los programas de usuario Nachos han de estar escritos en código MIPS, pero se puede utilizar un compilador de lenguaje C (ej. gcc). Así pues, los estudiantes pueden usar un lenguaje de alto nivel ya conocido por ellos para probar los servicios del núcleo Nachos.

b) Tareas propuestas en Nachos

Nachos es un sistema operativo deliberadamente mutilado: en general, sólo proporciona mecanismos básicos de sincronización, recepción de eventos, traducción de direcciones de memoria, etc. No tiene implementados servicios de llamadas al sistema, ni memoria virtual. El estudiante es quien tiene que rellenar las carencias. Con este fin, el autor del Nachos propone cinco trabajos de programación: 1)hilos y sincronización; 2)llamadas al sistema y multiprogramación; 3)TLB y memoria virtual; 4)archivos; 5)redes y sistemas distribuidos. El código fuente del Nachos está organizado teniendo en cuenta esta división de trabajos.

De los cinco trabajos, el primero consiste en implementar cerrojos y variables condición (Nachos sólo implementa hilos y semáforos) e implementar programas concurrentes de prueba. En el segundo trabajo, hay que implementar la captura de excepciones y llamadas al sistema, y tras ello implementar la carga de programas de usuario y su ejecución multiprogramada.

c) ¿Por qué Nachos?

Nachos encaja a la perfección con nuestro modelo didáctico (implementación de sistemas operativos). Nachos nos resulta especialmente atractivo porque, siendo extremadamente simple y pequeño, abarca todos los ámbitos de interés en la materia de Sistemas Operativos. Hay que reconocer el acierto de los autores en destilar la quintaesencia de un sistema operativo completo. Por lo demás, está avalado por su uso desde hace varios años en decenas de universidades extranjeras.

Es software de uso gratuito y se puede ejecutar sobre GNU-Linux, con lo cual no hay que invertir ni una peseta en *software* para trabajar con él. Esta misma razón, unido a su pequeño tamaño, facilitan que los alumnos puedan realizar las prácticas en casa si lo desean. Afortunadamente, esta característica la comparten muchos otros sistemas.
Como Nachos se ejecuta como un proceso UNIX, no exige ser instalado en exclusiva en el *hardware*, como ocurriría con el Minix. Por tanto, puede haber varias copias de Nachos ejecutándose en una misma máquina, con el consiguiente ahorro de recursos. Nachos tiene la gran ventaja sobre Minix y Linux de correr sobre un hardware emulado, que se puede controlar al milímetro. Por ejemplo, se puede contabilizar la ocurrencia de fallos de página, interceptar excepciones, etc.

Como se está emulando un procesador MIPS, los alumnos pueden escribir programas de usuario tan complejos como quieran en lenguaje C. Estos programas son traducidos a código MIPS con un compilador y luego ejecutados por Nachos. Por ejemplo, podemos escribir dos versiones de un programa de multiplicación de matrices y analizar cuántos fallos de página genera cada uno. Esta facilidad no es frecuente en otros sistemas basados en emuladores.

d) Inconvenientes potenciales

Desde un principio, encontramos dos características que podrían dificultar al alumno el aprendizaje de Nachos. En primer lugar, el código y la documentación originales están escritos en inglés, lo cual es una dificultad añadida para los estudiantes castellanoparlantes. En segundo lugar, la documentación suministrada es insuficiente para que el alumno pueda comprender la herramienta y mucho menos para guiarle en las tareas que debe realizar.

3.- LA EXPERIENCIA EN LAS PALMAS DE GRAN CANARIA

Nachos fue implantado en el curso 1998/99, el primer año en que se impartía la nueva asignatura de Sistemas Operativos.

a) Perfil de los estudiantes

El primer año, la población de alumnos contenía aproximadamente un 25% de personas provenientes del antiguo plan de estudios, muchos de los cuales habían realizado parte de las prácticas anteriores. Los estudiantes deberían conocer el lenguaje C, aunque la realidad es que el nivel promedio es bastante elemental. La gran mayoría no ha utilizado sistemas UNIX anteriormente.

b) Adaptación del software

Fue necesario adaptar el código original del Nachos para que pudiera compilar en el entorno Linux y retocar algunos fuentes para adaptarlos al reciente estándar ISO de C++. También hubo que generar un compilador cruzado de C a MIPS, a partir del GCC.

c) Metodología docente

De los cinco trabajos propuestos en Berkeley, se han escogido sólo los dos primeros: el trabajo de hilos y sincronización y el trabajo de llamadas al sistema y multiprogramación. El trabajo sobre redes queda fuera del ámbito de la asignatura impartida. Hemos tenido que renunciar –por ahora- a los trabajos sobre memoria virtual y archivos porque a nuestro juicio no se pueden realizar con los créditos prácticos disponibles.

La documentación original de Nachos es insuficiente para que el alumno entienda bien las tareas propuestas. Para soslayar esta dificultad, hemos atacado en dos frentes. Primero, elaboramos una documentación en castellano que describe con más profundidad la estructura del Nachos. La descripción del Nachos está basada sobre todo en la guía de la Universidad de Duke [6].

Segundo, hemos descompuesto cada uno de los trabajos originales en una serie de ejercicios más simples que orientan el trabajo del alumno. A su vez, cada ejercicio se estructura en tres fases: 1)preguntas previas sobre conceptos que el alumno ya debe conocer antes de iniciar el trabajo; 2)explicación de conceptos necesarios para entender el ejercicio; 3)propuesta de trabajo. De esta forma, el estudiante dispone de un itinerario bastante claro para realizar los trabajos y, sobre todo, entender lo que está haciendo.

En las primeras clases prácticas, los alumnos reciben adiestramiento sobre las herramientas de desarrollo, es decir, el entorno Linux y el lenguaje C++ (sólo los elementos imprescindibles para entender Nachos). Tras esta fase, las horas prácticas se dedican alternativamente a dos tipos de actividades: a)explicar las subtareas y proponer su realización; b)supervisar y conducir el trabajo de los alumnos.

d) Resultados

La experiencia ha sido altamente positiva. Los estudiantes consiguen en general completar los trabajos con éxito y con menor esfuerzo que en años anteriores. Nos hemos sorprendido gratamente de la velocidad con que los estudiantes consiguen aprender las interfaces de Nachos. Eso sí, los estudiantes que se lanzan desde el primer día a escribir código suelen acabar atascados al poco tiempo, por no tener clara la arquitectura del sistema.

Hemos observado que, gracias a Nachos, los alumnos adquieren un más que notable conocimiento sobre los fundamentos estructurales de un sistema operativo: sistema de interrupciones, llamadas al sistema, modo dual de operación, etc. Esto se refleja positivamente en los resultados de los exámenes teóricos. En nuestra opinión, esto se debe a que los trabajos con Nachos les proporcionan una experiencia directa de las entrañas del núcleo de un sistema operativo.

e) Opinión de los alumnos

Nos resultaba de especial interés la opinión de los estudiantes repetidores, quienes habían experimentado los dos modelos de prácticas. El comentario unánime fue que las nuevas prácticas resultaban mucho más sencillas, tanto en el planteamiento como en el desarrollo; bastantes repetidores recordaban con espanto lo complicado que era depurar una aplicación concurrente en UNIX.

En el segundo año se han realizado encuestas acerca de la dificultad de los elementos del Nachos y sobre su utilidad en la asignatura. El 49% de los alumnos considera que las prácticas le han ayudado bastante o mucho a comprender los conceptos teóricos de la asignatura. Los alumnos valoraron la dificultad de tres elementos: el manejo de información en inglés (17'3%), el lenguaje C++ (20%) y el esfuerzo para comprender los trabajos (46'7%). [Entre paréntesis están los porcentajes de alumnos que consideran cada elemento muy difícil]. Como puede apreciarse, el inglés y el C++ no representan problema para los estudiantes; sin embargo, casi la mitad de los encuestados encuentran bastante difícil comprender los trabajos en su conjunto. En este punto creemos que la tutorización de los profesores resulta fundamental para que el alumno adquiera una visión general de la arquitectura del sistema Nachos.

f) Un punto débil: el modelo de concurrencia

El planificador de Nachos no es expulsivo (*preemptive*). Un hilo sólo abandona la CPU cuando se bloquea o cuando la cede voluntariamente. Esta política reduce las posibles combinaciones en la ejecución de un conjunto de hilos. Por este motivo, es difícil que el alumno verifique que sus programas concurrentes son correctos.

Por otra parte, en el primer trabajo los alumnos comienzan implementando unas herramientas que jamás han usado (cerrojos y variables condición), con lo cual ellos no son completamente conscientes de lo que están programando. Esto se suple con el siguiente paso, que consiste en usar esas herramientas en programas multihilo, pero como aprendizaje del *uso* de herramientas de sincronización no es ni de lejos la mejor opción didáctica.

g) El futuro

Nuestro principal reto es ser capaces de encajar en las 30 horas disponibles las prácticas de memoria y archivos. Esto tendrá que lograrse a costa de simplificar los dos trabajos actuales, tarea en la que nos encontramos actualmente. Además de ello, vamos a revisar la primera tarea (concurrencia) para que tenga más eficacia didáctica: modificaremos el planificador de Nachos para hacerlo expulsivo a voluntad y alteraremos el planteamiento del trabajo de sincronización.

5.- CONCLUSIONES

El sistema operativo didáctico Nachos se ha implantado eficazmente como herramienta práctica en una ingeniería informática. En escasamente treinta horas prácticas es posible adiestrar a los alumnos en Linux y C++ y además conseguir que completen los dos primeros trabajos.

El código fuente de Nachos está organizado de forma muy legible y fácil de aprender. En caso de no conocer C++, es bastante fácil para el alumno adquirir el nivel necesario para realizar los ejercicios.

Nachos sirve para introducir al estudiante en la concurrencia y sincronización entre procesos, aunque de forma limitada.

La documentación original es insuficiente para un óptimo desempeño del alumno. Es necesario suplirla con documentación propia. La acción tutorial también se considera importante.

Se observa que el aprendizaje práctico con Nachos repercute positivamente en la adquisición de los conocimientos teóricos.

6.- REFERENCIAS

[1] *Operating System Design and Implementation.* Andrew. S. Tanenbaum. Prentice-Hall, Englewood Cliffs. 1987.
[2] *DSO: Herramienta para el aprendizaje de sistemas operativos.* S. Sánchez, D. Meziat, I. García y M.J. Onievas. II Jornadas nacionales de innovación en las enseñanzas de las ingenierías. 1997.
[3] *Ús i descripció dels nivells del sistema operatiu ONION v.4.0.* C. Barrado, L. Doreste y otros. Departament d'Arquitectura de Computadors, Universitat Politècnica de Catalunya. 1991.
[4] *The Nachos Instructional Operating System.* W. Christopher, S. Procter, T. E. Anderson. Proceedings of the 1993 Winter USENIX Conference. 1993.
[5] *Operating System Concepts*, 4th edition. A. Silberschatz, P. Galvin. Addison-Wesley. 1994.
[6] *A Road Map Through Nachos.* T. Narten, Duke University. 1995.
Nuestra web sobre Nachos está en http://sopa.dis.ulpgc.es/nachos

SISTEMA DE SIMULACIÓN PARA EL DISEÑO LÓGICO Y DE COMPUTADORES.

Juan Pérez Rodríguez[1], Benedicto Gómez Díaz[1], Josep Mª Ribes i Bonet[1], Francesc Escudero i Costa[1]

[1]*Enginyeria i Arquitectura La Salle*
e-mail: juanp@salleURL.edu, se05050@salleURL.edu
jmribes@salleURL.edu, sisco@salleURL.edu

RESUMEN: En el presente artículo se presenta el programa de apoyo a la docencia que ha sido desarrollado por profesores de la asignatura de Introducción a los Ordenadores. Dicho programa permite la simulación de circuitos lógicos, ya sean básicos o de gran complejidad. La finalidad del simulador es la de ayudar a comprender el funcionamiento de las diferentes arquitecturas de computadores que se estudian en la asignatura y para ello se introduce en la parte práctica como herramienta de trabajo habitual. Las diferentes características del simulador permiten el diseño de arquitecturas nuevas, la simulación lógica de los diseños y el tratamiento de los bancos de memoria.

1.- INTRODUCCIÓN.

Desde la asignatura de Introducción a los Ordenadores, que se encuentra presente en el primer curso de todas las carreras de ingeniería de la Escuela Universitaria La Salle, se ha querido dar un alto grado de importancia a la realización de prácticas en el marco de la asignatura. Para ello se decidió desarrollar un software capaz de simular los circuitos lógicos que se enseñan a principios de curso con el Álgebra de Boole [1], los elementos de memoria que se explican a medio curso y las estructuras de un computador[3], que se explican a finales de curso.

La razón principal para desarrollar este software de simulación es la de poder contar con una herramienta versátil en la parte práctica de la asignatura que permita al alumno comprender en profundidad aquellos conceptos abstractos que ve en la teoría y que no es capaz de imaginarse. Para conseguir este propósito es cierto que existen paquetes de software en el mercado, pero la mayoría quedan limitados en cuanto a la posibilidad de modificar la arquitectura a estudiar o la forma de ver la evolución del sistema o, simplemente, son paquetes muy generales donde esta simulación es una parte más, poco desarrollada. Con un software propio se puede crear la herramienta adecuada al temario explicado y se dispone de la posibilidad de modificar en un futuro, con cierta facilidad.

Es muy fácil que esta idea de un paquete de simulación crezca en expectativas hasta intentar abarcar temarios o aspectos de asignaturas posteriores a la de Introducción a los Ordenadores. Y es por ello, que en su diseño se tiene en cuenta posibles ampliaciones y simulación de nuevas arquitecturas, así como la incorporación de un juego de instrucciones. Sin embargo, este artículo

tratará de la primera versión del simulador, en la que muchas cosas no estarán disponibles. Sólo cubrirá las necesidades de un primer curso de carrera universitaria.

No debe olvidarse el marco en que se realiza este paquete de simulación; se trata de una ingeniería común para informática, telecomunicaciones, electrónica y otras especialidades. No seria lógico dejar de lado, en este proyecto, las necesidades específicas de cada especialidad ya que el simulador estaría sentenciado al fracaso. Por ello, se intentan integrar características de simplificación de funciones lógicas (propias de una ingeniería electrónica) con métodos informáticos como Quine-McCluskey[4], la introducción de funciones por tablas de Karnaugh[5], la creación de componentes (muy útil en Informática para el diseño de nuevas arquitecturas), la simulación de memorias o el control del retraso sufrido por las señales (muy interesante en el campo de las comunicaciones). En resumen, un conjunto de características que en mayor o menor grado acabarán siendo útiles en todas las especialidades.

A lo largo de este artículo se describirán los datos más relevantes del simulador y se intentará dar una explicación de cómo se llevó a cabo el diseño del mismo. El siguiente apartado dará una idea del alcance del proyecto al exponer sus características principales y cuales son las que se han contemplado para una futura versión. Tras las características se exponen los criterios utilizados para seleccionar como debía ser el simulador y se cierra la ponencia con las conclusiones de la utilización del software de simulación, valorando el resultado final obtenido y su introducción en la asignatura.

2.- CARACTERÍSTICAS DEL SIMULADOR.

a) Condicionantes.

Las características del sistema en su primera versión son limitadas debido a una cuestión de tiempo, ya que el simulador debe estar funcionando a partir de setiembre de este mismo año para poder realizar pruebas de simulación. Ello obliga a que la estructura del programa este muy bien pensada para poder realizar ampliaciones en el futuro.

Otro condicionante, que no deja de ser una característica, es el entorno de trabajo para el alumno. Se debe pensar en un programa de fácil manejo y de una adaptación muy rápida. De esta manera se intenta evitar el rechazo por parte del alumno.

b) Características actuales.

Para hablar de las características del simulador es necesario recordar que se pensó para alumnos de primero de carrera y que, por tanto, debe encararse a un tipo de usuario que no esta acostumbrado a utilizar este tipo de simuladores. Como se ha dicho anteriormente, ello obliga a que la característica principal sea un entorno fácil de utilizar y muy intuitivo, con ordenes rápidas y sencillas como poner componente, realizar conexiones, simular, etc.

Centrándonos en el alcance de lo que puede hacer el programa, se puede empezar por las características de un diseño lógico. Para ello permite realizar diseños utilizando componentes existentes en las librerías básicas o crear componentes con la descripción de pines y el comportamiento que tiene. Permite crear nuevas librerías, incluir componentes que provienen de diseños ya realizados (encapsulando el diseño en un componente), crear componentes que responden a tablas de verdad o tablas de Karnaugh y cuya función lógica se obtiene a partir de la simplificación lógica a dos niveles mediante el método de Quine-McCluskey[4]. En estos

diseños se permite la introducción de fuentes de señal (onda cuadrada, continua o una forma de onda digital concreta, cuyos valores se establecen manualmente o con un fichero) y puntos de control que más adelante aparecerán en la simulación para poder estudiar su evolución en el tiempo. De la misma manera se podrían definir las entradas y salidas de un circuito y, de esta manera, poderlo convertir en un componente que se utilizará en posteriores diseños.

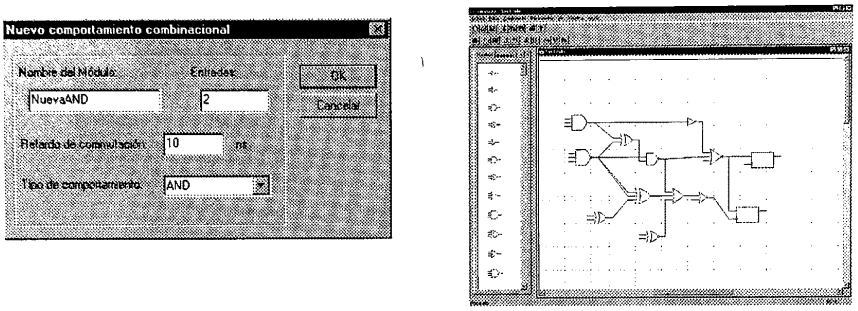


Figura 1. A. Proceso de creación de un nuevo componente a partir de la definición de los pines y su comportamiento. B. Ejemplo de un diseño lógico con puertas elementales y biestables.

Por otro lado, todo circuito o diseño que se realiza puede ser simulado y estudiado a partir de la evolución de las señales y de los valores que adquiere. Para realizar esta parte de forma visible se dispone de la pantalla de simulación en la que se muestran las señales escogidas en el diseño, con diferentes zooms y marcadores temporales. Tal pantalla se muestra en la figura 2.

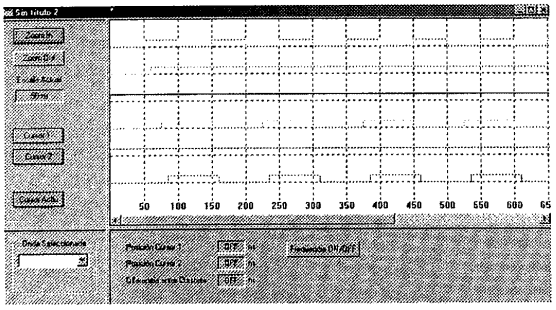


Figura 2. Pantalla con la simulación de las señales escogidas en el diseño.

El siguiente paso lleva a hablar de las características de los elementos de memorización. Básicamente, se dispone de los diferentes tipos de biestables y de las memorias estándares de mercado. Otros elementos intermedios, como pueden ser los registros se pueden construir a partir de estas librerías mínimas (un registro será una agrupación de biestables). En ellos se pueden definir los retrasos que sufren las señales y algunas propiedades más. Lo importante es que a nivel de memorias se permite rellenar esta con los valores de un fichero, con los valores introducidos manualmente o con los valores que introduzca el circuito. Sea de la forma que sea, se ofrece una herramienta para observar la evolución de los valores introducidos. Y esta característica es muy importante de cara a la simulación de arquitecturas de computadores.

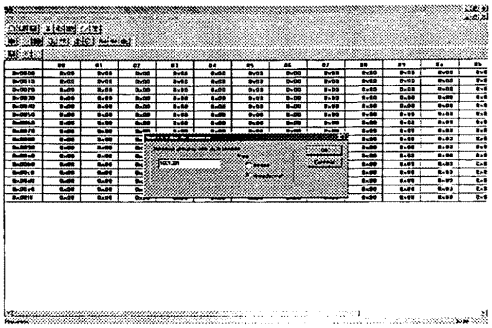


Figura 3. Pantalla en la que se observa el trabajo con la memoria y los valores que se introducen.

Las características que más hacen de este simulador una herramienta para la enseñanza son las que permiten la simulación y el diseño de pequeños núcleos de computadores. Para ello, se partirá de que será necesario disponer de una ALU, de un banco de registros y de una unidad de control (UC) que estén creadas para tal propósito.

Siendo así, el programa permite diseñar la UC de tipo microcableada a partir de la maquina de estados a la que tiene que responder. Permite la definición de las señales de entrada, de las señales de salida, los ciclos de reloj que puede ocupar y los opcodes que debe reconocer. Por ahora, el programa no permite el diseño de una UC microprogramada. Respecto a la ALU solo se permite la configuración de las instrucciones que reconoce, las operaciones a realizar y los tiempos que tarda en realizar dichas operaciones.

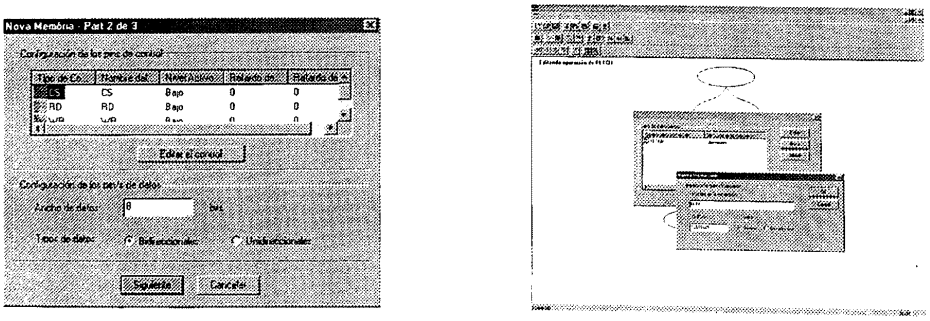


Figura 4. Detalle del diseño de una Unidad de Control, con su máquina de estados y el conjunto de señales de entrada y de salida.

De toda esta última parte, cabe destacar que el programa permite unir todos los elementos para que el diseño funcione como el núcleo de un pequeño computador, en el que se puede observar la evolución de los valores de los registros importantes, de los datos que pasan por el datapath y de la evolución de la memoria. Y se tiene acceso a toda esta información porque es el usuario el que ha creado la máquina que esta simulando y, por tanto, esta viendo el circuito.

c) Características futuras.

Entre las características futuras destacan una interficie capaz de entender VHDL para pequeños diseños o circuitos lógicos. Esta interficie será capaz de leer datos y de escribir datos en este lenguaje. La utilidad más importante será llevar los diseños a otros programas a través de un formato estándar.

Se intentará mejorar toda la parte de diseños de unidades aritmético-lógicas, para conseguir una mayor flexibilidad en los parámetros que se pueden configurar y en las operaciones a realizar. Tampoco se descarta la posibilidad de implementar ALUs donde se definan los procesos de cálculo, y por tanto, los tiempos que tardará por cada operación en función de la forma de calcular los datos.

En la misma línea, se debe buscar el diseño de una unidad de control de tipo microprogramada. De esta manera, la parte de diseño de arquitecturas de computadores quedaría bastante completa, teniendo en cuenta que se buscan pequeñas arquitecturas de carácter docente y no arquitecturas excesivamente complicadas.

Debe mejorarse la forma en que se ven los resultados de la simulación, no a nivel lógico de las señales, sino para las memorias, buses y grandes cantidades de datos.

Finalmente, destacar la posibilidad futura de cargar un programa en código máquina, obtenido por el porceso de compilación de un software estándar de mercado, sobre la memoria del sistema para que sea ejecutado sobre la arquitectura diseñada, siempre y cuando coincida el juego de instrucciones y los opcodes de ambas partes (programa y arquitectura). Actualmente, las instrucciones se introducen manualmente y se simula el comportamiento de dicha instrucción.

3.- ELECCIÓN DE CRITERIOS.

Los criterios para elegir las características del programa final se basan en los criterios de la asignatura impulsora y en las necesidades que esta tiene de cara a poder afianzar los conocimientos teóricos adquiridos.

Es fácil observar que para entender un circuito lógico básico no es necesario tener un potente simulador. Sin embargo, cuando se pide al alumno entender el movimiento de datos dentro de un computador, su evolución temporal o determinar donde se encuentran los problemas de una arquitectura concreta, se necesita poder trabajar con la máquina, ver lo que hace y poder modificar los procedimientos a seguir. Esto es algo impensable con una máquina real, debe utilizarse una simulación que sea lo más parecida a la realidad y suficientemente versátil como para permitir cambios.

Existen dos criterios principales que se han seguido para diseñar el simulador y que responden a las finalidades de la asignatura, diseño lógico a bajo nivel y con un alto grado de escalabilidad y simulación de computador con acceso a todos los puntos del sistema.

El primer criterio lleva a realizar un simulador que permita la realización de pequeños circuitos, su simulación y su posterior integración como componente discreto en grandes diseños.

El segundo obliga a la realización de los componentes de un computador por partes para tener acceso a todos los puntos. Por tanto se debe permitir el diseño de una unidad de control, de una ALU y de todo el conjunto de registros e interconexiones que forman la arquitectura.

Ello implica definir los criterios para diseñar este nuevo tipo de elementos tan específicos. El principal es tener una herramienta potente pero a la vez que sea ágil, para que el trabajar con ella no sea una cosa pesada.

4.- CONCLUSIONES.

Por ahora es demasiado pronto para sacar conclusiones acerca del grado de utilidad del simulador, ya que empezará a aplicarse con los alumnos a partir del próximo curso. Sin embargo, la experiencia con otros simuladores y las impresiones recogidas de los alumnos, hacen pensar que esta herramienta será de gran utilidad.

También se debe pensar que el simulador será entendido como un complemento a la teoría que se explica en clase. Teoría que muchas veces es abstracta y no se acaba de comprender porque no es fácil de imaginar. Con esta herramienta se resuelven muchos de estos problemas y permite un "margen de maniobra" para adaptarse a aquello que se explica en la asignatura.

En cuanto al resultado de la idea original se puede concluir que la primera versión responde a casi un 90 % de lo que se pretendía en un principio. Sin embargo, si tuviera que ser evaluada actualmente, con las nuevas ideas que se han propuesto, seguramente respondería a un 40-50% de lo que se pretende. Ello indica que existirán futuras versiones mejoradas y que poco a poco será una buena herramienta de trabajo para los alumnos.

5.- REFERENCIAS.

[1]. 'Principios Digitales'. Tercera Edición. Roger L. Tokheim. Mc Graw-Hill. 1995.

[2]. 'Introducción al diseño lógico digital'. Primera Edición. John P. Hayes. Addison-Wesley Iberoamericana. 1996.

[3]. 'Computer System Architecture'. Tercera Edición. M. Morris Mano. Prentice Hall. 1993.

[4]. 'Prime implicants by Quine-McCluskey Tabulational Method'. http://www.capitol-college.edu/academics/ugrad/classes/oldho/el304/quine.htm

[5]. 'The Map Method for Synthesis of Combinational Logic Circuits'. M. Karnaugh. Transactions of the AIEE, Part I, vol. 72, págs. 593-599. 1953.

PROPUESTA PARA LA ENSEÑANZA DE ARQUITECTURA DE COMPUTADORES

Lenin Lemus, Maria Angeles Pinar

Universidad Politécnica de Valencia
e-mail:lemus@disca.upv.es, mapinar@disca.upv.es

Resumen: A finales del siglo XX, los procesadores han tenido una influencia decisiva en el quehacer humano y los podemos encontrar en la mayoría de los dispositivos electrónicos utilizados por el hombre; sobra decir que su arquitectura ha requerido de una evolución rápida y constante. Se prevé que en la primera década del siglo XXI los procesadores tengan un papel aún más determinante en el diseño de sistemas electrónicos.

John Hennesy, expone que hasta ahora nos hemos dedicado a aumentar el rendimiento de los procesadores sin preocuparnos de su estructura interna, por lo que ahora se debe pensar en la estructura interna de los procesadores.

Por otro lado, los sistemas empotrados se están desarrollando mediante técnicas de diseño concurrente (codiseño) utilizando particionamiento vertical aunque se prevé que en la próxima década se utilice particionamiento horizontal, debido una vez más a la utilización de procesadores.

Con lo dicho hasta ahora, se puede ver que en el transcurso de la primera década del siglo XXI, se requiera la implementación de herramientas de diseño de aplicaciones informáticas basadas en procesadores a medida.

Al profesorado encargado de impartir asignaturas relacionadas con la Arquitectura de Computadores en la Universidad Politécnica de Valencia (UPV) nos parece coherente que el futuro Ingeniero en Informática esté preparado para afrontar el diseño tanto de herramientas de diseño como de aplicaciones informáticas basadas en procesadores a medida. Sin embargo se nos plantea el siguiente problema:

¿Cómo es posible lograr presentar a los futuros ingenieros las bases de la arquitectura de computadores, la aplicación de los procesadores para resolver problemas reales y las tendencias futuras sin crear un caos en los planes de estudio?

1.- INTRODUCCION.

A finales del siglo XX una parte muy significativa de los aparatos electrónicos se basan en procesadores.

La forma de diseñarlos ha evolucionado a una velocidad vertiginosa y aunado a este hecho se debe tomar en cuenta que los procesos de fabricación también han mejorado hasta el punto que hoy en día es posible integrar en una sola pastilla de silicio varios millones de transistores.

La tendencia futura es la de integrar sistemas empotrados en una sola pastilla (SoC). Esto es tener uno o varios procesadores mas dispositivos de E/S en un solo circuito integrado con el fin de abaratar costes de producción, aumentar la fiabilidad de los sistemas, disminuir el consumo de potencia y ocupar el mínimo espacio físico. Un buen ejemplo de este tipo de sistemas lo constituyen los teléfonos móviles.

Sin embargo, los sistemas empotrados no son los únicos que hacen amplio uso de los procesadores. Estos también están siendo más y más utilizados en computadores personales, en los computadores de bolsillo (palm tops), y, por supuesto, en el diseño de dispositivos para comunicaciones. Debido a esta amplia gama de aplicaciones, la arquitectura de los procesadores es variada.

En lo que respecta a los sistemas empotrados se utilizan arquitecturas orientadas a tener altas prestaciones en cálculo matemático (procesamiento digital de imágenes, etc). En este tipo de sistemas se están utilizando dispositivos lógicos programables y la tendencia es diseñar arquitecturas reconfigurables.

Mientras tanto, las arquitecturas de los computadores personales se orientan a altas prestaciones en lo que se refiere a la ejecución de las instrucciones.

Se ha previsto que el espectacular aumento en las prestaciones de los procesadores diseñados en el ocaso del siglo XX no pueden ser mantenidos si se sigue diseñando de la misma forma. Es por esta razón por la cuál se investigan nuevas arquitecturas basadas en el paralelismo. Ejemplos de estas técnicas las tenemos en EPIC (Explicitly Parallel Instruction Computing) y en las arquitecturas que tratan de explotar el paralelismo a nivel de instrucciones.

Llegados a este punto nos encontramos con la problemática de la enseñanza de la arquitectura de computadores en las aulas universitarias.

¿A qué debemos dar más énfasis en las aulas, a la enseñanza de las arquitecturas que se han convertido en las tradicionales (arquitectura i386, arquitectura SPARC, etc) o a enseñar las nuevas arquitecturas (EPIC, Arquitecturas ILP, etc).

La solución salomónica sería presentar las dos. Sin embargo, dado que en nuestra Universidad se tienen dos titulaciones de Ingeniero Superior en Informática, la primera en Gestión (ISIG) y la segunda en Sistemas (ISIS), ¿qué se debería enseñar a cada uno de ellos y qué parte debería ser común?

Desde el punto de vista de los profesores que imparten la asignatura optativa "*Diseño de Procesadores*" (DPO) que se imparte en el tercer año de la titulación nos estamos planteando el presentar a los alumnos las nuevas arquitecturas que tratan de sacar provecho del paralelismo a nivel de instrucción.

La ponencia ha sido organizada de la siguiente forma:

- En la primera sección se presenta la organización actual de la asignatura DPO.
- En la segunda sección se presenta la nueva organización propuesta de DPO.
- En la tercera sección se presentan una serie de comentarios respecto a lo que esperamos.
- Finalmente, en el apéndice A se presenta un resumen de la arquitectura EPIC [1], mientras que en el apéndice B se presenta un resumen de las arquitecturas reconfigurables [2].

2.- DISEÑO DE PROCESADORES

Diseño de Procesadores es una asignatura optativa ofertada en el segundo semestre del tercer año de las titulaciones Ingeniero Superior en Informática de Sistemas (ISIS) e Ingeniero Superior en Informática de Gestión (ISIG) de la Facultad de Informática de la UPV.

Cuenta con seis créditos, tres de los cuáles se utilizan para lección magistral y los tres restantes se dedican a prácticas de laboratorio.

La metodología empleada se explica en la referencia [3], mientras que la relación de DPO con otras asignaturas relacionadas con la enseñanza de la arquitectura de computadores se puede consultar en la referencia [4] .

a) Objetivos

Los objetivos de esta asignatura se pueden agrupar en dos clases, los didácticos y los pedagógicos. Desde el punto de vista didáctico se pretende presentar al alumno el diseño de procesadores desde un punto de vista cualitativo.

En lo que respecta a los objetivos pedagógicos, se desea que el alumno busque, analice y presente información de acuerdo a sus intereses personales.

b) Temario

Los temas se agrupan en tres bloques.

Bloque 1: Ley de Amhdal e interpretación de cómo mejorar las prestaciones de los procesadores
- Introducción
- Tema 1: Fundamentos del diseño de computadores.
- Tema 2: Prestaciones y costes

Bloque 2: Estudio comparativo de CPU's comerciales
- Tema 3: Diseño de repertorios de instrucciones: Alternativas y principios
- Tema 4: Técnicas básicas de Implementación de procesadores

Bloque 3: Memoria y dispositivos de Entrada/Salida.
- Tema 5: Diseño de la Jerarquía de Memoria
- Tema 6: Diseño del subsistema de Entrada/Salida I
- Tema 7: Diseño del subsistema de Entrada/Salida II

c)Prácticas

Práctica I: Estudio de mercado. Comparación precio/prestaciones de sistemas comerciales.
Práctica II: Estudio de la frecuencia de utilización de las instrucciones.
Práctica III: Influencia del juego de instrucciones en el diseño de la UCP. Estudio comparativo de UCP's reales. (Algunas familias de procesadores propuestas como casos de estudio: Intel 80x86, Motorola 68K, MIPS Rx000,PA RISC (HP), AMD y otros clónicos, etc.).
Práctica IV: Evaluación de prestaciones en las memorias cache.

3. PROPUESTA DE LA ASIGNATURA DISEÑO DE PROCESADORES

a) Objetivos

Se pretende mantener los objetivos básicos de la asignatura descritos en el apartado anterior.

En cuanto a los objetivos pedagógicos nos siguen pareciendo perfectamente válidos y no creemos necesarias modificaciones en la metodología seguida, ya que los resultados obtenidos son aceptables.

Es en los objetivos didácticos donde vemos necesario introducir un factor nuevo que es la actualización de los contenidos transmitidos a los alumnos. Sin abandonar los conceptos básicos de la materia se quiere introducir a los alumnos las nuevas tendencias en la materia del diseño de procesadores.

Para ello se requieren algunas modificaciones en el temario impartido, así como en las prácticas realizadas en el laboratorio; aunque es complicado realizar modificaciones significativas, debido a los pocos créditos asignados a la asignatura.

b) Temario

En cuanto al temario se han realizado las siguientes modificaciones:

- En el Bloque 1 se introducen los conceptos básicos de las arquitecturas VLIW e ILP.
- En el Bloque 2 se elimina el Tema 4 y se sustituye por *Arquitecturas de los procesadores EPIC y VLIW.*
- Y, por último, en el Bloque 3 se reduce el tiempo dedicado a la Entrada/Salida y se introduce el tema de Hardware Reconfigurable.

El temario propuesto queda de la siguiente forma:

Bloque 1: Ley de Amhdal e interpretación de cómo mejorar las prestaciones de los procesadores

- Introducción
- Tema 1: Fundamentos del diseño de computadores. **Fundamentos de las arquitecturas VLIW e ILP.**
- Tema 2: Prestaciones y costes

Bloque 2: Estudio comparativo de Arquitecturas de CPU's.
- Tema 3: Diseño de repertorios de instrucciones: Alternativas y principios
- Tema 4: **Arquitecturas de los procesadores EPIC y VLIW.**

Bloque 3: Memoria y dispositivos de Entrada/Salida.
- Tema 5: Diseño de la Jerarquía de Memoria
- Tema 6: Diseño del subsistema de Entrada/Salida
- Tema 7: **Hardware Reconfigurable**

b) Prácticas

En cuanto a las prácticas impartidas en laboratorio se han reestructurado también con el fin de dar cabida a los nuevos conceptos introducidos en el temario.

A continuación se detalla la nueva estructura de las prácticas de laboratorio, remarcando las modificaciones realizadas.

Práctica I: Estudio de mercado. Comparación precio/prestaciones de sistemas comerciales.
Práctica II: **Estudio de la arquitectura de un procesador de 64 bits.**
Práctica III: Influencia del juego de instrucciones en el diseño de la UCP. Estudio comparativo de UCP´s reales. (Algunas familias de procesadores propuestas como casos de estudio: Intel 80x86, Motorola 68K, MIPS Rx000,PA RISC (HP), AMD y otros clónicos, etc.).
Práctica IV: Evaluación de prestaciones en las memorias cache.
Práctica V: **Herramientas para el diseño de Hardware Reconfigurable.**

4.- COMENTARIOS FINALES.

En el presente artículo, los profesores de la asignatura Diseño de Procesadores hemos querido presentar una propuesta de cambio en los contenidos de la misma motivados por los avances en la materia.

Queremos plantear esta propuesta ante la comunidad universitaria, con el objetivo principal de conocer otras opiniones y experiencias docentes relacionadas, generando una cierta retroalimentación.

APENDICE A. ARQUITECTURA EPIC

Podriamos definir EPIC (*Explicitly Parallel Instruction Computing*) como un nuevo estilo de arquitectura cuyo objetivo es obtener un elevado grado de paralelismo a nivel de instrucciones de los programas, consiguiendo a la vez no solamente no aumentar la complejidad del hardware sino reducirla.

En los últimos tiempos, el procesamiento paralelo a nivel de instrucciones (ILP) ha sido la única forma de aumentar el rendimiento de los procesadores de forma importante sin tener que realizar cambios importantes en el software. Sin embargo, esto ha provocado una gran complejidad en el hardware, complejidad que ha ido aumentando a medida que la industria ha querido obtener más rendimiento en los procesadores.

Los sistemas basados en Paralelismo a nivel de Instrucciones (ILP) parten de un programa en lenguaje de alto nivel escrito para procesadores secuenciales y utilizan la tecnología del compilador y el hardware para explotar el paralelismo del programa. Los dos tipos más importantes de procesadores ILP hasta la fecha son:

- Procesadores Superescalares, implementados para arquitecturas secuenciales, en los cuales el ILP debe ser explotado por el hardware. Esto implica una gran complejidad en el hardware.

- Procesadores VLIW (Very Long Instruction Word), para los cuales el programa proporciona información explícita para explotar el paralelismo. El compilador identifica el paralelismo en el programa y lo comunica al hardware indicándole qué operaciones son independientes entre sí y pueden ejecutarse en el mismo ciclo de reloj. De esta forma se simplifica el hardware. El inconveniente es que estas máquinas están orientadas a computación numérica y su rendimiento baja bastante con otro tipo de cargas.

El estilo de arquitectura EPIC es una evolución de VLIW que ha absorbido mucho del concepto superescalar. Dentro de la filosofía EPIC podríamos englobar muchos EPIC ISA (Instruction Set Architecture). En función de las características del sistema EPIC éste puede ser utilizado tanto para la implementación de procesadores de propósito general como para procesadores empotrados, siendo capaz de alcanzar un alto grado de paralelismo tanto en aplicaciones numéricas como escalares.

El principal objetivo de EPIC ha consistido en mantener la tarea de construir estáticamente un Plan de Ejecución (POE) realizada por el compilador, propia de la filosofía VLIW pero mejorarlo con características propias de los procesadores superescalares que le permitan un mejor manejo de determinados factores dinámicos.

En definitiva, está claro que el diseño en tiempo de compilación del POE es esencial a niveles altos de ILP, pero que existen ambigüedades que sólo pueden ser resueltos por mecanismos dinámicos. EPIC intenta conciliar ambas herramientas pero bajando estos mecanismos dinámicos al nivel de arquitectura, para que el compilador pueda usarlos selectivamente cuando sea necesario.

APENDICE B. DISPOSITIVOS LOGICOS PROGRAMABLES.

Los sistemas empotrados son altamente especializados. Los procesadores de propósito general no satisfacen las necesidades de aplicaciones tan dispares y el hardware a medida no resulta apropiado. Los sistemas reconfigurables combinan la flexibilidad de los procesadores de propósito general con la eficiencia del hardware a medida.
Podríamos definir un sistema reconfigurable como una serie de lógica configurable interconectada por red junto a elementos de almacenamiento, capaz de implementar una gran configuración lógica utilizando muy poco hardware y apoyándose en una rápida reconfiguración del hardware disponible.

Por ejemplo, podemos llegar a tener un sistema segmentado funcionando perfectamente incluso cuando el hardware completo de dicho sistema nunca esté presente, sino distintas configuraciones que lo componen.

Pero el éxito de un sistema reconfigurable depende también de disponer de compiladores apropiados que puedan crear configuraciones eficientes en un tiempo breve. Necesitaremos un

compilador parametrizado que pueda tratar con diferentes arquitecturas. Las información sobre las diferentes arquitecturas será pasada al compilador mediante un lenguaje intermediario.

Un sistema reconfigurable puede superar a un procesador de propósito general en:

- sistemas que operan con un ancho de bit distinto del tamaño de palabra básico del procesador,
- cálculos con dependencias de datos que permiten que múltiples unidades funcionales operen en paralelo
- operaciones básicas que pueden combinarse en una única operación especializada
- sistemas que pueden ser segmentados
- sistemas que reutilizan datos de entrada

BIBLIOGRAFÍA

[1] Michael S. Schlansker, B. Ramakrishna Rau. "EPIC: Explicitly Parallel Instruction Computing" IEEE Computer, Febrero 2000.

[2] Seth Copen, Herman Schmit, Mihai Budiu, Srihari Cadambi, Matt Moe, R. Reed Taylor. "PipeRench: A Reconfigurable Architecture and Compiler" IEEE Computer, Abril 2000.

[3] Lenin Lemus, Ana Pont. "Teaching Methodology at the EUI-FI". Workshop on computer architecture education (WCAE-98). 27 de Junio al 1° de Julio de 1998. Barcelona, España.

[4] Lenin Lemus, Ana Pont. "Computer Organisation and Architecture Teaching in the University School of Computer Science and in the Computer Science Faculty of the University Polytechnic of Valencia". Workshop on computer architecture education (WCAE-98). 27 de Junio 27 al 1° de Julio de 1998. Barcelona, España.

CACHESIM: UN SIMULADOR DE CACHE CONFIGURABLE

Mª Luisa Córdoba Cabeza[1], Mª Isabel García Clemente[1], Mª Luz Rubio Martínez[1]

[1]*Departamento de Arquitectura y Tecnología de Sistemas Informáticos (DATSI)*
Facultad de Informática (UPM)
e-mail: **mcordoba@fi.upm.es**

RESUMEN: Los conceptos de memoria virtual y jerarquía de memoria son importantes en la docencia de arquitectura de computadores. La mayoría de los libros de texto presentan estos temas como una descripción completa de conceptos básicos, modos de funcionamiento, y algoritmos. Pero es básico completar esto con ejemplos prácticos de diseño de caches reales, en la que puedan observar de forma empírica la influencia de las decisiones de diseño sobre la jerarquía de memoria en el rendimiento global del sistema.

1.- INTRODUCCIÓN.

Este trabajo presenta un simulador de jerarquía de memoria de un computador (CacheSim) desarrollado en el D.A.T.S.I. de la Facultad de Informática de la UPM. Este simulador está orientado a alumnos de Arquitectura de Computadores. Pero la herramienta se puede utilizar tanto para el desarrollo de prácticas en laboratorio, como para el aprendizaje por computador de la teoría de este tema, gracias a un completo sistema interactivo de ayuda.

El simulador se utilizaría para realizar prácticas en las que los alumnos diseñen su propia jerarquía de memoria. Dado un programa fuente (representativo de una determinada carga de trabajo típica), los alumnos deberían seleccionar de forma óptima los parámetros que caracterizan la jerarquía de memoria.
Los alumnos también pueden comprobar y cuantificar cómo influyen los distintos valores de algunos parámetros de la jerarquía en las prestaciones y rendimiento del sistema de memoria.

2.- OBJETIVOS.

El principal objetivo en el desarrollo de esta herramienta era construir no sólo una herramienta pedagógica, sino también innovadora.
Se ha desarrollado un simulador del funcionamiento de la jerarquía de memoria de un computador centrándose en la memoria cache y en la memoria principal. Calcula tasas de fallos, tasas de aciertos, tiempos de ciclo de memoria, tiempos medios de acceso a memoria para una configuración de jerarquía de memoria determinada.

Las características principales de la herramienta son:

1. Pedagógica
2. Accesible
3. Fácil de utilizar
4. Poco esfuerzo del alumno
5. Innovadora
6. Modular

El simulador se ejecuta en PC, bajo Windows. Esto permite facilidad de acceso a la aplicación. Es un programa Windows, por lo que el interfaz y entorno de ejecución es muy extendido y conocido por los alumnos.
La interfaz con el usuario se ha diseñado para facilitar tanto la ejecución de la simulación, como la introducción de los valores de los parámetros de configuración.
Otra característica es que CacheSim proporciona un sistema de ayuda sensible al contexto, en el que cabe resaltar:

- Incluye toda la teoría sobre las jerarquías de memoria. Los alumnos pueden simultanear su consulta en cualquier instante mientras realizan la práctica. Se pueden utilizar diversos tipos de búsqueda por índice, y acceder a un libro de texto *on line* rápido mientras se realiza la práctica. Esto permite al alumno que sea capaz de comenzar una práctica sin un conocimiento profundo previo sobre memorias cache, guiándole sobre los principales conceptos, el comportamiento, y posibles mejoras. El aprendizaje de la teoría es una tarea menos tediosa.

- Dirige al alumno durante la ejecución del simulador y le enseña a utilizarlo, de modo que es autoexplicativa.

- Detección de errores en la asignación de valores: parámetros de la configuración de la jerarquía de memoria, y variables de ejecución de la simulación. Por ejemplo, la introducción de parámetros que tienen opciones limitadas, es a través de gráficos en forma de listas en cascada.

Todo esto facilita la introducción de los datos de entrada, la ejecución de la simulación, con un esfuerzo mínimo de comprensión del interfaz y la operatividad de la herramienta. Todas estas características se resumen en:

- El alumno se puede concentre su esfuerzo en el estudio del diseño y optimización de la jerarquía de memoria

- Utilización sencilla y cómoda de la herramienta. Se facilita la introducción de variables y la repetición de experimentos.

- La obtención de los datos de entrada y la visualización de los resultados sean tareas sencillas. Se enlaza con WgnuPlot, para visualizar gráficamente los resultados de rendimiento obtenidos en distintas simulaciones.

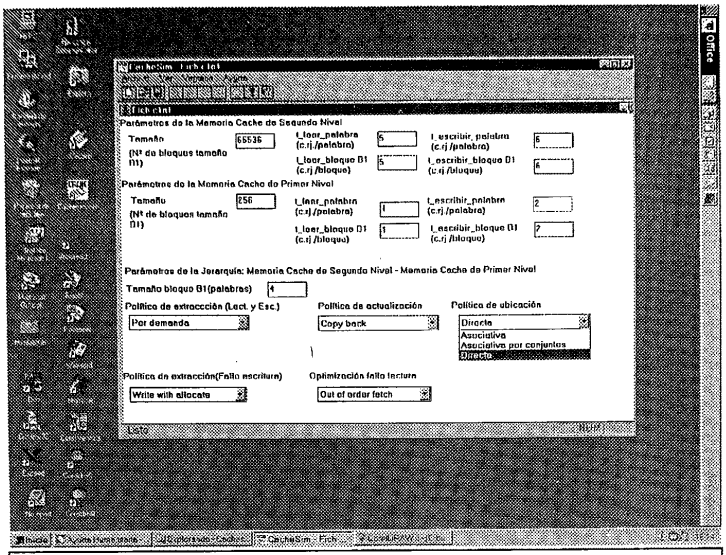


Figura 1.- Selección de parámetros de entrada

CacheSim es además innovador respecto a otras herramientas similares (DineroIII, SISMEC), ya que incluye :

- Calcula tiempos medios de acceso y tiempos de ciclo de memoria para tiempos de acceso tanto de palabra como de bloques, seleccionados por el usuario, en cualquier nivel de la jerarquía.
- Permite elegir tiempos de acceso para palabras y para bloques. Esto hace que sea posible mejorar el ancho de banda entre dos niveles adyacentes de la jerarquía de memoria. Al tener que proporcionar tiempos para palabras y para bloques, se puede aumentar el tamaño del bloque que se intercambia entre dos niveles de la jerarquía sin el correspondiente incremento en la penalización de fallos. El número de arquitecturas que es posible simular es mayor que con otras herramientas.
- Implementa la política de reemplazo generada por la aplicación del algoritmo LRU*. Este algoritmo se utiliza en las memorias cache multinivel cuando el segundo nivel de memoria cache tiene una política de ubicación asociativa o asociativa por conjuntos.

3.- EL SIMULADOR CACHESIM

CacheSim es un simulador de jerarquía de memoria cuyo sistema de memoria está compuesto por uno o dos niveles de memoria cache y un nivel de memoria principal. Es un simulador dirigido por traza. La entrada de datos es un fichero de traza con las referencias a memoria de una ejecución de un programa. Antes de la simulación hay que generar esta traza. Ésta se puede generar manualmente, o con otros programas de libre difusión como DLXsim.

CacheSim simula tres configuraciones posibles:

- Memoria cache única
- Memoria cache de datos separada de la memoria cache de instrucciones
- Dos niveles posibles de cache

Antes de la simulación también hay que configurar la jerarquía de memoria: tamaños y tiempos de acceso de caches, tamaño y tiempo de acceso de memoria principal, relación entre cache y memoria principal (tamaño de bloque), y todas las políticas de intercambio de bloques.
Durante la simulación se supone la propiedad de inclusión multinivel: todos los datos contenidos en la memoria de un nivel están incluidos también en la del nivel siguiente. Por lo tanto, en un fallo de cache de segundo nivel hay que invalidar todos los bloques del primer nivel correspondientes. Esto incrementa ligeramente la frecuencia de fallos del primer nivel.

La configuración de la memoria y las políticas se definen mediante ficheros. Diferentes tipos de ficheros contienen los parámetros de entrada. Se pueden crear ficheros de entrada totalmente nuevos, o modificando alguno existente.

Los resultados de la simulación (tasas de aciertos, tasas de fallos, tiempos medios de accesos, tiempos medios de ocupación) también se obtienen en ficheros.

Como salida opcional también se puede obtener un fichero de volcado con traza de la simulación con los contenidos de memoria y resultados tras cada acceso a memoria: cuál es el número de bloque de memoria principal de la dirección de memoria procesada, dónde están los bloques de memoria principal, los valores de los bits de validez y ensuciado, contenido de la lista del algoritmo de reemplazo, etc... Los alumnos pueden estudiar este fichero de volcado para diferenciar los distintos tipos de fallos de cache:

- fallos forzosos
- fallos por capacidad
- fallos por conflicto

La herramienta enlaza con WgnuPlot para directamente, si se desea, representar gráficamente los resultados de una o varias simulaciones.

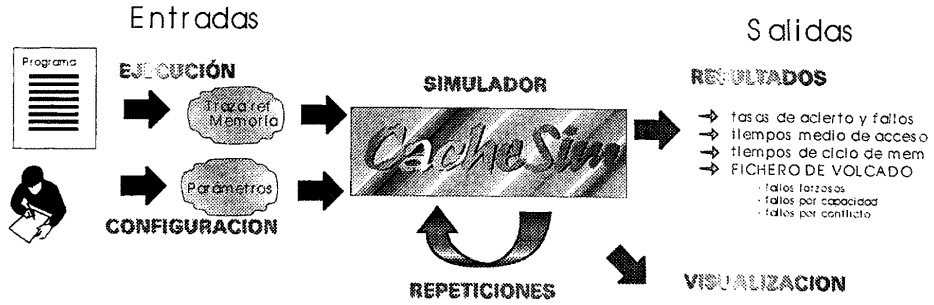


Figura 2.- Esquema de utilización de CacheSim

4.- EJEMPLOS

A modo de ejemplo de una práctica, mostramos los resultados que los alumnos obtendrían tras simular repetidamente una traza, variando algunos parámetros de configuración mientras otros permanecen constantes. En estos ejemplos nos hemos centrado en estudiar las variaciones de las tasas de aciertos, aunque también observamos otros resultados indicadores de prestaciones (tiempo de acceso, tiempo de ocupación o de ciclo, tasa de fallos, etc). Los resultados se han representado con WGnuPlot.

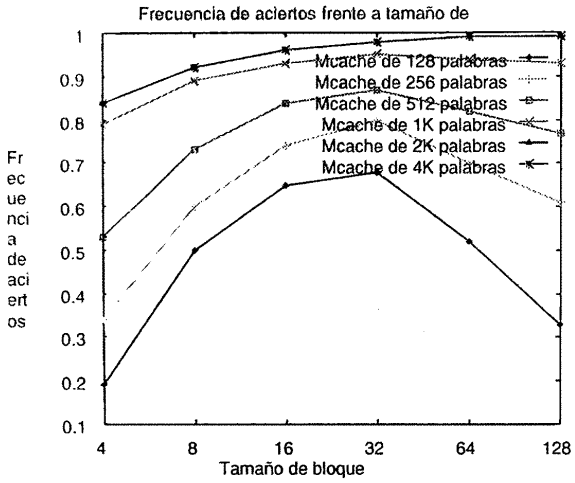


Figura 3.- Tasa de aciertos para cache única y distintos tamaños de cache y bloque

Cache única. *Configuración*: Memoria principal: tamaño 1M pal; tiempo de acceso a palabra 10 c; tiempo de acceso a bloque 10 c; política de ubicación directa.
Fetch por demanda; política de escritura *copy back*; política de escritura en fallo *write with allocate*; política de lectura *out of order fetch*
Memoria cache: tiempo de lectura de palabra 1c; tiempo de escritura de palabra 2c; tiempo de lectura de bloque 1c; tiempo de escritura de bloque 2c.

5.- CONCLUSIONES

En este artículo se presenta un simulador de jerarquía de memoria para propósitos didácticos: CacheSim. Es una herramienta pedagógica, fácil de utilizar, y con un sistema de ayuda sensible al contexto que además de guiar al alumno a través de la simulación, le explica todos los conceptos teóricos de jerarquía de memoria. Su utilización sencilla permite al alumno centrar sus esfuerzos en los conceptos de jerarquía de memoria en vez de aprender el manejo de la herramienta. Este simulador enseña al alumno además cómo diseñar una jerarquía de memoria, y le permite observar y cuantificar las consecuencias en las prestaciones y rendimiento del sistema debidas a sus elecciones en el diseño.

En cuanto a líneas de continuación del trabajo, tenemos planeado modificar los formatos de los datos de entrada . Además de un simulador dirigido por traza, estamos tratando de conectar el simulador con un generador de trazas de libre difusión para que la entrada al simulador pueda ser un programa en código fuente. Los alumnos podrían estudiar cómo programar un algoritmo de forma que se obtenga el mejor rendimiento de la configuración de la jerarquía de memoria. Esto les permitiría identificar en qué instrucciones del programa se producen los fallos. Sería posible entonces, trabajar en técnicas de optimización de compiladores tales como el intercambio de bucle y la fusión de bucle o bloqueo.

6.- BIBLIOGRAFÍA

[1]. University of California.
Dlxcc-DLX Compiler (Dlx Specific Implementation of gcc)
University of California, 1988.

[2]. García, M. Isabel.
Sistema de Memoria.
Publicaciones, Univ. Politécnica de Madrid, 1998.

[3]. Córdoba, M.L.; García, M.I.; Rubio, M.L.
CacheSim: A Cache Simulator for Teaching Memory Hierarchy Behaviour.
Proc. Of 4th Annual SIGCSE/SIGCUE (ACM). ITiCSE'99. Kracow, Poland, 1999.

[4]. Gómez, J.A.; Sánchez, J.M.; Moreno, J.A.
An Educational Tool for testing Hierarchical Multilevel Caches.
Dept. Informática de la Universidad de Extremadura, 1997. Hostetler, L.B.; Mirtich, B.
DLXSim - A Simulator for DLX.
November, 1993. Wang, Wen-Hann.
Multilevel Cache Hierarchies.
Tech. Report 89-09-13. Dept. of Comp. Science and Engineering. FR-35. Univ. of Washington, 1989.

[5]. Kotz,D.
Latex and the Gnuplot Plotting Program (ver. 3.0).
david.kotz@dartmouth.edu July, 1991.

[6].David A. Patterson, John L. Hennessy.
Computer Organization and design. The hardware/software interface. Second Edition.
Morgan Kaufmann Publishers, Los Altos, USA, 1998.

INNOVACIÓN DOCENTE EN LA ASIGNATURA DE ARQUITECTURA DE COMPUTADORES MEDIANTE EL USO DE SIMULADORES

Miguel Ángel Vega[1], Juan Manuel Sánchez[1], Juan Antonio Gómez[1]

[1]*Departamento de Informática. Universidad de Extremadura*
Escuela Politécnica. Campus Universitario s/n. 10071 Cáceres
e-mail: mavega@unex.es, sanperez@unex.es, jangomez@unex.es

RESUMEN: En la Universidad de Extremadura, durante el cuarto curso de la titulación de Ingeniero en Informática se imparte la asignatura de Arquitectura de Computadores. Los profesores de esta asignatura han apostado por la incorporación de nuevas tecnologías para ofrecer una mayor calidad en la docencia. En este sentido cabe destacar el desarrollo de simuladores específicos para prácticas dentro del campo de los multiprocesadores. Presentamos dos de estos simuladores. El simulador RedME para la simulación de redes multietapa, y el simulador SMPCache para la simulación de sistemas de memoria caché en multiprocesadores simétricos.

1.- INTRODUCCIÓN.

Una de las titulaciones existentes en la Universidad de Extremadura es la de Ingeniero en Informática. Durante el cuarto curso de esta titulación se imparte la asignatura de Arquitectura de Computadores, siendo ésta una asignatura troncal anual con un fuerte contenido tecnológico. Debido al constante avance en esta área, es imprescindible que, sin perder de vista los fundamentos y bases que permanecen en el tiempo, se oferten unos contenidos teóricos y prácticos que reflejen el estado actual, sin detrimento de una buena pedagogía. La apuesta realizada por los profesores de la asignatura se centra en la incorporación de nuevas tecnologías para ofrecer una mayor calidad en la enseñanza. En este sentido cabe destacar el desarrollo de simuladores específicos para prácticas avanzadas dentro del campo de los multiprocesadores. Los multiprocesadores se están convirtiendo en una forma de paralelismo muy útil y extendida para maximizar el rendimiento, fiabilidad, flexibilidad y disponibilidad de un sistema informático, siendo cada día más importantes.

En los multiprocesadores la interconexión entre los elementos puede realizarse mediante un conmutador de barras cruzadas, una red multietapa o un bus común. Las redes multietapa son mucho más baratas y sencillas que los conmutadores de barras cruzadas para sistemas multiprocesador de gran escala. Más aún, las redes multietapa son modulares y fáciles de controlar, permitiendo su expansión incremental y su reparabilidad. Estas características hacen a las redes multietapa imprescindibles para multiprocesadores de gran escala. El simulador RedME permite al alumno realizar un estudio detallado sobre distintas redes multietapa.

Las cachés son un componente crítico en el rendimiento de cualquier sistema. La forma más frecuente de arquitectura paralela son los multiprocesadores de pequeña a moderada escala denominados multiprocesadores simétricos (SMPs) [1]. En la actualidad, entre las categorías de las jerarquías de memoria en multiprocesadores la más extendida es la de memoria compartida por bus común. El simulador SMPCache permite al alumno simular mediante trazas sistemas de memoria caché en multiprocesadores simétricos con memoria compartida por bus.

En la próxima sección mencionamos las diferentes características que ofrece el simulador RedME. En la sección 3 se explica el simulador SMPCache. Finalmente, en la última sección se presentan una serie de conclusiones obtenidas del desarrollo y uso de los simuladores. Las consideraciones teóricas sobre redes multietapa y sistemas de memoria caché dentro de multiprocesadores aparecen perfectamente desarrolladas en muchos textos de arquitectura de computadores [1]-[9], por lo que no las mencionaremos aquí. Todas las operaciones y algoritmos utilizados responden fielmente a lo expuesto en estos textos. Por tanto, los resultados obtenidos con los simuladores tienen una traducción casi fidedigna en el mundo real.

2.- EL SIMULADOR REDME.

RedME 1.0 es una herramienta software para la evaluación de redes multietapa en multiprocesadores, que opera sobre una plataforma PC con sistema operativo MS-DOS o Windows. RedME (Red MultiEtapa) ofrece una interfaz gráfica típica de Windows, mediante un entorno agradable y fácil de manejar, basado en iconos.

El simulador permite estudiar redes multietapa bilaterales (*two-sided*). Distinguimos tres tipos de redes bilaterales: con bloqueo, reorganizables (*rearrangeable*) y sin bloqueo. Los tres tipos tienen cabida dentro del simulador. En particular, RedME permite estudiar los siguientes tipos de redes: red Omega, red de Benes, red de Clos, red Delta, red Baseline (o red de línea base), red R, red n-cubo binario, red Banyan (o red Banyano) y red Flip. Para todas ellas el simulador pide el número de entradas/salidas y muestra en pantalla el aspecto que la red en cuestión tendría.

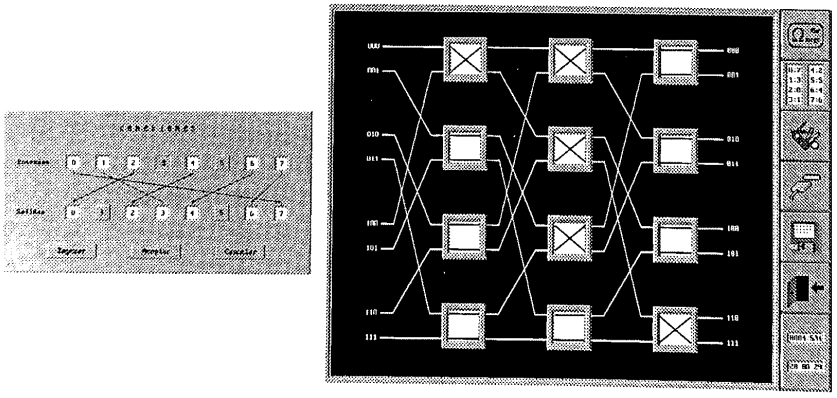


Figura 1. (a) Introducción de la permutación π_1=(0,7,6,4,2)(1,3)(5). (b) Permutación π_1 implementada en una red Omega sin bloqueo.

Además, RedME calcula el encaminamiento de los datos dentro de la red. Por ejemplo, en la figura 1(b) se muestra una red Omega con 8 entradas/salidas, junto con el encaminamiento necesario para realizar la permutación π_1. En la figura 1(a) se observa cómo el usuario ha introducido las conexiones entre las entradas y salidas. En este caso, la permutación solicitada es π_1=(0,7,6,4,2)(1,3)(5), por tanto, las conexiones deseadas son 0→7, 7→6, 6→4, 4→2, 2→0, 1→3, 3→1 y 5→5. Observamos que no existe bloqueo en esta red para implementar la permutación π_1. Nótese que en la parte superior derecha de pantalla aparecen las conexiones entre entradas-salidas que se están utilizando en cada instante.

Actualmente, el simulador es capaz de calcular el encaminamiento para las redes: Omega, Benes, Clos, Baseline y R. Si se trata de una red con bloqueo, también es posible utilizar el simulador para calcular la solución a los bloqueos existentes. Por ejemplo, consideremos ahora la permutación π_2=(0,6,4,7,3)(1,5)(2) para una red Omega. Observamos en la figura 2(a) que existe bloqueo entre algunas de las conexiones necesarias para implementar la permutación π_2. Los conflictos aparecen en tres de los conmutadores (A, B y C). Para resolver los conflictos las conexiones deben realizarse (agruparse) en varios pasos. De manera que de entre las conexiones en conflicto, sólo una se realizará en un paso determinado. La figura 2(b) muestra la solución a los bloqueos dada por el simulador.

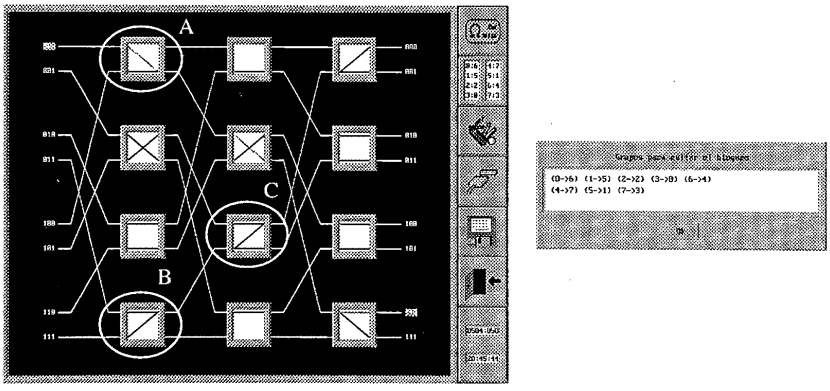


Figura 2. (a) Permutación π_2=(0,6,4,7,3)(1,5)(2) implementada en una red Omega con bloqueo. (b) Solución a los bloqueos planteados por la permutación π_2.

Realizado el encaminamiento, el simulador también permite seleccionar una entrada cualquiera, resaltando la ruta a seguir dentro de la red para llegar desde dicha entrada a la salida asociada. En la figura 2(a) se resalta la ruta a seguir para llegar desde la entrada 0 (000) a la salida 6 (110).

RedME permite guardar en disco la configuración actual del sistema. Almacenando la red bajo estudio y su número de entradas/salidas, además de las conexiones actuales entre las entradas y las salidas. Posteriormente, será posible cargar desde un fichero en disco estos datos, evitando volver a indicar estos parámetros en posteriores sesiones.

3.- EL SIMULADOR SMPCACHE.

SMPCache 1.0 es una herramienta software para la evaluación de sistemas jerárquicos de memorias cachés en SMPs con memoria compartida por bus, que opera sobre una plataforma PC con sistema operativo Windows 98 (ó 95), y ha sido escrito utilizando un lenguaje visual. SMPCache ofrece una interfaz gráfica típica de Windows (figura 3(a)), disponiendo además de una ayuda contextual muy completa.

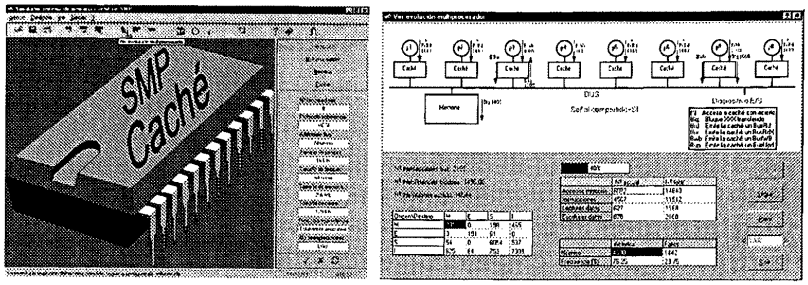


Figura 3. (a) Interfaz gráfica. (b) Visión del multiprocesador en una simulación.

El simulador permite establecer la configuración de los parámetros que definen la arquitectura del sistema (un resumen de los mismos aparece en la tabla 1), ofreciéndonos la respuesta del mismo al someterlo a los accesos de memoria generados por los programas (trazas de memoria que se hayan cargado en los distintos procesadores). Por tanto, se trata de una aplicación que podría encargarse de la evaluación de sistemas de memoria en multiprocesadores con fines de investigación.

Dada su facilidad de manejo, el simulador también puede ser utilizado igualmente con fines didácticos, ya que permite observar de una forma clara y gráfica cómo va actualizándose el sistema completo a medida que se avanza en la ejecución de los programas (las trazas son leídas). Permitiendo una visión completa de la evolución del multiprocesador (ver figura 3(b)), de una caché en particular o incluso de un determinado bloque de memoria (figura 4(a)). Mostrando, en cada instante, los accesos a memoria realizados por cada procesador, el estado del bus, de cada caché, de cada bloque dentro de cada caché,...

El simulador también permite estudiar el sistema de memoria que mejor se ajusta a nuestras necesidades antes de su implementación física, o simplemente nos ayuda a simular sistemas actuales para ver su eficacia y poder comparar resultados de una forma fácil. Algunos de los factores que se pueden estudiar con el simulador son: Localidad de los distintos programas; influencia del número de procesadores, del protocolo de coherencia caché, del algoritmo de arbitración del bus, del tipo de correspondencia, de la política de reemplazo, del tamaño de la caché (bloques en caché), del número de conjuntos de caché (en correspondencia asociativa por conjuntos), del número de palabras por bloque (tamaño del bloque), del ancho de la palabra,...

Procesadores en el SMP	1, 2, 3, 4, 5, 6, 7 u 8
Protocolos de coherencia caché	MSI, MESI o DRAGON
Algoritmos de arbitración del bus	Aleatorio, LRU o LFU
Anchos de palabra (en bits)	8, 16, 32 ó 64
Palabras en un bloque	1, 2, 4, 8, 16, 32, 64, 128, 256, 512 ó 1024
Bloques en memoria principal	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144, 524288, 1048576, 2097152 ó 4194304
Bloques en caché	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 ó 2048
Funciones de correspondencia	Directa, asociativa por conjuntos o totalmente asociativa
Conjuntos de caché (para asociativa por conjuntos)	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 ó 2048
Algoritmos de reemplazo	Aleatorio, LRU, FIFO o LFU
Políticas de escritura	Post-escritura (por protocolos de coherencia utilizados)
Niveles de caché	1
Referencias	A palabras de memoria
Tamaño de bloque máximo	8 Kb
Tamaño de memoria principal máximo	32 Gb
Tamaño de caché máximo (sin etiquetas, bits de estado, campos de cómputo,...)	16 Mb

Tabla 1. Resumen de las características arquitectónicas soportadas por el simulador.

Además, SMPCache presenta mediante datos estadísticos y todo tipo de gráficas (la figura 4(b) muestra un ejemplo), medidas de interés como:

- Número de transacciones en el bus.
- Número de transferencias de bloques a través del bus.
- Tráfico en el bus teniendo en cuenta las dos medidas anteriores.
- Número de transiciones de estado (cada bloque en caché tiene un estado asociado).
- Número de transiciones de estado según el estado origen y el destino.
- Accesos a memoria realizados, desglosándolos por tipos: captura de instrucción, lectura de dato o escritura de dato.
- Nº de aciertos y fallos de caché, así como frecuencia de aciertos y fallos.

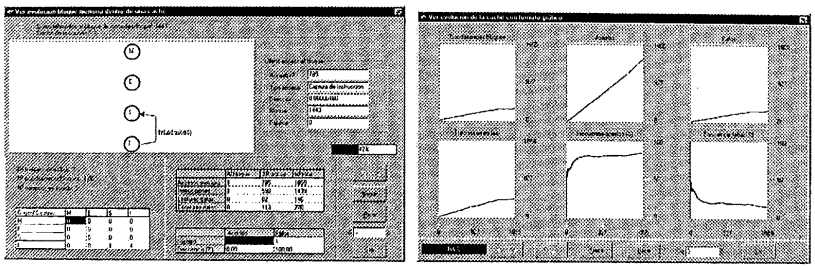


Figura 4. (a) Diagrama de transición de estados de un bloque de caché concreto durante una simulación. (b) Datos en formato gráfico de una caché.

Todos estos datos se presentan a distintos niveles, aunque siempre teniendo en cuenta la interrelación de todos los elementos del sistema. Podemos, por tanto, realizar una simulación observando el multiprocesador completo, y todos los bloques de memoria o un único bloque. También podemos observar una caché específica, y todos los bloques de memoria o un bloque concreto.

La simulación realiza una aproximación programada de las operaciones que realizarían los componentes del sistema de memoria caché en un multiprocesador real. Para ello, se realizan los cálculos adecuados y, al mismo tiempo, se van ofreciendo los resultados, tanto actuales como acumulados. Existen tres tipos de simulación (paso a paso, con punto de interrupción y ejecución completa), admitiéndose el cambio de un tipo de simulación a otro en mitad de la misma. También existe la posibilidad de abortar la simulación en cualquier instante, con el objeto de corregir algún detalle de la organización.

Las distintas selecciones que se hacen en el simulador para configurar una organización determinada pueden guardarse en ficheros ASCII de datos para una futura carga, evitando así la realización de numerosas selecciones para configurar la misma arquitectura. Más aún, es posible establecer una configuración inicial por defecto para el simulador. Estas características permiten construir una base de datos con las diferentes arquitecturas de memoria, pudiendo emular arquitecturas como Silicon Graphics, Sequent Symmetry, PowerPC, Sun, y otras.

4.- CONCLUSIONES Y LÍNEAS FUTURAS.

En este trabajo hemos mostrado las principales características de dos simuladores: RedME y SMPCache. La experiencia más inmediata nos ha dado muestra de los beneficios de éstos con fines docentes. Los alumnos los han utilizado para comprobar, experimentalmente y de forma gráfica, los distintos aspectos teóricos sobre multiprocesadores tratados en la asignatura de Arquitectura de Computadores, con lo cual los conceptos se afianzan de una forma más segura.

El simulador RedME se encuentra aún en periodo de construcción, puesto que se tiene planificado añadirle nuevas características, como la posibilidad de estudiar el encaminamiento para las redes multietapa: Delta, n-cubo binario, Banyan y Flip; también se tiene intención de añadirle nuevas redes, como la red Butterfly,... De igual forma, estamos trabajando en la incorporación de nuevas características al simulador SMPCache, tales como tiempo medio de accesos, penalización de fallos,...

5.- REFERENCIAS.

[1] D. E. Culler, J. P. Singh y A. Gupta, "Parallel Computer Architecture. A Hardware/Software Approach", Morgan Kauffmann, 1999.

[2] H. J. Siegel. "Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies", 2ª edición, McGraw-Hill, New York, 1989.

[3] R. W. Hockney y C. R. Jesshope. "Parallel Computer 2: Architecture, Programming and Algorithms", Institute of Physics Publishing, Bristol and Philadelphia, 1992.

[4] K. Hwang. "Advanced Computer Architecture: Parallelism, Scalability, Programmability", McGraw-Hill, Inc., 1993.

[5] K. Hwang y F. A. Briggs. "Computer Architecture and Parallel Processing", McGraw-Hill, Inc., 1984.
[6] J. Archibald y J.-L. Baer, "Cache Coherence Protocols: Evaluation Using a Multiprocessor Simulation Model", ACM Transactions on Computer Systems, vol. 4, nº 4, pg. 273-298, Noviembre 1986.
[7] S. Eggers, "Simulation Analysis of Data Sharing in Shared Memory Multiprocessors", Ph. D. Thesis, Univ. de California, Berkeley, Computer Science Division, Technical Report UCB/CSD 89/501, Abril 1989.
[8] D. Sima, T. Fountain y P. Kacsuk, "Advanced Computer Architectures. A Design Space Approach", Addison-Wesley, 1998.
[9] J. L. Hennesy y D. A. Patterson, "Computer Architecture. A Quantitative Approach", 2ª edición, Morgan Kauffmann, 1996.

LAS ASIGNATURAS DE ARQUITECTURA DE COMPUTADORES EN LA TITULACIÓN DE INGENIEROS DE TELECOMUNICACIÓN.

Miguel A. Mateo Pla[1], Julio Pons Terol[1]

[1]*Departamento de Informática de Sistemas y Computadores (DISCA)*
e-mail: {mimateo,jpons}@disca.upv.es

RESUMEN: La implantación de los planes de estudio de 1996 en la titulación de Ingenieros de Telecomunicación alcanzó en el curso 1999/2000 el 7° y 8° cuatrimestre (4° curso). Como consecuencia se han impartido por primera vez las asignaturas de Arquitectura de Computadores y Sistemas Operativos I y II (ACSO1 y ACSO2).

En este artículo se presenta el trabajo realizado durante este primer año de docencia. Partimos de los objetivos y el plan de docencia planteado en septiembre de 1999, desarrollándose a continuación la historia del curso, con los problemas y satisfacciones observados por los profesores.

1.- INTRODUCCIÓN.

En el curso 1999-2000 se han impartido, por primera vez, los cuatrimestres séptimo y octavo, correspondientes a cuarto curso, del plan de estudios de 1996 de la titulación de Ingenieros de Telecomunicación en la Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT) de la Universidad Politécnica de Valencia (UPV).

En cuarto curso, pero en el plan de estudios de 1990, se impartía la asignatura de Arquitectura de Computadores y Sistemas Operativos, una asignatura de 10 créditos, distribuidos en 7,5 de teoría y 2,5 de prácticas, impartidos en 25 semanas de clase.

En el plan del 96 no existen asignaturas anuales por lo que en principio la asignatura se divide en dos ACSO1 y ACSO2. Los profesores de la anterior asignatura son los encargados de poner en marcha estas nuevas asignaturas, barajándose dos posibilidades: dividir el temario de la asignatura anterior o aprovechar el cambio de plan para modificar tanto el temario como las prácticas. Con la experiencia obtenida en los años de docencia en ACSO, los profesores optan por esta segunda opción.

Tomando como punto de partida del presente artículo el momento en que se toma esta decisión, se describe el entorno de la asignatura, los objetivos que se persiguen en las nuevas asignaturas, el desarrollo de los mismos y las conclusiones obtenidas tras la finalización del curso.

2.- DATOS DE LAS ASIGNATURAS

Las asignaturas ACSO1 y ACSO2 tienen una carga de 4,5 créditos cada una de ellas. Estos créditos se reparten en 2 de teoría, 1 de problemas de aula y 1,5 créditos de prácticas.

Los semestres docentes se suponen de 15 semanas en la UPV, lo que significa que un alumno recibe cada semana 2 horas de docencia en aula y 1 hora de laboratorio.
Según esa distribución los alumnos dispondrían de 15 sesiones de prácticas de 1 hora, pero esta duración parece poco apropiada para una sesión de prácticas. Por este motivo se acuerda distribuir las horas de prácticas en 7 sesiones de 2 horas, equivalente para los alumnos a una sesión cada 15 días. Se reserva una sesión adicional para recuperaciones y evaluaciones.

Los alumnos que se matriculan de las asignaturas han podido cursar las siguientes asignaturas:

- Programación. Troncal de semestre 1. Asignatura que se centra en los fundamentos de la programación en Pascal y C.
- Fundamentos de computadores. Troncal de semestre 2. Introducción al HW de un computador.
- Introducción a los ordenadores personales. Optativa de semestre 2. Conceptos medios y avanzados de la programación a bajo nivel de computadores personales con MS-DOS.
- Sistemas electrónicos digitales. Troncal de semestre 5. Estudio de un sistema microprocesador con especial énfasis en la entrada salida. Utilización del ensamblador de Motorola 68000 en la programación de dispositivos.
- Programación Avanzada. Optativa de semestre 6. Programación de aplicaciones en Internet basándose en lenguaje Java.

Aunque los alumnos conocen parte de las unidades funcionales del computador, desconocen en gran medida como funciona la CPU y posibles formas de mejorar las prestaciones de un computador.

Así mismo, los alumnos tienen un desconocimiento casi completo sobre Sistemas Operativos, y las implicaciones de la arquitectura del computador y la estructura del procesador en el diseño de los mismos.

3.- OBJETIVOS DE LAS ASIGNATURAS.

Los objetivos generales que se desean cubrir entre las dos asignaturas se pueden resumir en los siguientes puntos:

- Completar los conocimientos sobre la estructura y funcionamiento de los computadores adquiridos en asignaturas de cursos anteriores.
- Definir el concepto moderno de arquitectura y distinguir los parámetros que influyen sobre las prestaciones de la misma.
- Enumerar los aspectos fundamentales del juego de instrucciones de un computador.
- Comprender las técnicas de segmentación y aplicarlas a la unidad de instrucción del computador. Comprender y aplicar las técnicas de gestión dinámica de instrucciones para alcanzar altas prestaciones.
- Conocer y comprender la arquitectura de los computadores superescalares y de los supercomputadores actuales.
- Conocer y comprender la visión de usuario y la de programador de los sistemas operativos y su integración con la arquitectura de computadores.

- Distinguir los niveles de utilización de un sistema operativo.
- Describir el funcionamiento de componentes fundamentales de un sistema operativo.
- Conocer, comprender y utilizar las llamadas al sistema de UNIX.
- Conocer, comprender y saber resolver el problema de la sincronización de procesos concurrentes.
- Conocer y evaluar algoritmos de planificación de procesos, tanto de propósito general como de tiempo real.
- Conocer y comprender las funciones de la gestión de memoria.
- Conocer y comprender la organización y la programación de los sistemas de entrada/salida, centrándose en los sistemas de ficheros.
- Enumerar los requisitos de una aplicación en tiempo real.
- Conocer el concepto de redes de estaciones de trabajo, siendo capaces de utilizarlas para la resolución de problemas sencillos.

La distribución de estos objetivos entre las dos asignaturas se realiza de forma que ACSO1 reúna todos aquellos relacionados con la estructura del computador y los temas generales de introducción. ACSO2 se centra en los objetivos relacionados con los sistemas operativos y los conceptos avanzados con mayor relación con ellos (sistemas de tiempo real y redes de estaciones de trabajo). Esta división permite realizar una planificación temporal en la que al alumno se le presentan los conceptos relacionados entre sí en una secuencial temporal lógica.

Los objetivos no se pueden alcanzar sin la realización de prácticas adecuadas. Este hecho hace que las prácticas adquieran una especial significación. Las prácticas preparadas para las asignaturas las podemos clasificar en dos tipos, según se orienten a:

- Afianzar conceptos. El alumno compara resultados obtenidos en prácticas con los conceptos introducidos en las clases teóricas.
- Adquirir habilidades. El alumno consigue nuevas habilidades que le preparan en su desarrollo profesional.

Más de la mitad de las prácticas impartidas en el curso han sido preparadas especialmente para estas asignaturas. De la desaparecida ACSO se recuperan aquellas prácticas que se actualizaron en los dos últimos años de docencia y que han tenido unos resultados claramente satisfactorios.

4.- ACSO1.

En esta asignatura se da un mayor peso a la parte correspondiente a la arquitectura del computador. Se trata de conectar los conceptos de la arquitectura de computadores con los de sistemas operativos, aunque estos últimos ocupan algo de un 10% del tiempo de teoría.

Esta distribución cambia en prácticas, en las que se utiliza en torno al 50% del tiempo para cada uno de estos apartados.

a) Temario.

La organización del temario de ACSO1 y su duración prevista es:

U. Temática 1: Conceptos de Arquitectura de computadores
- T1. Unidades funcionales del computador 4h
- T2. Introducción a los sistemas operativos 2h
- T3. Concepto de Arquitectura 4h
- T4. Diseño del juego de instrucciones 2h

U. Temática 2: Computadores segmentados
- T5. Unidades segmentadas 4h
- T6. Unidades de instrucción segmentadas 5h

U. Temática 3: Arquitecturas Avanzadas
- T7. Computadores superescalares 5h
- T8. Multiprocesadores 4h

En la duración de los temas se incluye tanto la teoría como los problemas de aula, por eso el total de horas estimado es 30 horas. Otro dato importante es que las tres unidades temáticas tienen un peso y duración similar, lo que se corresponde con los objetivos planteados.

b) Practicas.

Las practicas planteadas para la asignatura son:

- Práctica 1. Introducción al sistema operativo LINUX. (2 ses)
- Práctica 2. Programación C en LINUX. (2 ses)
- Práctica 3. Unidad de instrucción segmentada I. (1 ses)
- Práctica 4. Unidad de instrucción segmentada II. (2 ses)

Además de estas siete sesiones (14 horas de laboratorio), se reserva una hora por grupo para la recuperación de prácticas y presentación de resultados.
La primera práctica no necesita una introducción previa de teoría y sirve para que el alumno conozca el entorno del laboratorio donde realizará las prácticas relacionadas con sistemas operativos. Para estas prácticas los alumnos dispondrán durante todo el curso de una cuenta en un servidor Linux del laboratorio docente, cuentas que utilizarán en ACSO2.
Las dos últimas prácticas se desarrollan entorno a un simulador de máquina segmentada desarrollado en la unidad docente de Arquitectura de Computadores del DISCA. Este simulador permite observar las técnicas de segmentación explicados en teoría.

5.- ACSO2.

La asignatura ACSO2 se centra en estudiar los objetivos relacionados con los Sistemas Operativos. Para esta asignatura se toma como base para teoría el libro de Andrew S. Tanenbaum y Albert S. Woodhull titulado "*Sistemas Operativos. Diseño e implementación*". El contenido de este libro se complementa en algunos temas específicos, como el caso de los sistemas de tiempo real o las redes de estaciones de trabajo.

a) Temario.

El temario de ACSO2 se organiza en cinco unidades temáticas divididas en un total de 10 temas. Los temas y su duración prevista son:

U. Temática 1: Visión del programador de un Sistema Operativo	
T1. Estructura del Sistema Operativo.	2 h
T2. Llamadas al sistema.	4 h
U. Temática 2: Gestión de procesos	
T3. Programación concurrente.	4 h
T4. Planificación de procesos.	2 h
U. Temática 3: Gestión de memoria	
T5. Multiprogramación	3 h
T6. Memoria Virtual	2 h
U. Temática 4: Gestión de ficheros	
T7. Manejadores de dispositivo	3 h
T8. Sistema de ficheros	4 h
U. Temática 5: Conceptos Avanzados	
T9. Núcleos en tiempo real.	3 h
T10. Redes de estaciones de trabajo.	3 h

Tras realizar esta primera división se teme haber sido demasiado estricto con el tiempo dedicado a los temas 2 a 8. Previniendo que estos temas tengan una duración mayor de la esperada, se decide eliminar los temas 9 y 10 como temas independientes, introduciéndolos en el temario de la siguiente forma:

- El contenido del Tema 9 (Tiempo Real) se distribuye como puntos adicionales en aquellos temas en los que se tratan aspectos similares, por ejemplo, en el tema de planificación de procesos se añade un punto adicional de planificación en tiempo real.

- El contenido del Tema 10 (Redes de estaciones de trabajo) tendrá un introducción en una sesión de teoría, y posteriormente se realizará una práctica donde el alumno podrá adquirir las habilidades necesarias para la utilización de este tipo de sistemas.

Tras esta modificación, se dispone de un colchón de unas 4 ó 5 horas que se repartirán entre los temas en los que se prevé una mayor dificultad. Entre los que se destaca el Tema 2, que es necesario para poder desarrollar la segunda práctica de forma satisfactoria.

b) Prácticas.

La distribución de las horas se realiza igual que en ACSO1 por lo que el alumno dispone de 7 sesiones de 2 horas, y una adicional para la presentación de resultados o recuperación.
Las prácticas planteadas para la asignatura son:

Práctica 1. Programación SHELL en LINUX.	(1 sesión)
Práctica 2. Diseño de un interprete de órdenes.	(3 sesiones)
Práctica 3. Redes de estaciones de trabajo.	(3 sesiones)

Se puede observar el reducido número de prácticas. Esto es debido a la complejidad que tienen las prácticas y a la dificultad observada en los alumnos para programar en C.
La primera de ellas sirve para profundizar en el conocimiento y manejo del sistema UNIX, introducido en ACSO1.

En la segunda el alumno debe poner en práctica gran parte de las llamadas al sistema descritas en el tema 2, relacionándolas con los contenidos de temas siguientes. La práctica se entrega al final del curso y las sesiones de prácticas se utilizan para resolver una serie de problemas planteados.

La práctica 3 se centra en las redes de estaciones de trabajo. El objetivo fundamental de esta práctica es que el alumno sea capaz de resolver problemas de forma paralela y distribuida mediante un sistema de redes de estaciones de trabajo. Esta práctica completa el temario de la asignatura. Como en la práctica de shell, la evaluación se realiza una vez acabado el cuatrimestre y fuera del horario normal.

6.- EVALUACIÓN Y RESULTADOS

La evaluación se realiza mediante la realización de un examen (teoría + prácticas) y la presentación de trabajos de las prácticas, siendo en conjunto el 66% correspondiente a teoría y el otro 33% a prácticas.

Para poner la nota final se dispone de dos valores: la nota de un examen y la evaluación de las prácticas (asistencia + trabajos). La idea que se persigue es que aquel alumno tenga que realizar un mínimo de trabajo en prácticas.

Los resultados son los siguientes:

	Matriculados	Presentados	Aprobados	Nota Media (var)
ACSO1	189	163	157	7.11 (1.25)
ACSO2	146	105	104	7.26 (1.16)

Estos resultados corresponden a la primera convocatoria de ACSO1 y ACSO2.
Estos valores sorprendieron gratamente al profesorado ya que son superiores a los obtenidos en exámenes de la desaparecida ACSO, teniendo en cuenta que la complejidad prevista al preparar el examen era similar a los de la asignatura del plan de 1990.

7.- CONCLUSIONES.

La labor realizada en la puesta en marcha de ambas asignaturas se ha visto recompensada con el cumplimiento de los objetivos, los resultados obtenidos por los alumnos y la motivación e interés que algunos temas han despertado en ellos.

Esta motivación ha sido mayor en aquellos temas en los que los alumnos han visto una aplicación práctica de la teoría o han visto satisfecha una duda o curiosidad anterior, por ejemplo, cuál es el funcionamiento de una antememoria.

Estas observaciones hacen suponer que si se consigue enlazar cada elemento de la teoría con una aplicación práctica del mismo, el interés de los alumnos sería mayor.

La utilización de colecciones de transparencias especialmente diseñadas para la asignatura de ACSO1, ha permitido seguir un ritmo más alto de explicación y ajustarse mejor a la

planificación inicial de la asignatura, así como facilitar la labor de sincronización entre los diferentes profesores.

En ACSO2 no se han utilizado tantas transparencias y se ha podido comprobar que los alumnos no se han sentido tan cómodos. Aunque la atención prestada fuera similar a ACSO1, la participación ha sido menor en algunos casos. Este hecho nos obliga a replantearnos el método docente seguido en ACSO2 y pensar en la realización de transparencias o apuntes para aquellos temas que todavía no los tienen.

BIBLIOGRAFÍA

J.L. Hennessy, D.A. Patterson, Computer Architecture: A Quantitative Approach, 2ª edición, Morgan Kaufmann Publishers, 1996.

J.L. Hennessy, D.A. Patterson, Arquitectura de Computadores: Un Enfoque Cuantitativo. McGraw-Hill, 1994.

A.S. Tanenbaum , Albert B. Woodhull, Sistemas operativos: Diseño e implementación, 2ª edición, Prentice-Hall, 1998

D. Culler, J.P. Singh, A. Gupta, Parallel Computer Architecture: A Hardware / Software Approach, Morgan Kaufmann Publishers, 1998.

Brian W. Kernighan, Dennis M. Ritchie. "El lenguaje de programación C"; 2a. ed. - México [etc.] : Prentice-Hall Hispanoamericana, cop. 1991.

Silberschatz, J. Peterson, P.galvin "Sistemas Operativos: Conceptos fundamentales"; Addison Wesley 3ª edición 1994

Remy Card, Eric Dumas, Franck Mevel. "Programación Linux 2.0: API de sistema y funcionamiento del núcleo". Gestión 2000. ISBN 8-480-88207-7.

W.A. Halang, K.M. Sacha. "Real-time systems"; World Scientific, 1992

SIMULADOR PARA LA ENSEÑANZA DE LA PLANIFICACIÓN DINÁMICA DE INSTRUCCIONES

Sergio Sáez[1], Juan Luis Posadas[1], Pedro López[1]

[1]*Departamento de Informática de Sistemas y Computadores (DISCA)*
Facultad de Informática, Universidad Politécnica de Valencia
E-mail: { ssaez, jposadas, plopez } @disca.upv.es

RESUMEN: El presente trabajo ilustra el desarrollo y utilización de un simulador para la enseñanza de la gestión dinámica de instrucciones en un procesador segmentado a alumnos de cuarto curso de Ingeniería Informática. El objetivo de dicho simulador es facilitar la comprensión del algoritmo de Tomasulo y del hardware necesario para su implementación y funcionamiento, así como proporcionar la posibilidad de visualizar el estado del procesador (instrucción en cada etapa, bancos de registros de enteros y de coma flotante, estaciones de reserva, memoria, etc.) al final de cada uno de los ciclos de reloj necesarios para la ejecución de un determinado conjunto de instrucciones. Este simulador se utiliza en las prácticas de la asignatura de Arquitectura de Computadores II de la titulación de Ingeniero de Informática de la UPV.

1. INTRODUCCIÓN

En los actuales planes de estudios para la obtención del título de Ingeniero de Informática (II) en la Universidad Politécnica de Valencia (UPV), el estudio de la arquitectura del computador es competencia de dos asignaturas de segundo ciclo de carácter troncal, Arquitectura de Computadores 1 (ARQ1) y Arquitectura de Computadores 2 (ARQ2) [5], ubicadas en el séptimo y octavo semestre de la titulación, respectivamente.
Previamente a estas asignaturas el alumno ha cursado en el primer ciclo las materias Estructura de Computadores 1 (EC1) y Estructura de Computadores 2 (EC2) ubicadas en el segundo y tercer semestres, respectivamente, donde se estudia la organización general del computador y ya se introduce el concepto de procesador segmentado [1].
En la asignatura ARQ1 se define el concepto moderno de arquitectura, y se aplican las técnicas de segmentación a la unidad de instrucción del computador. Sólo se consideran las instrucciones enteras y los riesgos se solucionan mediante técnicas sencillas (inserción de ciclos de parada y adelantamiento).

La asignatura ARQ2 es una continuación de ARQ1 que comprende los siguientes objetivos:

- Aplicar las técnicas de segmentación a la unidad de instrucción del computador con operaciones multiciclo. Diferenciar las técnicas de **gestión** estática vs. **dinámica de**

instrucciones. Conocer y comprender las técnicas de gestión especulativa de instrucciones. Conocer y comprender el concepto de procesador superescalar.

- Distinguir los tipos de computadores orientados al procesamiento de vectores. Conocer y comprender los computadores vectoriales segmentados.
- Diferenciar los multiprocesadores de memoria compartida y los multicomputadores.
- Identificar los aspectos que más influencia tienen sobre las prestaciones de éstos.

La organización de la docencia de ambas asignaturas corresponde con 3 créditos de teoría y problemas y 1,5 créditos de trabajo en el laboratorio. Dicho trabajo consiste en la realización, de un modo coordinado con las clases de teoría y problemas, de un conjunto de prácticas que refuerzan la comprensión y aplican los conceptos estudiados.

En este trabajo se presenta un simulador desarrollado específicamente para la realización de la segunda práctica de la asignatura ARQ2 que trata sobre la planificación dinámica de instrucciones basada en el algoritmo de Tomasulo. Previamente, el alumno ya ha trabajado en la primera con la gestión estática de instrucciones. El simulador utilizado permite al alumno realizar la implementación de un planificador basado en el algoritmo de Tomasulo, así como comprobar su funcionamiento. Para ello, se proporciona al alumno el código de un simulador en el que falta por escribir el código correspondiente al planificador de instrucciones, trabajo que éste debe desarrollar en el laboratorio. Para verificar el correcto funcionamiento, la herramienta desarrollada permite la visualización ciclo a ciclo de la ejecución de las instrucciones, observándose el contenido de las diferentes etapas junto con el estado interno del procesador. La principal contribución de esta herramienta es precisamente la posibilidad de, no sólo analizar el funcionamiento de un planificador como permiten otras herramientas [2] sino también realizar su implementación de forma sencilla. El tiempo disponible para la realización de la práctica es de dos sesiones de dos horas cada una.

Este trabajo está organizado como sigue: en la sección 2 se presenta el simulador y el trabajo que debe realizar el alumno en el laboratorio. En la sección 3 se presenta el funcionamiento del simulador y los resultados y ficheros para visualización que genera. Finalmente, en la sección 4 se exponen algunas conclusiones de la experiencia.

2. SIMULADOR DE UN PROCESADOR SEGMENTADO CON PLANIFICACIÓN DINÁMICA DE INSTRUCCIONES

El objetivo del simulador presentado en este trabajo es ofrecer al alumno la posibilidad de analizar de una forma didáctica y visual el correcto funcionamiento del algoritmo de Tomasulo para la ejecución de instrucciones con operaciones multiciclo donde se permite alterar su orden de lanzamiento a ejecución. El simulador ha sido implementado en lenguaje C bajo el sistema operativo LINUX.

El algoritmo de Tomasulo descompone la ejecución de instrucciones en tres fases. En primer lugar en la fase de lanzamiento (fase Issue), se decodifica la instrucción, se envía a la estación de reserva del operador correspondiente, y se comprueban las dependencias de datos, transfiriendo desde los registros los operandos que estén disponibles (para prevenir riesgos WAR) o bien copiando el identificador (marca) del operador del que depende la instrucción en curso (para resolver riesgos RAW). También se reserva el registro destino, indicándole el

identificador del operador correspondiente. Seguidamente, en la fase de ejecución, los operadores toman instrucciones de las estaciones de reserva que tengan todos sus operandos disponibles, y realizan la operación correspondiente. En caso de que ninguna estación de reserva tenga sus operandos disponibles, el operador está ocioso. Finalmente, en la fase de escritura de resultados (fase Write Back), los operadores escriben sobre el bus común de datos el resultado de la operación y su identificador, de tal manera que todas la operaciones pendientes de ese resultado lo toman del bus y lo transfieren a la estación de reserva correspondiente.

Al alumno se le proporciona un simulador en el que el código correspondiente a las fases de lanzamiento y escritura de resultados se ha eliminado, dejando únicamente un esqueleto de dichos procedimientos. También se le proporciona un algoritmo en pseudocódigo en el que se definen todas la acciones que tienen lugar en cada una de las fases del algoritmo. El trabajo del alumno consiste en escribir dichos procedimientos y conseguir un simulador que funcione.

La práctica por tanto está dividida principalmente en dos apartados: una primera parte donde el alumno debe familiarizarse con la estructura de datos utilizada en la implementación del simulador y acabar de desarrollar el código correspondiente al algoritmo de Tomasulo, y una segunda parte donde el alumno tiene que comprobar el funcionamiento correcto del código que ha generado simulando la ejecución de varios programas en ensamblador. La primera parte de implementación obliga al alumno a comprender cómo funciona el algoritmo así como analizar qué recursos se necesitan para diseñar un procesador que lo aplique, debiendo proporcionar soluciones a problemas tales como la escritura simultánea en el bus común de datos por parte de los operadores, los cuales no se abordan con los métodos tradicionales. En la segunda parte, con la simulación de la ejecución de programas y posterior visualización de los resultados obtenidos, el alumno puede comprobar si la versión del algoritmo que ha realizado es correcta.

Los resultados que se generan en una simulación son ficheros ".html" que permiten visualizar el cronograma de ejecución de las instrucciones junto con el estado del procesador ciclo a ciclo. En el caso de obtenerse un funcionamiento incorrecto, el alumno puede observar paso a paso la evolución de la ejecución de las instrucciones y detectar dónde falla el algoritmo, para así proceder a su rectificación y volver a simular. Este método permite profundizar de forma sencilla en el estudio y comprensión del algoritmo al poderse comprobar de forma práctica y visual cualquier situación teórica.
La evaluación de la práctica y comprensión que el alumno tiene que haber adquirido se realiza a través de un cuestionario escrito con preguntas relativas a la ejecución de los programas ejemplo y así forzar a su seguimiento visual ciclo a ciclo.

La experiencia llevada a cabo ha sido satisfactoria y los alumnos agradecen el material disponible ya que les facilita la comprensión de los conceptos estudiados en clase, además de permitirles preparar el examen con mayor facilidad al disponer de múltiples ejemplos de ejecución que ellos mismos pueden generar y visualizar tantas veces como necesiten.

3. VISUALIZACIÓN DE LOS RESULTADOS OBTENIDOS

El simulador, denominado DLX/FP, está basado en el procesador DLX [3] [4] con instrucciones de coma flotante. Permite interpretar código ensamblador desarrollado con un conjunto de instrucciones enteras reducido, e instrucciones de coma flotante aritméticas y de carga/almacenamiento de simple precisión.

Una vez implementado y compilado el simulador, se comprueba su funcionamiento mediante una serie de ejemplos. Uno de ellos es el código en ensamblador correspondiente al bucle que

realiza el producto escalar de un entero "*a*" por un vector "*x*" y elemento a elemento se suma el resultado con un vector "*y*". En el código desarrollado los vectores sólo son de 2 elementos (lo que supone dos iteraciones en el bucle) pues es suficiente para comprobar el funcionamiento del algoritmo y así se reduce el número de ficheros resultado. El nombre del fichero en ensamblador a ejecutar se pasa como parámetro al simulador (por ejemplo: dlxfp –f saxpy.s). Entonces, el simulador genera un fichero "index.html" con las características del simulador y el código que se ha ejecutado y, además, dos ficheros "ciclox.html" y "estadox.html" por cada ciclo de reloj que se necesitaría en la ejecución real del código, donde x indica el número de ciclo.

De esta forma, los resultados de la simulación quedan almacenados en disco, y utilizando un visualizador html puede reproducirse de forma sencilla la ejecución del código ensamblador mediante la presentación del cronograma de las instrucciones y del estado del procesador.

El fichero "index.html" para el ejemplo indicado puede observarse en la figura 1. Presenta la configuración con la que se ha realizado la simulación: un banco de ocho registros de coma flotante, tres estaciones de reserva de suma y resta, dos estaciones de reserva de multiplicación y división, tres tampones de lectura, tres tampones de escritura, un operador de suma y resta con un tiempo de cuatro ciclos, un operador de multiplicación y división con un tiempo de siete ciclos y, por último, una memoria de datos con un tiempo de acceso de tres ciclos donde puede apreciarse su contenido una vez inicializada. Además, también se presenta la memoria de instrucciones con el código ensamblador simulado. En este caso puede observarse el código correspondiente al bucle *saxpy*. Es importante destacar que el simulador permite cambiar la configuración para así poder comparar diferentes resultados dependientes de ésta (número estaciones de reserva, tiempo de los operadores, etc.).

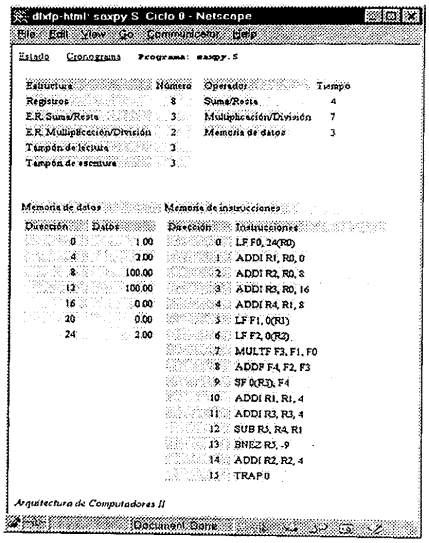
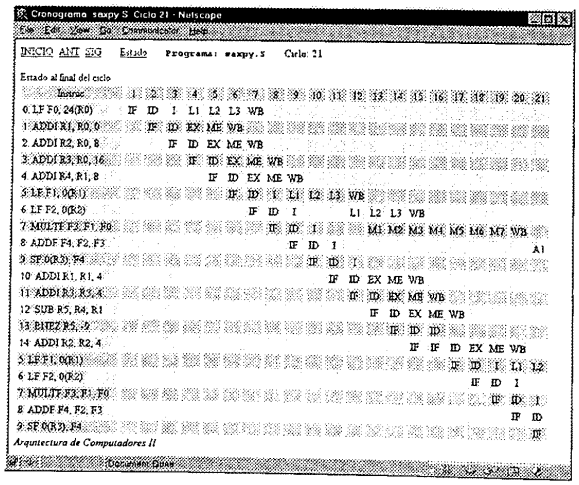
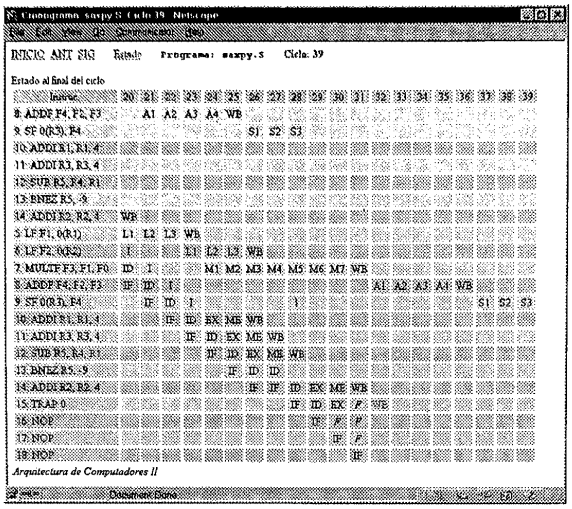


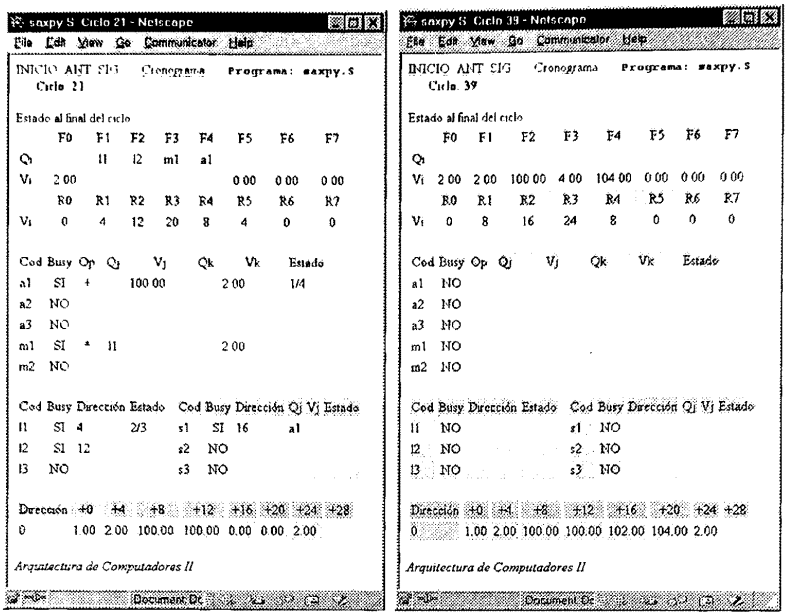
Figura 1: fichero index.html





Figuras 2 y 3: ficheros ciclo21.html y ciclo39.html

En las figuras 2 y 3 puede observarse el cronograma de ejecución de las instrucciones correspondiente al bucle *saxpy*. El simulador ha generado un total de 39 ciclos de reloj que son, empleando la configuración indicada, el tiempo global de ejecución del código ejemplo. En las figuras se muestra el estado del cronograma en el ciclo 21 y en el último ciclo 39. Puede observarse que cada ventana dispone en la parte superior de un enlace al ciclo anterior y otro al posterior, permitiendo fácilmente avanzar o retroceder ciclo a ciclo en la ejecución de las instrucciones. Viendo los diferentes cronogramas generados (uno por cada ciclo) se comprueba cómo el *hardware* lanza a ejecución instrucciones posteriores a otras que se han parado y cómo se altera dinámicamente el orden de la ejecución, evitando que el hecho de parar una instrucción afecte a las que le siguen. En los cronogramas también puede apreciarse la existencia de otro enlace con nombre *estado* que sirve para abrir una nueva ventana donde se visualiza el estado del procesador en ese ciclo.



saxpy.S Ciclo 21 - Netscape

File Edit View Go Communicator Help

INICIO ANT SIG Cronograma Programa: saxpy.S
Ciclo 21

Estado al final del ciclo

	F0	F1	F2	F3	F4	F5	F6	F7
Qi		l1	l2	m1	a1			
Vi	2.00					0.00	0.00	0.00

	R0	R1	R2	R3	R4	R5	R6	R7
Vi	0	4	12	20	8	4	0	0

Cod	Busy	Op	Qj	Vj	Qk	Vk	Estado
a1	SI	+		100.00		2.00	1/4
a2	NO						
a3	NO						
m1	SI	*	l1			2.00	
m2	NO						

Cod	Busy	Dirección	Estado	Cod	Busy	Dirección	Qj	Vj	Estado
l1	SI	4	2/3	s1	SI	16		a1	
l2	SI	12		s2	NO				
l3	NO			s3	NO				

Dirección	+0	+4	+8	+12	+16	+20	+24	+28
0	1.00	2.00	100.00	100.00	0.00	0.00	2.00	

Arquitectura de Computadores II

saxpy.S Ciclo 39 - Netscape

File Edit View Go Communicator Help

INICIO ANT SIG Cronograma Programa: saxpy.S
Ciclo 39

Estado al final del ciclo

	F0	F1	F2	F3	F4	F5	F6	F7
Qi								
Vi	2.00	2.00	100.00	4.00	104.00	0.00	0.00	0.00

	R0	R1	R2	R3	R4	R5	R6	R7
Vi	0	8	16	24	8	0	0	0

Cod	Busy	Op	Qj	Vj	Qk	Vk	Estado
a1	NO						
a2	NO						
a3	NO						
m1	NO						
m2	NO						

Cod	Busy	Dirección	Estado	Cod	Busy	Dirección	Qj	Vj	Estado
l1	NO			s1	NO				
l2	NO			s2	NO				
l3	NO			s3	NO				

Dirección	+0	+4	+8	+12	+16	+20	+24	+28
0	1.00	2.00	100.00	100.00	102.00	104.00	2.00	

Arquitectura de Computadores II

Figuras 4 y 5: estado del procesador en ciclos 21 y 39

En las figuras 4 y 5 pueden verse las ventanas correspondientes al estado del procesador al final de los ciclos de reloj 21 y 39.
En las ventanas del estado del procesador se presenta: el banco de registros de enteros; el banco de registros de coma flotante con las marcas (Qj, Qk) en ese instante asociadas a cada uno de ellos (la marca de un registro depende de la estación reservada por la instrucción que va a escribir en dicho registro); las estaciones de reserva, indicándose las operaciones que las ocupan, la disponibilidad de los operandos (valores, o, en su defecto, las marcas correspondientes) y el número de ciclos ejecutados en el caso de estar en ejecución dichas operaciones; los tampones de lectura y escritura con las direcciones de acceso; y, por último, el contenido de la memoria de datos para permitir comprobar si al final de la ejecución del código se han obtenido los resultados correctos.

4. CONCLUSIONES

En este trabajo se ha presentado un simulador para la gestión dinámica de instrucciones que se está utilizando en las prácticas de la asignatura Arquitectura de Computadores 2 de la Facultad de Informática de la UPV. El uso de dicho simulador ha facilitado enormemente al alumno el estudio de los algoritmos para la planificación dinámica de instrucciones al permitirle modificar fácilmente su código para implementar un algoritmo de planificación así como visualizar ciclo a ciclo el cronograma de ejecución de las instrucciones y observar al mismo tiempo el estado del procesador al final de cada uno de los ciclos de reloj.

El alumno puede comprobar de una forma práctica y con diferentes configuraciones del procesador como se altera dinámicamente el orden de ejecución de las instrucciones

permitiéndose que instrucciones posteriores a una parada no se detengan y acaben antes su ejecución.

REFERENCIAS

[1] J.L. Posadas, A. Robles. Innovación Educativa en la Enseñanza del Procesador Segmentado. JENUI 99

[2] J.L. Hennessy, D.A. Patterson. Herramientas y Simuladores: http://www.mkp.com/books_catalog/ca/hp2e_res.asp#other

[3] J.L. Hennessy, D.A. Patterson. Arquitectura de Computadores, un enfoque cuantitativo. New York. Mc. Graw Hill. 1993

[4] D.A. Patterson, J.L. Hennessy. Organización y Diseño de Comput. Mc. Graw Hill. 1995

[5] WEB de las asignaturas ARQ1 y ARQ2: http://dlxv.disca.upv.es

ÁREA DE CARTOGRAFÍA,GEODESIA Y FOTOGRAMETRÍA.

Responsable: Dr. Miguel Angel Bernabé Poveda.
(Universidad Politécnica de Madrid)
E-mail: mab@nivel.euitto.upm.es

APLICACIÓN EN ENTORNO MATLAB PARA MANIPULACIÓN DE ECUACIONES RESULTANTES DE OBSERVACIONES GPS

Pilar Abad Real[1]

[1]*Profesora Titular de Escuela Universitaria*
e-mail: pabad@cicei.ulpgc.es

RESUMEN: Esta aplicación está formada por una serie de programas para el análisis de datos GPS, utilizando los observables pseudodistancia y medida de fase para estimar la posición tridimensional de estaciones en Tierra. El software está diseñado como una toolbox para MATLAB. El objetivo fundamental es trabajar sobre modelos no diferenciados, para evitar la pérdida de información inherente al uso de combinaciones lineales, aunque también se tienen en cuenta los modelos de simples dobles y triples diferencias. El software está compuesto de distintos módulos que ejecutan las funciones de; preparación de datos para el procesado, generación de las órbitas precisas, cálculo de las matrices de coeficientes para los distintos modelos y análisis y resolución usando mínimos cuadrados.

1.- INTRODUCCIÓN.

El trabajo consiste en la realización de una aplicación que permita generar las ecuaciones ya linealizadas asociadas a las observaciones de los satélites de la constelación GPS, tanto para el observable fase, en una o ambas frecuencias como para el observable pseudodistancia y para los modelos no diferenciados, los de simples, dobles y triples diferencias, a partir de los datos en el formato RINEX-2 (Receiver INdependent Exchange Format #2)

Los objetivos son:

1. Disponer de los modelos para su manipulación, estudio y resolución utilizando los recursos de tratamiento de ecuaciones de que dispone el software MATLAB.
2. Obtención de la estructura de la matriz de coeficientes asociada a cada modelo, en términos de dispersidad, y la variación en esta estructura al realizar reordenaciones de sus filas o columnas.
3. Visualización gráfica de los valores de la matriz de coeficientes y términos independientes, que permita detectar posibles datos erróneos.
4. Resolución y estudio detallado del modelo no diferenciado para una única estación.

Sobre el último de los objetivos es importante explicar que la supresión de la degradación en la señal (SA) desde primeros de Mayo hace aún más interesante el estudio de la determinación absoluta de la posición. Hasta ahora la introducción de la disponibilidad selectiva (SA), degradaba la precisión que se puede obtener en la posición usando GPS desde una única estación, por eso no se le había prestado mucha atención a este modelo y la mayoría de los usuarios, para conseguir un posicionamiento preciso utilízan el posicionamiento relativo respecto a una estación conocida, para eliminar los efectos de la SA, y reducir los errores comunes a las dos estaciones, esto supone usar medida de fases para determinar la posición del observador y que las ambigüedades asociadas se resuelvan de alguna forma. La distancia entre el receptor móvil y el receptor fijo puede variar desde unos pocos kilómetros hasta cientos de ellos, y a medida que la separación aumenta, los errores dependientes de ella también lo hacen y la resolución de la ambigüedad de la portadora puede experimentar grandes cambios, con la considerable perdida de precisión. Veremos más adelante como se puede aprovechar mucho más el uso de los datos obtenidos desde un solo receptor y las ventajas que esto puede reportar, más aún sin el inconveniente de la degradación de la señal.

Mediante la manipulación de los modelos de ecuaciones se pueden estudiar estrategias para eliminar o mitigar las diferentes fuentes de error en cortas, medias o largas distancias, obtener el máximo provecho de los datos según la diferente metodología de observación, o incluso mejorar el modelo funcional, y todo ello puede ser posible si se atienden los siguientes aspectos:

1. Resolución de ambigüedades usando algoritmos más estables y precisos.
2. Resolución de los sistemas por métodos que tengan en cuenta la dispersidad de la matriz asociada.
3. Trabajar con efemérides precisas para que los errores en la posición de la órbita no influyan de forma relevante.
4. Comparación de los resultados que se obtienen dependiendo de distintas hipótesis en el modelo funcional.
5. Comparación de resultados que nos permitan contrastar modelos ionosféricos.
6. Uso de modelos no diferenciados, con un sólo receptor, que pueden dar precisiones inferiores al metro.

2.-ANTECEDENTES Y DESARROLLO DE LA APLICACIÓN

En la actualidad casi todo el software de post-proceso existente está basado en la utilización de dobles diferencias y el uso de dos receptores simultáneamente, el tratamiento posterior, mediante lo que se conoce como posicionamiento relativo determina la posición con gran exactitud, y este tratamiento es correcto si la línea base es de pocos kilómetros de largo, en cuyo caso se puede suponer que las condiciones son similares en ambas estaciones, pero no es tan cierto cuando la longitud aumenta. Por otra parte la solución obtenida depende fuertemente de las estaciones que se toman como referencia, lo que implica nuevos cálculos si se cambian dichas estaciones, y distorsiones que afectan a todas las estaciones calculadas a partir ella. La solución obtenida por un solo receptor en tiempo real, tiene muy poca precisión, del orden de 100 m, debido al uso de efemérides poco precisas, y al tratamiento únicamente del observable pseudodistancia. La introducción de efemérides precisas y correcciones a los relojes de los satélites, junto con un tratamiento post-proceso de las observaciones desde un solo receptor, usando tanto las ecuaciones de fase como de pseudodistancia pueden mejorar esas precisiones, hasta llegar al nivel del metro, en este sentido existe un software desarrollado por el Jet Propulsión Laboratory (JPL) que trabaja sobre plataformas HP900/700 de UNIX, denominado GIPSY II.(GPS-Inferred Positioning System). Las soluciones obtenidas por este tipo de posicionamiento absoluto preciso son independientes en cada estación, lo que permite un más fácil cambio de marco de referencia sin que sea necesario procesar de nuevo todas las

estaciones. Para todo el software de post-proceso existente los recursos de CPU al resolver gran número de estaciones simultáneamente crecen en proporción cúbica al número de estas, lo que hace difícil el tratamiento de redes con muchas estaciones. Con el posicionamiento absoluto preciso los recursos de CPU crecen de forma lineal, lo que permite el análisis de grandes redes con equipos ordinarios. Esta forma de procesar los datos puede satisfacer a gran número de usuarios.

El National Geodetic Survey (NGS) genera la posición precisa, y el estado de los relojes de los satélites del Sistema de Posicionamiento Global (GPS), en intervalos de 15 minutos en el formato estándar SP3 (Standard Product #3) , que fundamentalmente consta de dos registros diferentes, el primero de ellos marcado con P incluye la posición y el estado del reloj para cada satélite de la constelación, y el segundo que es opcional y está marcado con una V incluye las variaciones en la velocidad y en el estado. El movimiento de un satélite se describe a partir de seis condiciones iniciales (3 de posición y 3 velocidades o 6 elementos orbitales de la órbita kepleriana), mas el modelo de fuerzas que actúan sobre el satélite a lo largo de su trayectoria. Un modelo preciso para su movimiento requiere conocer la influencia gravitatoria del Sol la Luna y la Tierra, y también influencias debidas a la presión de radiación solar y emisiones de gas de los propios satélites. En un principio la trayectoria puede ser generada mediante expresiones analíticas o integrando las ecuaciones del movimiento, siendo esta última modalidad la más utilizada. La posición del satélite en función del tiempo queda pues registrada en la tabla de efemérides generada en dicha integración.

Las etapas de desarrollo serán las siguientes:

1. Determinación de las órbitas de los satélites a partir del mensaje de navegación.
2. Determinación de las órbitas con efemérides precisas.
3. Modelo general de ecuaciones a partir de los datos de navegación, para la estación de que se trate, sea esta la fija o la móvil. Pudiendo generarlas a partir de las órbitas con efemérides precisas o sin ellas.
4. Resolución del modelo no diferenciado.
5. Modelos calculados de simples dobles y triples diferencias, para la estación móvil.
6. Visualizaciones de estructura, y de coeficientes de las matrices de los modelos.

3.-MODULOS DE LA APLICACION

Cada día se refinan más los métodos de resolución de sistemas de ecuaciones, y es conveniente estudiar si pueden representar mejoras, en precisión rapidez o almacenamiento, en casos concretos.
El resultado del trabajo, permitirá analizar posteriormente con las herramientas de tratamiento de ecuaciones de que dispone MATLAB, cómo se comportan los modelos frente a distintos métodos de resolución, tanto directos como iterativos, con la aplicación o no de factorizaciones y reordenaciones.

Respecto a la resolución del sistema no diferenciado para una única estación de referencia se esperan obtener resultados que confirmen que efectivamente la precisión utilizando efemérides precisas y correcciones al estado del reloj esté alrededor del metro.

A continuación se describen los módulos que conforman la aplicación.

EFEMER1 Generación de un fichero con las coordenadas y el radio orbital de cada uno de los satélites a partir de los datos del mensaje de navegación del fichero de observaciones. El

intervalo entre épocas y el número de épocas se introducirán como valores de entrada del programa.

EFEMER2 Generación de un fichero con las coordenadas y el radio orbital de cada uno de los satélites a partir de los datos del fichero SP3. El intervalo entre épocas y el número de épocas se introducirán como valores de entrada del programa.

COEFICI Generación de un fichero que contiene las distancias a cada satélite en cada época para un valor inicial aproximado de las coordenadas del a estación y los coeficientes de las incógnitas de posición dx,dy,dz para cada satélite correspondientes a la linealización de la ecuación de la distancia. Utiliza como entrada el fichero generado en el paso anterior.

SELECCI Generación de ficheros para separar los datos que se refieren a los mismos satélites (cada vez que entra un satélite o desaparece uno).

OBSERVA Reordenación de los datos de observación por épocas, y generación de un fichero con los datos de fase de la portadora y otro con los datos de pseudodistancia que constituirán los términos independientes de las ecuaciones de fase en una o dos frecuencias y de las ecuaciones de distancia.

BLOQUES Generación de los bloques por cada satélite que compondrán la matriz de los diferentes modelos. BLOQUEA1 contiene los coeficientes correspondientes a las incógnitas de posición. BLOQUEA2 los correspondientes a las incógnitas de ambigüedades, BLOQUEA3 los correspondientes a las incógnitas de errores del reloj del receptor , BLOQUEA4 los coorespondientes a las incógnitas de errores de los relojes de los satélites, BLOQUEA5 los correspondientes al retardo troposférico e ionosférico, y BLOQUEA6 los correspondientes a un error aleatorio propio del proceso observacional.

FASCODXX Formación a partir de los Bloques anteriores de las distintas matrices asociadas a los modelos para un número determinado de épocas, habiendo un mínimo de épocas requerido para que el sistema formado tenga al menos el mismo número de ecuaciones que de incógnitas. Una vez formada la matriz representa gráficamente su estructura dispersa (fig.1) y los valores de sus filas(fig. 2).

PROCESO Tratamientos diferentes, calculo de rangos, y números de condición del modelo no diferenciado. Obtención de soluciones y residuos.

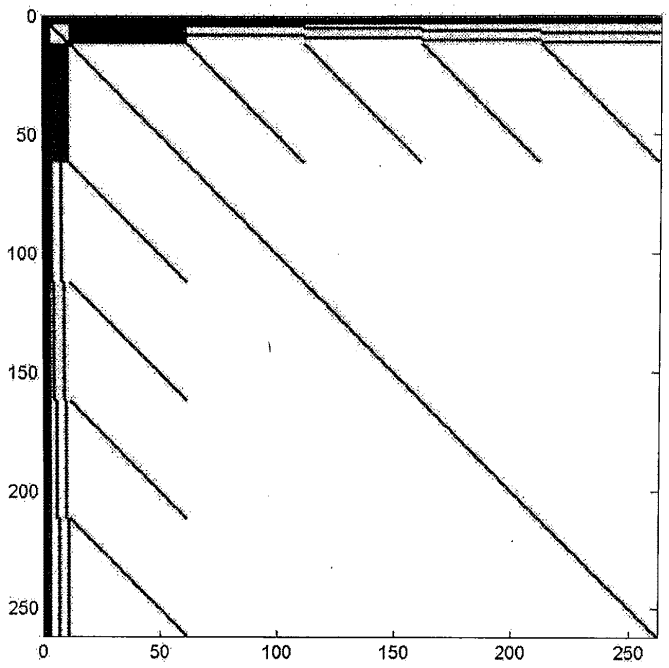


Fig. 1. Estructura delas ecuaciones normales

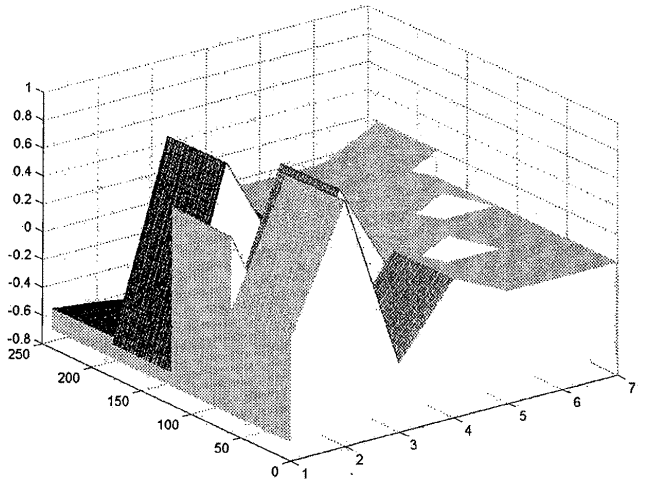


Fig. 2. Coeficientes de la matriz de diseño para cuatro satélites

4.-BILBLIOGRAFÍA

[1] MATLAB.5.3 *The Language of Technical Comp*uting. MathWorks 1999.

[2] http://www-gpsg.mit.edu. The GPS and Geodynamics group in the Massachusetts Institute of Technology

[3] http://sinbad.ngs.noaa.gov/GPS/GPS.html National Geodetic Survey

[4] Günter Seeber. *Satellite geodesy foundations, methods and applications.* Berlin ; New York :Walter de Gruyter, 1993

[5] Pierre Héroux and Jan Kouba. "GPS Precise Point Positioning with a Difference". Geomatics'95 Ottawa, Ontario ,Canada, June 13-15-1995

[6] E.T. Hesper. "Investigation on the use of GPS for geodetic and orbit determination applications". Delft University Press, February 1991

[7] A. E. Roy. O*rbital motion. (Third edition).* University Glasgow 1988

[8] Gene H. Golub, Charles F. van Loan. *Matrix computations. -- 2nd e*d.. Baltimore : Johns Hopkins University Press, 1989

[9] José Luis de la Fuente O'Connor. *Tecnologías computacionales para sistemas de ecuaciones, optimización lineal y entera.* Barcelona : Reverté, 1993

[10] S. Chandrasekrant and I.C.F. Ipsen. "On The Sensitivity of Solution components in Linear Systems Of Equations". SIAM J. Matrix ANAL. APPL. vol. 16, nº 1, pp. 93-112, January 1995

[11] David R. Hill.*Experiments in computational matrix algebra.* New York : Random House,1988

CONTENIDOS DE LA ASIGNATURA "FUNDAMENTOS Y LENGUAJES INFORMÁTICOS" Y ALGUNAS EXPERIENCIAS EN LA APLICACIÓN DE INTERNET EN SU DOCENCIA

Adolfo Lozano Tello [1]

[1] *Profesor Asociado del Area de Lenguajes y Sistemas Informáticos. Departamento de Informática. Universidad de Extremadura.*
e-mail: alozano@unex.es

Resumen: En este artículo se presentan algunos contenidos e ideas sobre la asignatura de "Fundamentos y Lenguajes Informáticos" que se da por primera vez en la recién creada titulación de "Ingeniero en Geodesia y Cartografía" en la Universidad de Extremadura. Se exponen también brevemente algunas experiencias en el uso de herramientas de Internet aplicadas a la docencia de esta asignatura.

1.- INTRODUCCIÓN.

La titulación de "Ingeniero en Geodesia y Cartografía" ha empezado este año en la Universidad de Extremadura. El alumnado viene en su práctica totalidad del Centro Universitario de Mérida desde el que hace muchos años (1975) se imparte la titulación "Ingeniero Técnico en Topografía". Ante esta situación, muchos antiguos alumnos de título medio han decidido cursar los estudios superiores al cabo de unos años. La consecuencia es que muchos tienen trabajo consolidado (cuando no su propia empresa) y deben realizar un esfuerzo considerable para seguir las clases.

Una de las dos asignaturas de informática que se da en este primer año de la titulación superior es "Fundamentos y Lenguajes Informáticos" (troncal, cuatrimestral). El cometido de la asignatura se ha centrado en el aprendizaje de los conceptos fundamentales de programación. Aunque el contenido no podía diferir mucho de los descriptores que posee la asignatura, se ha intentado que el alumno enfoque su aprendizaje mediante el empleo de ejemplos y prácticas afines a su trabajo.

La otra asignatura de este primer curso es "Base de Datos" (troncal, cuatrimestral), en la que en principio no ofrece ninguna relación con la primera. De igual forma, tanto las prácticas exigidas como los ejemplos vistos en clase han perseguido un vínculo con temas de Topografía, Geodesia y Cartografía. Este artículo expone los contenidos y problemas de la asignatura de "Fundamentos y Lenguajes Informáticos" y dejo para otra ocasión los de "Base de Datos", aunque la problemática del uso de Internet en la docencia de la asignatura (expuesta en la sección 4) es común a ambas.

En la sección 2 de este artículo se describe brevemente el temario de la asignatura y la experiencia de este primer año; en la sección 3 se comenta la idea de utilizar un lenguaje de programación visual como base para la enseñanza de la asignatura; y en la sección 4 se describe las ventajas e inconvenientes de emplear Internet como ayuda en la docencia para el alumno.

2.- CONTENIDO DE LA ASIGNATURA "FUNDAMENTOS Y LENGUAJES INFORMÁTICOS".

El objetivo que se ha perseguido en esta asignatura es ofrecer la base para que los ingenieros en Geodesia y Cartografía conozcan los métodos fundamentales de programación en un lenguaje estructurado, y puedan aplicarlos en la resolución de problemas topográficos y cartográficos. Se ha escogido el lenguaje de programación C por su versatilidad, facilidad y adecuación en el cálculo científico. El contenido del temario se expone a continuación de forma resumida:

TEMA 1: Conceptos básicos de los lenguajes de programación. (Conceptos previos sobre programación. Lenguaje de programación: sintaxis y semántica).

TEMA 2: Introducción al concepto de programa. (Estructura de un programa. Constantes y variables. Declaración de variables. Tipos de datos. Instrucción de asignación).

TEMA 3: Operadores y Expresiones.(Operadores aritméticos. Operadores lógicos. Operadores relacionales).

TEMA 4: Sentencias de control de flujo. (Estructuras condicionales. Estructuras iterativas.).

TEMA 5: Funciones.(Definición y concepto de función. Transferencia de información entre funciones. Variables locales y globales).

TEMA 6: Vectores. (Definición de vector. Operaciones con vectores. Paso de vectores como argumentos a funciones. Vectores bidimensionales. Vectores de caracteres).

TEMA 7: Sistemas de ficheros. (Definición de estructura. Estructuras anidadas. Operaciones sobre estructuras. Paso de estructuras como parámetros de funciones. Archivos de texto y binarios. Archivos de acceso secuencial y de acceso directo. Archivos binarios).

TEMA 8: Funciones gráficas. (Funciones de inicialización de la memoria de vídeo. Funciones de representación gráfica de datos).

TEMA 9: Punteros y variables dinámicas.(Conceptos básicos. Direcciones de memoria. Punteros. Punteros y vectores. Punteros y estructuras. Matrices dinámicas).

La asignatura consta de dos clases teóricas y una práctica semanales, aunque el único criterio de evaluación que se ha seguido es la entrega de una práctica (http://webepcc.unex.es/~alozano/obligat1.doc) que el alumno debe desarrollar, relacionada con temas topográficos. El método de evaluación se realiza de la siguiente forma: a partir de las especificaciones indicadas en el enunciado de la práctica, el examen consiste en alguna modificación de su práctica en un tiempo determinado. Esta modificación de la práctica frente al

ordenador garantiza (en cierta medida) que es el alumno el que ha desarrollado la práctica, o que por lo menos domina el lenguaje y los conceptos de programación. Si superan esta prueba, la nota se incrementará respecto a las opciones que han ido añadiendo de forma voluntaria a los requisitos básicos del enunciado.

La experiencia de este primer año ha sido muy positiva. La mayoría de los alumnos se han tomado con gran interés la asignatura y, según me comentan[2], los conceptos aprendidos en clase los están usando para desarrollar algunos programas simples en su trabajo.

3.- ELECCIÓN DE UN LENGUAJE DE PROGRAMACIÓN ADECUADO PARA EXPLICAR LOS CONTENIDOS TEÓRICOS.

Una parte muy importante de la asignatura es el aprendizaje de un lenguaje de programación, que será utilizado para mostrar ejemplos de pequeños programas sobre los conceptos teóricos indicados en la sección anterior. Por esta razón, la elección de un lenguaje adecuado para exponer los ejemplos es fundamental.

A pesar de que la mayoría de los alumnos no han tenido ningún problema al enfrentarse por primera vez a un lenguaje de programación, también hemos discutido la idoneidad del lenguaje que nos ha servido de ejemplo para explicar la nociones básicas de programación. La forma en la que se han explicado las instrucciones y sintaxis de C ha sido de la forma más clara y básica posible (se podría haber escogido PASCAL u otro, pero en mi opinión si se explica de forma ordenada y clara no tiene porqué presentar mayor problema).

Como he comentado, el objetivo de la asignatura es el aprendizaje de los conceptos básicos de la programación (variables, estructuras de control de flujo, funciones, etc.), pero nos surgió la pregunta de si era aconsejable aprenderlos usando un lenguaje en modo texto basado en MS-DOS (o UNIX), o si por el contrario era más útil usar un lenguaje en un entorno visual bajo *Windows*, como en mi opinión es la tendencia actual. Esta opinión también me la han hecho saber varios alumnos que trabajan en empresas topográficas.

Por un lado, parece que el mayor inconveniente que surge al explicar los fundamentos de programación en un lenguaje visual es que se necesitan explicar otros conceptos como formas, objetos, propiedades, métodos y eventos, que quizás se aparten del temario y descriptores de la asignatura. En cambio, algunos alumnos me han comentado que les sería mucho más útil conocer lenguajes como *VisualBasic* para desarrollar aplicaciones o módulos para los programas de Topografía que utilizan normalmente. Además, la colocación de objetos que se sitúan en ventanas y en los que se pueden realizar una serie de acciones parece ser muy intuitivo para el alumno, ya que la mayoría están familiarizados con el entorno *Windows*. Como experiencia, el próximo año voy a seguir el mismo temario, pero utilizando como lenguaje base *VisualBasic* para los ejemplos y para desarrollar la práctica obligatoria. La única modificación será explicar en temas iniciales los conceptos propios de los lenguajes visuales.

4.- VENTAJAS Y PROBLEMAS DEL USO DE HERRAMIENTAS DE INTERNET EN LA ASIGNATURA.

[2] El reducido número de alumnos (49) que se han matriculado este primer año en la nueva titulación de Geodesia y Cartografía, y el alto porcentaje de ellos que trabajan en este campo, me ha permitido obtener valiosos comentarios sobre la aplicación real de los contenidos de las asignaturas.

En la introducción se ha comentado que gran parte del alumnado de esta titulación compagina su trabajo con la asistencia a clase, unido a que más de un ochenta y cinco por ciento no tienen su residencia habitual en la ciudad de Cáceres. Estos problemas son superados con el gran interés que muestran para la obtención del título de grado superior. Teniendo en cuenta este escenario, la asistencia a las clases prácticas no es obligatoria, aunque por otro lado no se puede decir que ésta sea baja (depende de la etapa del curso, pero suele ser sobre el 70 %).

Aunque he empleado Internet en otras titulaciones, la demanda que se ha producido en las asignaturas de Geodesia y Cartografía ha sido muy superior. En ningún momento he intentado plantear la asignatura como si nuestra universidad fuera a distancia, pero sí me he preocupado en mantener la información actualizada de material docente y avisos en una página *web*, y de atender con prontitud la consultas por correo electrónico. En concreto, ofrezco al alumno los siguientes "servicios":

- Una página web (http://webepcc.unex.es/~alozano/fundam.htm) donde se ofrece el temario, enunciados de prácticas, avisos de convocatorias de exámenes y otros, y las calificaciones de cada convocatoria.

- Correo electrónico para tutorías, donde los alumnos lo utilizan para consultarme conceptos concretos o incluso dudas sobre el código de una práctica casi terminada.

- Correo electrónico para la entrega de prácticas, permitiendo al alumno entregar su práctica desde casa.

Estas posibilidades ofrecen una agradecida serie de ventajas para el alumnado, sobre todo teniendo en cuenta la idiosincrasia de su composición en esta titulación. En cambio, estas ventajas debemos apreciarlas en su justa medida, y teniendo en cuenta los problemas y esfuerzos que llevan relacionados. En este aspecto, indico a continuación las contraprestaciones y algunos problemas que me han surgido, y que con toda seguridad no serán los únicos:

- El mantenimiento de la página *web* conlleva un continuado y periódico gasto de tiempo.
- Algunos alumnos me han cuestionado en cierta forma si es legal mostrar las calificaciones obtenidas a cualquier usuario de la red.
- Las tutorías por e-mail implican un gran gasto de tiempo en explicar los conceptos (o programas enteros mal realizados) de forma escrita.
- Y sobre todo, el gran problema me ha surgido en el envío de las prácticas adjuntas al correo electrónico, donde se han dado (repetidas veces), casos como:

    ⇒ Enviar el mensaje indicando que entregan la práctica, olvidando adjuntar tal práctica.
    ⇒ El correo en vez de tener la práctica adjunta tiene un virus ejecutable (por suerte tengo activado siempre un antivirus).
    ⇒ La práctica es entregada varios días después de la fecha de entrega, y ellos "aseguran" que la enviaron al servidor dentro de los días de entrega.

A estos problemas señalados hay que añadirle que los alumnos no poseen en nuestro Centro correo electrónico, por lo que el acceso a Internet lo realizan de formas variadas: algunos tienen Internet en su trabajo, otros se conectan desde casa, desde casa de algún amigo (con una dirección de correo desconocida para mí) o incluso se me han dado varios casos que envían las

prácticas desde *cybercafes*. Como se puede comprender, la pertinente respuesta al *e-mail* por mi parte sobre los errores cometidos, no siempre llega a quien debiera.

A pesar de los inconvenientes indicados no me parecería oportuno prescindir de estas utilidades, ya que si bien es cierto que acarrean algunos problemas y conllevan un considerable gasto de tiempo, muchos alumnos se benefician de este pequeño esfuerzo. Quiero que sirva esta sección no como una crítica sobre uso de estas tecnologías de Internet para la docencia sino todo lo contrario; animo a profesores con asignaturas de contenido parecido a que utilicen estos nuevos recursos pero teniendo presente los inconvenientes que pueden ocasionar.

UTILIZACIÓN DE HOJAS DE CÁLCULO PARA LA ENSEÑANZA DE MODELOS MATEMÁTICOS EN FOTOGRAMETRÍA ANALÍTICA

J. Cardenal, J.E. Caracuel, J. Delgado, M. Torres

Dpto. Ingeniería Cartográfica, Geodésica y Fotogrametría.

Escuela Politécnica Superior. Universidad de Jaén.

c/ Virgen de la Cabeza, 2 –23071 Jaén (España)

e-mail: jcardena@ujaen.es; caracuel@ujaen.es; jdelgado@ujaen.es; mtorres@ujaen.es

RESUMEN: Se presenta una aplicación sobre hojas de cálculo utilizada en la enseñanza de Fotogrametría Analítica en la Universidad de Jaén en las titulaciones de Ingeniería en Geodesia y Cartografía e Ingeniería Técnica Topográfica. Se han seleccionado 2 ejemplos: orientación externa de un par estereoscópico mediante el método de dos pasos y la calibración de una cámara mediante el método de las líneas rectas. Cabe señalar que pueden ser implementados otros modelos basados en la resolución de sistemas de ecuaciones mediante mínimos cuadrados. La utilización de las hojas de cálculo permite a los alumnos visualizar directamente las ecuaciones empleadas y la forma en la que las mismas son actualizadas en el proceso iterativo durante las sucesivas aproximaciones. Las hojas de cálculo actuales permiten una gran capacidad de cálculo que incluyen herramientas de gran utilidad (p.ej., álgebra matricial), dentro de un entorno gráfico de gran comodidad. Las ecuaciones pueden ser escritas mediante un editor de fórmulas y la resolución se facilita gracias al lenguaje de macros de cualquier hoja convencional.

1.- INTRODUCCIÓN

Una hoja de cálculo es una tabla compuesta por filas y columnas, en las que cada intersección se denomina celda. Cada celda almacena un valor numérico (dato) o fórmula (lógica/matemática). Los valores iniciales (observaciones u aproximaciones iniciales) constituyen la base de datos inicial, distribuidos en celdas individuales. Los cálculos y formulaciones pueden aplicarse al nivel de celda, fila, columna o bloques. Además, se pueden diseñar plantillas para la resolución de modelos matemáticos mínimos cuadrados (mm.cc.). En este trabajo se presentan algunos ejemplos desarrollados por los autores y empleados en la docencia de Fotogrametría Analítica (Ingeniería en Geodesia y Cartografía e Ingeniería Técnica Topográfica) en la Universidad de Jaén: una plantilla para la orientación externa de pares

estereoscópicos mediante el método de dos pasos y otra para la calibración de cámaras mediante el método de las líneas rectas.

2.- ORIENTACIÓN EXTERNA EN DOS PASOS

La Figura 1 muestra un ejemplo de plantilla para la orientación externa (OE) de un par estereoscópico (aéreo o terrestre) por el método de dos pasos. Esta plantilla está preparada para resolver la orientación del par con seis puntos de Von Gruber y cuatro puntos XYZ de apoyo. Además, permite calcular las coordenadas espaciales de cualquier otro punto medido en el modelo. Tras introducir los datos, se realiza la orientación relativa (mediante la condición de coplanaridad), luego se calculan las coordenadas modelo (mediante la condición de colinealidad) y se finaliza con la orientación absoluta (mediante una transformación 3D). La plantilla incluye macros para el cálculo iterativo de la condición de coplanaridad y la transformación 3D y, también, para establecer las aproximaciones iniciales.

a) Orientación Relativa (OR)

La OR del par se realiza mediante la condición de coplanaridad por el método dependiente (Ghosh, 1987). Este método fija la posición y orientación de la fotografía izquierda y el parámetro b_x (escala) del modelo. Los cinco parámetros libres son los giros de la cámara derecha (expresados como T, N y 6) y las componentes Y y Z de la base. Como observaciones se introducen los puntos de Von Gruber para las fotografías izquierda y derecha. También se incorpora la información sobre la distancia focal de la cámara empleada, la altura de vuelo (o la distancia objeto/cámara) y la base del modelo (b_x) introducida bien como fotobase o como la base aérea o terrestre.

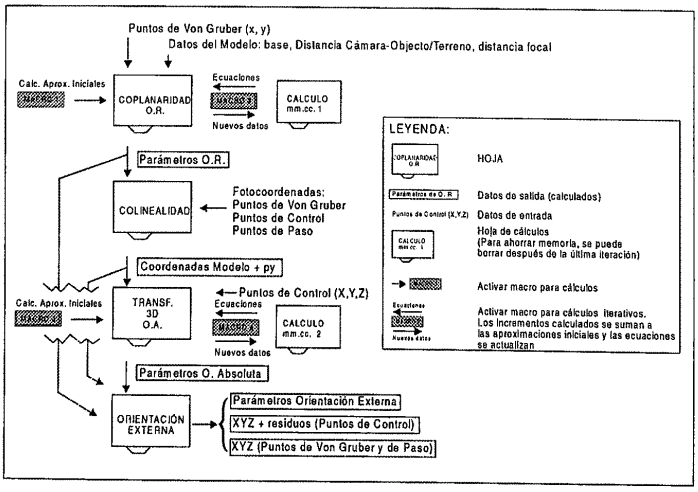


Figura 1. Diagrama de flujo de la plantilla de orientación externa.

Antes de realizar los cálculos se aplica una macro para inicializar la plantilla. Así, se borran datos previos y se asignan las aproximaciones iniciales ($T_2=N_2=6_2=b_Y=b_Z=0$). Las ecuaciones de coplanaridad linealizadas para los 6 puntos de Von Gruber tienen la forma general de:

$$[A]_{(6,24)}(V)_{(24,1)}+[B]_{(6,5)}(\Delta)_{(5,1)}=(-F_0)_{(6,1)} \qquad (1)$$

Siendo [A]=*(F)/*(coordenadas medidas), matriz de coeficientes de residuos; [B]=*(F)/*(parámetros), matriz de coeficientes; ($-F_0$)=F (función evaluada con observaciones y parámetros aproximados); (V) = vector de residuos de fotocoordenadas; $(\square)^T$ = (*T_2, *N_2, *6_2, *b_Y, *b_Z), vector de correcciones a los parámetros. La resolución de las ecuaciones se realiza por mm.cc. (Mikhail, 1976). Las correcciones a los parámetros se suman automáticamente a las aproximaciones iniciales, de forma que las ecuaciones se actualizan para las nuevas aproximaciones. Al mismo tiempo se realizan los cálculos de la varianza a posteriori y la desviación estándar de las variables calculadas. El criterio de convergencia del sistema de ecuaciones lo establece el propio usuario en función de la magnitud de las correcciones o de la varianza del ajuste.

b) Cálculo de las coordenadas modelo

La condición de colinealidad permite calcular las coordenadas modelo de los puntos de Von Gruber, de control y de paso medidos. Se realiza en una hoja diferente, pero ligada a la anterior. La expresión 2 muestra las ecuaciones de colinealidad para las coordenadas modelo (X_{Mi}, Y_{Mi}, Z_{Mi}) de la fotografía izquierda y derecha para un punto i.

$$\begin{vmatrix} X_{Mi} \\ Y_{Mi} \\ Z_{Mi} \end{vmatrix} = \frac{1}{k'} M_R^T \begin{vmatrix} x_i' \\ y_i' \\ -f' \end{vmatrix} + \begin{vmatrix} X_0' \\ Y_0' \\ Z_0' \end{vmatrix}_{OR} = \frac{1}{k''} M_R^T \begin{vmatrix} x_i'' \\ y_i'' \\ -f'' \end{vmatrix} + \begin{vmatrix} X_0'' \\ Y_0'' \\ Z_0'' \end{vmatrix}_{OR} \qquad (2)$$

A partir de los factores de escala (k' y k"), los parámetros de OR y las fotocoordenadas izquierda y derecha se calculan las coordenadas modelo, junto con el paralaje residual (py) a escala modelo y fotografía. Para un punto i, cuyas fotocoordenadas ocupan cuatro celdas (de la misma fila), se necesitan seis celdas para calcular k', k", X_{Mi}, Y_{Mi}, Z_{Mi}, $py_{(MODELO)}$ y $py_{(FOTO)}$. Sólo se requieren, pues, seis fórmulas (escritas previamente para el primer punto) y las referencias de los parámetros de la OR obtenidos de la primera hoja. Estas fórmulas se copian para el resto de los puntos.

c) Orientación Absoluta (OA)

La estructura de la hoja de la OA es similar a la de OR. Requiere dos macros para la obtención de las aproximaciones iniciales y la resolución mediante mm.cc. de la transformación tridimensional. Los valores de entrada son las coordenadas modelo (previamente calculadas) y las coordenadas terreno de cuatro puntos de apoyo XYZ. Estos puntos dan lugar a 12 ecuaciones con 7 incógnitas (factor de escala, tres giros y tres traslaciones). El modelo de la transformación rígida tridimensional es el mismo de Wolf (1983) y la expresión linealizada (sistema de ecuaciones de observaciones indirectas; Mikhail, 1976) para cuatro puntos:

$$[B]_{(12,7)}(\Delta)_{(7,1)} = (L)_{(12,1)} + (V)_{(12,1)} \qquad (3)$$

donde [B]=*(F)/*(parámetros), es la matriz de coeficientes; (L)=Vector de discrepancia (coordenadas medidas-calculadas); (V)=Vector de residuos; $(\square)^T$=(*s, *Σ, *M, *5, *T_X, *T_Y, *T_Z), vector de correcciones a los parámetros.

La aproximaciones iniciales se calculan activando una macro (la cual elimina también cálculos previos). El factor de escala inicial se estima a partir de la relación de distancias entre los dos primeros puntos en terreno/objeto y modelo, y los valores iniciales para los ángulos Σ y M se asignan a 0. El valor inicial de 5 (5_0) es estimado a partir de la diferencia de los arctg de los dos primeros puntos en espacio objeto (Xt, Yt) y modelo (Xm, Ym):

$$5_0 = \arctan((Yt'-Yt'')/(Xt'-Xt''))-\arctan((Ym'-Ym'')/(Xm'-Xm'')) \qquad (4)$$

Los tres vectores de traslación (T_X, T_Y, T_Z) se estiman a partir de las diferencias de los valores medios de las coordenadas X, Y and Z de todos los puntos en los espacios objeto y modelo.. A continuación una segunda macro resuelve el sistema mediante mm.cc. Los cálculos se realizan en una hoja aparte y las correcciones de cada iteración son sumadas a los parámetros iniciales. Los nuevos parámetros son copiados en la hoja activa en la posición apropiada de forma que las ecuaciones se actualizan nuevamente. Se procede así hasta terminar el proceso de orientación absoluta del modelo.

d) Orientación Externa (OE)

En la última hoja se resumen los datos más importantes de las hojas anteriores (parámetros de OR y OA y coordenadas modelo) y se calculan los parámetros de OE (posición y orientación de ambos fotogramas), así como las coordenadas en el espacio objeto para todos los puntos medidos en el modelo estereoscópico (figura 1).

3.- CALIBRACIÓN DE CÁMARAS POR EL MÉTODO DE LAS LÍNEAS RECTAS

El método de calibración de las líneas rectas ("plumb line calibration"; Brown, 1971) establece que una línea recta en el espacio objeto es captada por una lente ideal (sin distorsión) como otra línea recta en el espacio imagen. Sin embargo, las lentes reales están afectadas de distorsión y las desviaciones que se aprecien en la fotografía respecto a una línea recta se consideran debidas a distorsión (si otros errores potenciales son debidamente corregidos o no considerados). Este es un método que permite calibrar sólo los parámetros de distorsión de la cámara. Así, una línea recta en un sistema 2D puede ser expresada de acuerdo con la ecuación 5 (figura 2):

$$x' sen(\Theta_i) + y' cos(\Theta_i) = \rho_i \qquad (5)$$

Todo punto j perteneciente a una línea i (x'_{ij}, y'_{ij}) debe verificar la ecuación 1. Pero, si los puntos reales medidos sobre las líneas en las fotografías (x_{ij}, y_{ij}) no se ajustan a la expresión (por efecto de la distorsión radial simétrica y descentrado), aquellos pueden ser corregidos mediante los coeficientes de distorsión (ecuaciones 6a y 6b; ver Ghosh, 1987):

$$\left.\begin{aligned} x_{ij}' &= x_{ij}(1 - K_1 r^2) + \Delta x \\ y_{ij}' &= y_{ij}(1 - K_1 r^2) + \Delta y \end{aligned}\right\} \quad (6a) \qquad \left.\begin{aligned} \Delta x &= P_1(r^2 + 2x_{ij}^2) + 2P_2 x_{ij} y_{ij} \\ \Delta y &= P_2(r^2 + 2y_{ij}^2) + 2P_1 x_{ij} y_{ij} \end{aligned}\right\} \quad (6b)$$

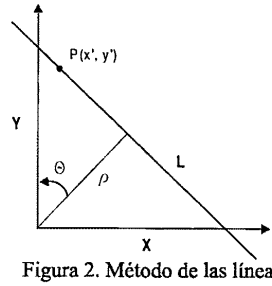


Figura 2. Método de las líneas rectas

Donde K_1 es el coeficiente de distorsión radial simétrica y P_1 y P_2 son los coeficientes de distorsión de descentrado (Ghosh, 1987). El modelo de distorsión empleado no incluye coeficientes de grado superior a K_1 (no suelen ser significativos en cámaras no métricas de pequeño formato) o las coordenadas del punto principal de referencia (incluido en la distancia radial, r) ya que, de acuerdo con Brown (1971), es difícil de registrar con este método. Si las ecuaciones 6a y 6b son sustituidas en la expresión 5, se obtiene la ecuación de observación para cada punto j en una línea i de la forma:

$$f(x_{ij}, y_{ij}, K_1, P_1, P_2, \Theta_i, \rho_i) = 0 \qquad (7)$$

Las incógnitas son los coeficientes de distorsión (K_1, P_1 y P_2, comunes a todas las ecuaciones) y los dos parámetros geométricos de cada línea (Θ_i y ρ_i, comunes a todas las ecuaciones de los puntos de una misma línea). Si se consideran más ecuaciones de observación que incógnitas el cálculo se realizará por ajuste mínimo cuadrático de las ecuaciones (previamente linealizadas por desarrollo en series de Taylor).

Este método requiere fotografiar un número suficiente de líneas (en la misma o en distintas imágenes). Se deben consideran al menos dos familias de líneas que mantengan con fuerte ángulo entre sí. Así, se pueden hacer dos fotografías en laboratorio para generar las 2 familias de líneas; la primera con las líneas verticales o subverticales (pueden ser hilos o cordeles elásticos lastrados, cuyos extremos se sumergen en aceite para evitar vibraciones); y la segunda con la cámara rotada unos 90° para captar las mismas líneas en posición "horizontal". El siguiente paso, consiste en medir fotocoordenadas de puntos sobre las líneas en las fotografías .

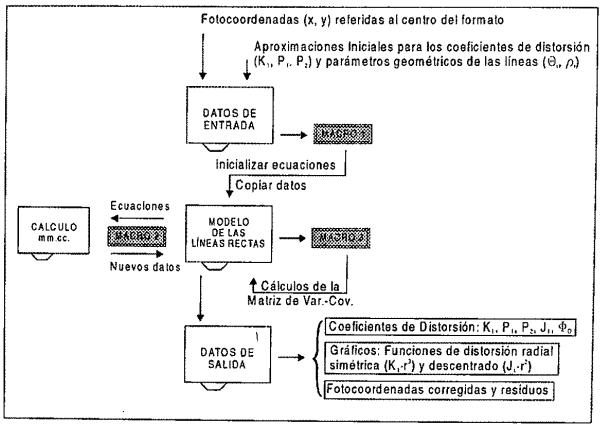


Figura 3. Diagrama de flujo de la plantilla del método de las líneas rectas. La leyenda es la misma de la figura 1.

La plantilla para el método de las líneas rectas se divide en 3 hojas (figura 3). La primera (entrada de datos) permite la introducción de las fotocoordenadas de 50 puntos medidos en 10 líneas (5 puntos por línea). También se introducen las aproximaciones iniciales de los coeficientes de distorsión y los parámetros geométricos de las líneas. Generalmente basta con aproximar $K_1=P_1=P_2=0$, $\Theta_i=90°$ y 0° para las líneas verticales y horizontales respectivamente, y para ρ_i se asigna el valor medio de abcisas (líneas verticales) u ordenadas (líneas horizontales). Activando una macro se borran los datos previos, se copian las aproximaciones iniciales y se actualiza el sistema de 50 ecuaciones y 23 incógnitas (3 parámetros internos y 20 geométricos: 2 por línea). El sistema de ecuaciones linealizadas se resuelve por mm.cc. tras varias iteraciones (3 ó 4 suelen ser suficientes). También, dado el entorno de trabajo, se representan gráficamente las funciones de distorsión radial simétrica ($dr_S = K_1r^3$) y de descentrado ($d_D = J_1r^2$, donde $J_1=[P_1^2 + P_2^2]^{0.5}$; Ghosh, 1987).

Una macro resuelve las ecuaciones por mm.cc. y los cálculos de los residuos y desviaciones standard de los parámetros calculados. La última hoja de cálculo (salida de datos, Figura 3) incluye la medida de puntos, corrección de fotocoordenadas y valores de distorsión de simetría radial y tangencial para cada punto.

4.- CONCLUSIONES.

Este trabajo ha mostrado como se pueden adaptar algunos de los modelos matemáticos habituales en fotogrametría en un entorno de hoja de cálculo. Ello conlleva varias ventajas dado que el alumno puede ver físicamente las ecuaciones y como funcionan los procesos de cálculo iterativo. Además la programación en lenguaje macro es simple y fácil (incluso para usuarios no habituados a este tipo de programas), por lo que los alumnos pueden desarrollar sus propias

plantillas antes de ser introducidos en lenguajes de programación avanzados (Fortran o C, p. ej.).

Si bien sólo se han mostrado dos ejemplos (método de dos pasos y calibración por líneas rectas) se pueden adaptar otros modelos matemáticos: orientación en un paso, cálculo de la transformación lineal directa (DLT), calibración "in situ" de una cámara por colinealidad o el método de Church, etc. (Cardenal et al., 2000). Todos estos modelos han sido empleados en las clases prácticas (y en algunos casos desarrollados por los propios alumnos) en las titulaciones de Ingeniería en Geodesia y Cartografía e Ingeniería Técnica Topográfica de la Universidad de Jaén. Las plantillas usadas en este trabajo se han preparado con Corel Quattro Pro (versiones 5.0 y 8.0; estensiones *.wb1 y *.wb3), si bien pueden ser fácilmente adaptables a otras hojas de cálculo.

5.- REFERENCIAS.

- Brown, D.C., 1971. Close-Range Camera Calibration. Photogrammetric Engineering, 37(8): 855-866
- Cardenal, J., Delgado-García, J, Torres, M. (2000). On the use of spreadsheet for teaching analytical photogrammetry. IAPRS, Vol. 33, Part 5, Amsterdam (NL).
- - Ghosh, S., 1987. Analytical Photogrammetry. Pergamon Press, NY, USA. 308 p.
- Mikhail, E.M., 1976. Observations and Least Squares. Harper & Row Pub. NY, USA. 497 p.
- Wolf, P.R., 1983. Elements of Photogrammetry 2nd ed. MacGraw Hill Pub. 628 p.

MEJORA DE LA ENSEÑANZA DE ALGORITMOS SIG EN LA E.T.S.I. GEODÉSICA, CARTOGRÁFICA Y TOPOGRÁFICA DE LA UNIVERSIDAD POLITÉCNICA DE VALENCIA.

Jesús Palomar Vázquez[1]

[1]*Departamento de Ingeniería Cartográfica, Geodesia y Fotogrametría de la Universidad Politécnica de Valencia*
e-mail: jpalomav@upvnet.upv.es

RESUMEN: En el presente artículo se muestra una forma de trabajo que facilita la docencia de técnicas y algoritmos que se engloban dentro de la informática gráfica y en concreto dentro del área de los Sistemas de Información Geográfica (SIG). Al final del mismo, se analizan los resultados obtenidos sobre la metodología empleada teniendo en cuenta el contexto educativo en el que se desarrolla la experiencia.

1.- INTRODUCCIÓN.

Dentro del campo del manejo de la información gráfica en general y geográfica en particular, nos estamos acostumbrando al uso de herramientas SIG que nos proporcionan una rápida y fácil solución para su gestión. La problemática surge cuando "atamos" a nuestros titulados a un determinado *software* comercial, lo que provoca su dependencia del mismo. Si esto se evitara no solo se haría independiente al estudiante, sino que le posibilitaría una mayor comprensión y aprendizaje, una comparación más ajustada entre diferentes productos del mercado e incluso la confección de sus propias herramientas.

Este hecho, unido a que no existen unos textos bibliográficos que engloben todos o al menos una gran parte de los algoritmos básicos generales de estos sistemas, hace necesarios trabajos de este tipo donde se recopilen en una base de datos los conocimientos y experiencias que los alumnos de año en año van depositando para que sean aprovechados por otros alumnos. De esta forma, la posibilidad de visualizar el funcionamiento de un algoritmo, variar sus parámetros, acceder a su bibliografía, etc., redunda en la mejora considerable del aprendizaje.

2.- CONTEXTO EDUCATIVO

El conjunto de esta experiencia se desarrolla dentro del marco de la asignatura denominada Técnicas Gráficas, asignatura adscrita al plan de estudios de la titulación de Ingeniería en Geodesia y Cartografía como optativa de 5° curso (último) con una carga docente de 4.5 créditos.

Teniendo en cuenta las particularidades propias de esta titulación, los objetivos generales de la asignatura van enfocados a:

- Dar a conocer las bases de la representación de gráficos por ordenador.
- Proporcionar conocimientos sobre diversas estructuras de datos.
- Conocer las etapas necesarias para enfrentar al alumno ante un problema geométrico y su resolución mediante la aplicación de un algoritmo.
- Conocer el funcionamiento de los algoritmos más usuales usados en los SIG, tanto sobre información vectorial como raster.

Así mismo, los puntos sobre los que se centra el temario tres principalmente:

- Visión introductoria del dibujo de gráficos en ordenador, donde se estudia la interface gráfica, las técnicas de representación de primitivas simples, rellenos, recortes, etc. y las transformaciones geométricas.
- Estudio de los principales algoritmos usados en SIG, con una previa referencia a las estructuras de datos espaciales.
- Un bloque práctico en el que el alumno estudia, analiza, e implementa la resolución de un determinado problema geométrico que él mismo elige.

3.- METODOLOGÍA DE ENSEÑANZA

El objetivo principal de la asignatura es que el alumno aprenda a conocer los fundamentos de las herramientas que utiliza constantemente dentro del mundo de los SIG. Este objetivo se materializa en un sistema de aprendizaje interactivo en el cual el alumno puede probar los algoritmos existentes, hacer modificaciones sobre ellos e incluir nuevas implementaciones en cualquier lenguaje que considere adecuado, todo ello desde el punto de vista de que sea el propio alumno el que aporte experiencia para sus compañeros que formarán parte de la asignatura en años sucesivos, con lo cual se asegura una retroalimentación interna (durante el curso) y externa (entre cursos sucesivos).

La idea global del diseño del sistema de aprendizaje puede verse en la figura 1 donde aparece el mapa mental que lo describe y del que se desprenden las siguientes características, agrupadas en cuatro grandes ramas:

a) Programas

El alumno parte de algoritmos descritos durante las clases teóricas e implementados sobre un SIG, de forma que puede estudiar el análisis del problema, visualizar el algoritmo, revisar su diagrama de flujo y mirar el código implementado y modificarlo.

b) Práctica

El alumno puede poner en práctica de forma inmediata sus conocimientos sobre los algoritmos vistos haciendo pruebas sobre datos existentes, creando sus propios datos, modificando los parámetros de entrada de los algoritmos y añadiendo comentarios sobre los resultados obtenidos para su posterior uso por parte de otro compañero.

c) Ayuda complementaria

En todo momento se dispone de una ayuda adicional a la tradicional clase teórica, donde el alumno puede encontrar información sobre aspectos determinados del algoritmo o problema de estudio.

En este apartado, el propio estudiante podrá añadir a la base de datos más bibliografía, referencia a artículos, proyectos final de carrera, páginas *web*, etc., así como también podrá tener acceso a demostraciones grabadas del funcionamiento de los diferentes algoritmos y a ejemplos del uso de dicho algoritmo para la solución de problemas reales, teniendo la posibilidad también de añadir nuevas aportaciones para estos dos últimos conceptos.

d) Feedback

Con el uso de esta herramienta el alumno puede probar y ver los resultados, así como modificar constantemente los parámetros de entrada, todo lo cual adentra al estudiante en procesos de autoaprendizaje, con el incentivo adicional motivador de verse implicado personalmente en un proyecto vivo que cada año se incrementa y sirve para ayudar a otros compañeros.

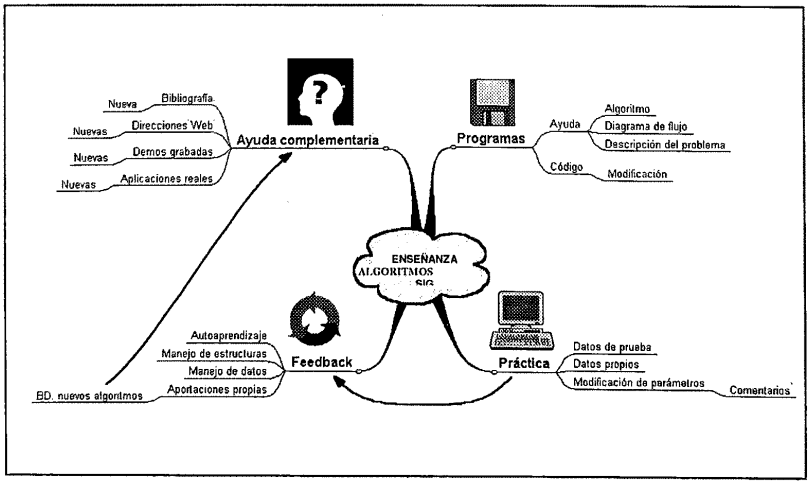


Figura 1. Mapa mental del proyecto

En cuanto a los algoritmos analizados, la asignatura parte de los más básicos, como pueden ser el cálculo de distancias y acimutes, la creación de intersecciones o el negativo de una imagen y llega hasta los que pueden entrañar un grado dificultad media, como puede ser el cálculo de caminos óptimos, la creación de polígonos o el análisis de visibilidad de un modelo digital del terreno. En años posteriores, conforme la velocidad de aprendizaje mejore debido a la gran cantidad de ejemplos e información aportados por sus compañeros, el alumno estará en condiciones de abordar problemas más complicados (figura 2) y la base de datos seguirá alimentándose de problemas aportados por el alumno y también derivados de otras materias (geografía humana, reparcelaciones catastrales, gestión medioambiental, etc.).

Otra idea respecto a los algoritmos estudiados es que se trata de hacer especial hincapié en la importancia de la organización inicial de los datos a la hora de aumentar la sencillez del algoritmo.

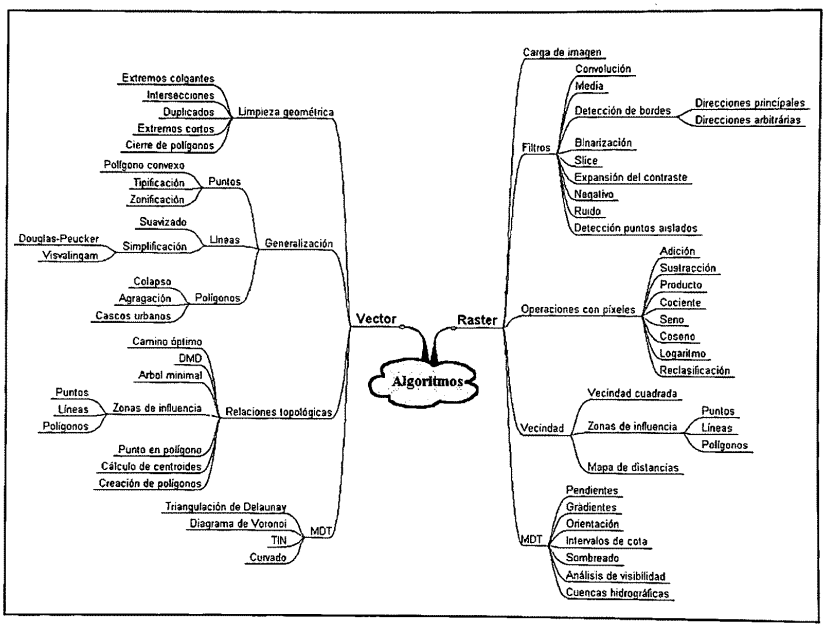


Figura 2. Mapa mental de los algoritmos

4.- DESARROLLO DE LA BASE DE DATOS

La comprensión de estos algoritmos hace que el alumno pueda después implementarlos en cualquier lenguaje de programación, lo que lo hace independiente de los sistemas con los que pueda trabajar profesionalmente y mejora la comunicación con expertos informáticos a la hora de generar aplicaciones. Una parte importante de esta comprensión es la visualización del funcionamiento de un algoritmo, por lo que se ha optado por hacerlo sobre un entorno SIG conocido por nuestros alumnos para que la operatividad en la práctica sea inmediata. En este caso en concreto se ha utilizado el *software* ArcView® 3.2 sobre plataforma Microsoft® NT™. La implementación de los diferentes algoritmos ha sido realizada principalmente en Avenue™.

lenguaje propietario de este SIG, aunque el alumno tiene libertad para escoger otro lenguaje (aportando una demo grabada de los resultados a la base de datos), siendo esta una parte menos importante de la evaluación del trabajo del alumno.

El resultado de todo esto se plasma en una serie de menús y herramientas integradas dentro de este SIG, desde los que tenemos acceso a la biblioteca de algoritmos, la base de datos bibliográfica, enlaces a páginas *web* y la carga de los datos de prueba (figura 3) y sobre los cuales el alumno volcará el resultado de su trabajo al final de la asignatura.

Los datos vectoriales de prueba han sido adquiridos a partir de la cartografía 1:10.000 del Instituto Cartográfico Valenciano (ICV), mientras que los datos raster han sido obtenidos a partir de imágenes escaneadas. Todos los datos vienen acompañados de sus metadatos correspondientes.

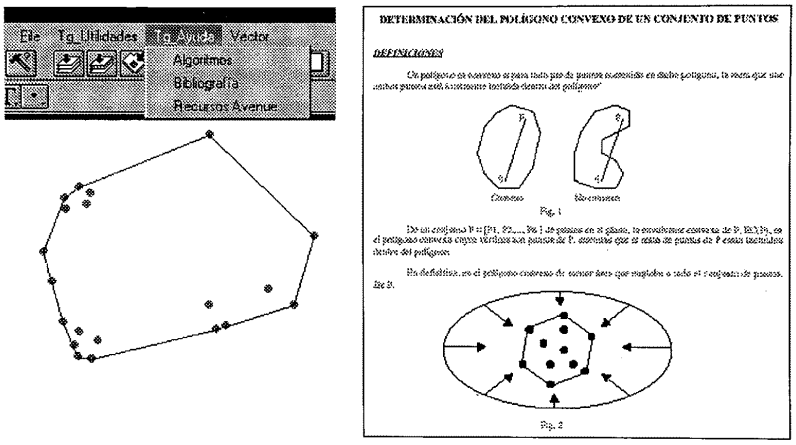


Figura 3. Polígono convexo. Funcionamiento y ayuda

5.- RESULTADOS Y CONCLUSIONES

Para determinar la bondad del método de mejora de la enseñanza se ha requerido por parte del alumno que rellene una encuesta cualitativa, para conocer mejor la realidad del proceso de aprendizaje, con una serie de preguntas que van orientadas a conocer el grado de satisfacción del estudiante por la metodología empleada, motivación inicial, nivel y calidad de conocimientos adquiridos, aprovechamiento de los mismos, opinión sobre la mejora del sistema, etc. y cuyos resultados más importantes se ven en la tabla 1, donde aparece la categorización de las preguntas.

El cuestionario y la tabla de resultados se muestran a continuación:

1. ¿Qué inquietudes tenías sobre el tema antes de cursar la asignatura?

2. ¿Cómo considerarías tu conocimiento inicial sobre la materia?

3. ¿Cuáles eran tus expectativas con respecto a la asignatura?

4. ¿Qué conocimientos previos crees que deberían tenerse para abordar la asignatura?

5. ¿Cómo valoras el sistema metodológico de enseñanza?

6. ¿Qué es lo que más valoras de esta metodología?

7. ¿Qué es lo que cambiarías o mejorarías en ella?

8. ¿En qué te ayudan los conocimientos adquiridos para otras materias?

9. ¿Se justifica la asignatura?

10. ¿Qué número de alumnos crees que sería el adecuado?

11. ¿Qué porcentaje darías a la teoría y a la práctica?

12. ¿Qué aspectos valorarías para mejorar la enseñanza de temas informáticos en titulaciones donde esta componente así como la preparación previa para abordarla no son muy altos?

Bloque	Respuestas
Motivación	1. (62.5 %) profundizar funcionamiento de un SIG y aprender un lenguaje 2. (70%) muy pocos 3. (75%) aprender algoritmos y un lenguaje nuevo 4. (37.5%) aprender algoritmos; (50%) aprender un lenguaje
Satisfacción	5. (70%) adecuado; (30%) creación de la base de datos de algoritmos 6. (75%) la teoría se ve adecuadamente reforzada con la práctica (continuidad) 7. (75%) aumento de prácticas
Utilidad	8. (50%) aprender un lenguaje y desarrollo de herramientas específicas; (25%) el pensamiento lógico ayuda para otras materias 9. (75%) totalmente de acuerdo
Estructura	10. (62.5%) 10 a 15 11. (50%) 30 % teoría y 70% práctica
Mejoras generales	12. (25%) acercar la informática como herramienta habitual desde el primer año; (25%) enfoque práctico y orientado hacia el mundo real; (25%) adaptación de los contenidos a las particularidades de la titulación

Tabla1. Resultados de la encuesta

A partir de los resultados obtenidos y una vez filtradas y aglutinadas todas las respuestas se obtienen las siguientes conclusiones:

- Se valora la mejor comprensión de las funciones generales del funcionamiento de un SIG.

- Se hace hincapié en la aplicabilidad de la asignatura al entorno laboral.

- Agradecen el pensamiento lógico y estructurado que se desprende del análisis de problemas y su aplicación a otras materias.

- Consideran muy adecuada la formación de una base de datos de algoritmos como ayuda a la enseñanza.

- Considerarían positivo un aumento de la carga práctica.

- Sería muy positiva una contextualización de los contenidos.

Como conclusión final, podemos decir que la docencia de una asignatura con contenidos que a priori pueden parecer áridos, ha causado buena impresión en general al alumnado, haciéndola más dinámica, cercana al mundo real y adaptada al esfuerzo que cada estudiante quiera desarrollar.

6.- REFERENCIAS

- De Verg, M. (1997): Computational Geometry. Algorithms and Applications. Ed. Springer. 365 pag.
- Domingo Ajenjo, A. (1994): Tratamiento Digital de Imágenes. Ed. Anaya. 170 pag.
- Foley; Van Dam; Feiner; Hughes; Phillips (1991): Introducción a la Graficación por Ordenador. Ed. Addison-Wesley Iberoamericana. 637 pag.
- Rodríguez Gómez, G.; Gil Flores, J.; García Jiménez, E. (1996): Metodología de la investigación cualitativa. Ed. Aljibe. 378 pag.
- Sellarés i Chiva, J.A. (1987): Fundamentos de los Gráficos por Ordenador. Ed. Edunsa. 314 pag.
- Publicaciones sobre los talleres del II Programa de Formación Inicial Pedagógica para el Profesorado Universitario (FIPPU). Instituto de Ciencias de la Educación (ICE). Universidad Politécnica de Valencia. 1999 / 2000

GENERACION DE MODELOS DIGITALES DE TERRENO A PARTIR DE MAPAS DE ELEVACIÓN

Autor: Jose Poveda[1], Javier Sevilla[2], Michael Gould[3]

[1]*Departament d'Informàtica. Universitat de València*
e-mail: ***jose.f.poveda@uv.es***

[2]*Instituto de Robótica. Universitat de València*
e-mail: ***jsevilla@glup.irobot.uv.es***

[3]*Departament d'Informàtica. Universitat Jaume I, Catellón*
e-mail: ***mgould@inf.uji.es***

RESUMEN: Uno de los problemas fundamentales con el que nos enfrentamos, ya sea tanto en la enseñanza de sistemas de información geográfica como en el desarrollo de los mismos, es la carencia de datos reales con los que poder testear tales sistemas. Obviamente tales datos existen, pero su coste los hace inaccesibles para la mayoría.

En este sentido, en el presente artículo se desarrolla un nuevo método para la generación de Modelos Digitales de Terreno (MDT[3]), a partir de mapas de elevación. El objetivo perseguido es el de generar un modelo digital de terreno realista de forma sintética y por lo tanto gratuita, evitando obtener los errores típicos de cortes abruptos de los métodos de interpolación clásicos, al mismo tiempo que maximizar en general la suavidad del DTM generado.

El método definido para la generación del DTM está basado en la teoría de campos electroestáticos de la física clásica.

1.- INTRODUCCIÓN.

Una de las mayores dificultades que se encuentra en el estudio y desarrollo de técnicas y métodos en sistemas de información geográfica es la dificultad de disponer de datos reales de forma gratuita, o en su defecto de la posibilidad de crear manualmente datos similares a los reales. Esta dificultad es debida al gran volumen de información que un DTM implica. En este sentido existen métodos de síntesis de modelos digitales de terreno, basados en algoritmos aleatorios, fractales, recursivos etc., que producen fantásticos terrenos pero más cercanos a lo exótico que al modelo real.

[3] En inglés Digital Terrain Models (DTM)

Por otro lado la generación de modelos digitales de terreno a partir de líneas de contorno con algoritmos de triangulación[4], o con métodos similares[5] producen errores o efectos no deseados.

2.- MODELO DEL MAPA DE ELEVACIONES

Nuestro propósito es el de definir un proceso que nos permita diseñar una superficie con características similares a las características que presentaría una superficie que representara un terreno real. En este sentido, la idea que pretendemos desarrollar está basada en el hecho de que una posible representación de un terreno real puede venir dada por un mapa de líneas de elevación. Obviamente este tipo de representación pierde gran parte de la información de la superficie real a la que representa, pero nos aporta gran simplicidad y una idea muy clara del perfil del terreno que representamos con tan sólo unas cuantas polilíneas en un espacio de dos dimensiones.

La idea que nos surge estaría pues orientada en este sentido. Teniendo en cuenta que la representación de un mapa de isolíneas es una forma simple de representar un terreno y que además posee la mínima información relevante para la reconstrucción del mismo. Nos planteamos pues diseñar terrenos a partir de un simple diseño efectuado a mano con un editor gráfico o simplemente con un lápiz y un papel que posteriormente digitalizaremos para generar el terreno 3D, a partir del mapa de elevaciones 2D.

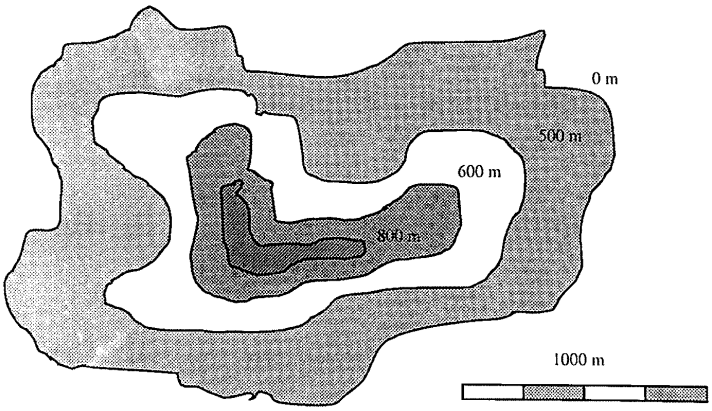


Fig. 1 – Diseño de terreno a partir de su mapa de elevaciones

3.- DISEÑO DEL TERRENO Y ADQUISICIÓN DE DATOS

Diseñado el terreno a partir del mapa de elevaciones, pasamos a identificar y vectorizar todas y cada una de las líneas de elevación. Para ello, a partir de la imagen digitalizada identificamos el pixel de mayor altura cuyo valor sea nulo y seleccionando un radio r, procedemos a muestrear la isolínea a la que pertenece dicho punto, tal como muestra la figura 2, hasta volver al punto inicial. Después de dicho proceso habremos obtenido una secuencia de puntos pertenecientes a la isolínea más externa con un radio de muestreo r. Seguidamente, recorremos nuevamente la

[4] TYDAC SPANS:contouring, o ESRI Arc/Info:TIN module
[5] GRASS: spline interpolation

secuencia de puntos obtenidos pertenecientes a la isolínea con la misma máscara de radio r y centrada en cada uno de los puntos pertenecientes a la línea de elevación obtenidos anteriormente, asignado a cada punto interior a la máscara el valor 255.

Una vez efectuado el proceso anteriormente descrito, seguimos muestreando la imagen por filas hasta obtener un nuevo pixel de valor nulo, en cuyo caso pasaríamos a muestrear la siguiente línea de elevación, o alcanzar el vértice inferior izquierdo de la imagen, en cuyo momento habremos obtenido todas las secuencias de pixeles correspondientes a las isolíneas que conforman el mapa de elevaciones.

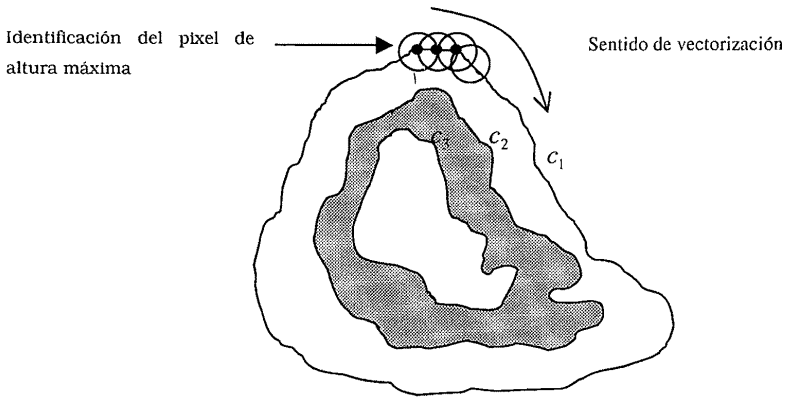


Fig. 2 – Proceso de vectorización de las curvas de elevación

Obtenidas las secuencias $m_1, m_2, ..., m_n$ resultado de la discretización de las curvas $c_1, c_2, ..., c_n$ correspondientes a las isolíneas que forman el mapa de elevaciones, es decir:

$$m_i = \left\{ p_j = (x_j, y_j) \, tal \; que \, p_j \in c_i \wedge \left| p_{k+1} - p_k \right| = r \right\} \qquad (3.1)$$

donde r es el radio de la máscara utilizada para discretizar las curvas $c_1, c_2, ..., c_n$ del mapa de elevaciones.
Ahora, a partir de las sucesiones de puntos $\{m_p\}$ procedemos a construir las siguientes sucesiones puntos

$$m_k^{ij} = \left\{ p_k = (x_k, y_k) \, tal \; que \, p_k \in c_i \vee p_k \in c_j \right\} con \; i < j \qquad (3.2)$$

las cuales utilizaremos para el cálculo del campo electroestático en el área interior a la curva de elevación c_i y exterior a la isolínea c_j y que será la base de nuestro método de interpolación.

4.- ECUACIONES DE CAMPO

Determinadas la sucesiones de puntos existentes en nuestro mapa de elevaciones, el siguiente objetivo que nos planteamos es el del cálculo del campo electroestático en los puntos del dominio del DTM, definido como sigue

$$D = \{p_{n,m} = (x_n, y_m) = (hn, hm) \ \ donde\ m,n \in Z,\ h \in R\} \tag{4.1}$$

y que podemos particionar en conjuntos disjuntos como sigue

$$D = \sum P^{ij} \ \ con \ \ i < j \tag{4.2}$$

y donde los conjuntos P^{ij} están definidos como sigue

$$P^{ij} = \{p_{n,m} \in D \ \ tal\ que\ p_{n,m} \in \mathcal{C}_i \wedge p_{n,m} \notin \mathcal{C}_j \ donde\ \overline{C}_k = c_k\} \tag{4.3}$$

el parámetro h, nos determina la densidad de puntos pertenecientes a D y en particular a cada uno de los conjuntos P^{ij} en los que hemos particionado el dominio. Seguidamente pasamos a calcular el campo para cada P^{ij} suponiendo que la única contribución a este viene dada por las sucesiones de puntos del espacio m_k^{ij} que identificamos en una primera aproximación como distribuciones discretas de k cargas localizadas espacialmente en los puntos determinados por la propia sucesión m_k^{ij} y cuya polaridad vendría dada por la siguiente ecuación:

$$q_k^{ij} = \begin{cases} +z_i \ \ si\ m_k^{ij} \in m_i \\ -z_j \ \ si\ m_k^{ij} \in m_j \end{cases} \tag{4.4}$$

Teniendo pues, en cuenta que en un campo electroestático el campo en el interior de los conductores necesariamente ha de ser cero, si estos fueran fuente o sumidero de líneas de campo, necesariamente estas tendrían que entrar o salir perpendicularmente de ellos. Es por tanto que una segunda mejor aproximación vendría dada por una distribución filiforme continua de carga a lo largo de las isolíneas del mapa de elevaciones. En principio para el cálculo del campo creado por una distribución filiforme de carga partiríamos de la ecuación 4.5,

$$\overline{E}(r) = \int_l \frac{\lambda(\overline{r}')\ (\overline{r} - \overline{r}')}{4\pi\varepsilon\ |\overline{r} - \overline{r}'|} dl \tag{4.5}$$

que adaptada para su evaluación de forma discreta, ya que en definitiva el cálculo que podemos hacer en el ordenador es discreto, ver ec. 4.6, nos quedaría que en definitiva como si consideráramos la contribución al campo de un conjunto discreto de cargas, pero haciendo tender el radio de muestreo a cero, cosa que por otra parte ya intuíamos .

$$\overline{E}(r) = \sum_{k=1}^{n} \frac{dq(\overline{r}_k')\ (\overline{r} - \overline{r}_k')}{4\pi\varepsilon\ |\overline{r} - \overline{r}_k'|} \tag{4.6}$$

Lo que nos interesa pues para hacer una aproximación mejorada con un radio de muestreo pequeño pero no diferencial, es calcular la contribución al campo electroestático que crearía un segmento no diferencial cargado homogéneamente, ver fig. 3, donde sus extremos vendrían determinados por puntos consecutivos de los puntos obtenidos en el muestreo.

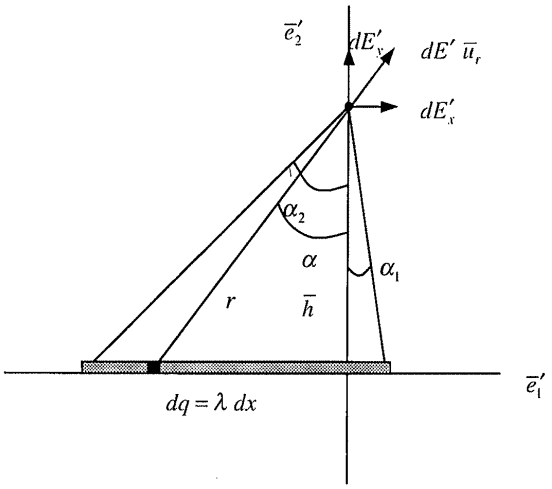


Fig. 3 – Campo creado por un segmento en un punto

El campo total resultante debido a todo el segmento cargado homogéneamente vendrá dado por la integración de la contribución diferencial siguiente

$$d\overline{E}' = \frac{\lambda\, dx}{4\pi\varepsilon\, r^2}\,\overline{u}_r \tag{4.7}$$

que referido a la base nos daría las contribuciones diferenciales siguientes, expresado en función del ángulo con el eje y.

$$dE_x' = \frac{\lambda\ dx}{4\pi\varepsilon\ r^2}\sin\alpha; \quad dE_y' = \frac{\lambda\ dx}{4\pi\varepsilon\ r^2}\cos\alpha; \tag{4.8}$$

sumando todas las contribuciones, nos daría el campo total resultante de un segmento no diferencial, lineal y homogéneamente cargado, ec. 4.9

$$\overline{E}' = \frac{\lambda}{4\pi\varepsilon\, d}\left((\cos\alpha_2 - \cos\alpha_1)\ \overline{e}_1' + (\sin\alpha_1 - \sin\alpha_2)\ \overline{e}_2'\right) \tag{4.9}$$

luego en principio y de forma general tendríamos que para un segmento genérico situado en un punto cualquiera del espacio el campo generado en cualquier otro punto vendría cuantificado por el campo dado en la ec. 4.10 aplicando la transformación de coordenadas dada por la ec. 4.10.

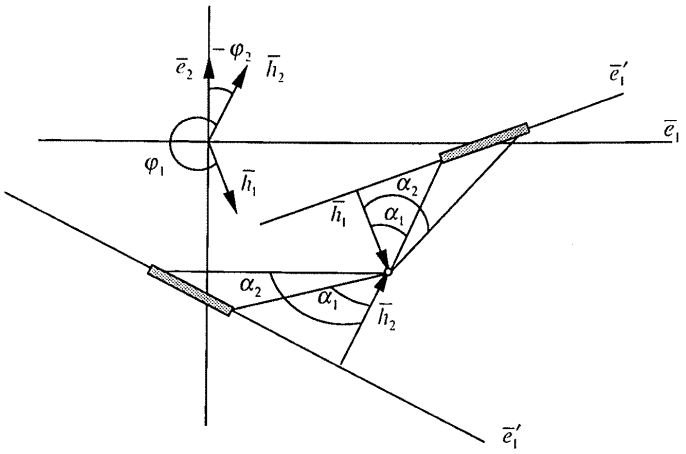


Fig. 4 – Ángulo de rotación en el cambio de base para el cálculo del campo

Tendremos pues que el campo vendrá dado por

$$\begin{bmatrix} E_x \\ E_y \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} E'_x \\ E'_y \end{bmatrix} \tag{4.10}$$

pero teniendo en cuenta que el campo calculado en la ecuación 4.9, ha sido calculado sólo para puntos situados en el subespacio para el cual la componente del punto es positiva, hay que poner especial cuidado en el cálculo del ángulo de rotación de los ejes, ver fig. 4.

5.- INTERPOLACIÓN A TRAVÉS DEL CAMPO

Procedemos ahora que ya tenemos todos los elementos necesarios, a plantear como se interpolaría un nuevo valor en el DTM. Dado un punto del dominio p_{nm} al cual queremos asociar una altura, si este pertenece al conjunto P^{ij}, entonces calcularemos la longitud de la línea de campo que nace en c_i, pasa por p_{nm}, y muere en c_j. Este campo será debido a dos distribución lineales de segmentos cargados uniformemente con una densidad lineal que vendrá dada por

$$\lambda_k = \begin{cases} +z_i \ si \ c_i \\ -z_j \ si \ c_j \end{cases} \tag{5.1}$$

y donde la posición de los extremos de los segmentos viene determinada por la siguiente sucesión

$$m_i = \left\{ (x_j, y_j) \, tal \; que \, p_j \in c_i \wedge \left| p_{k+1} - p_k \right| = r \right\} \tag{5.2}$$

sumando las contribuciones de todos los segmentos de c_i y los de c_j obtenemos el campo total en el punto que queremos interpolar, a partir del campo obtenido definimos el vector desplazamiento.

$$\bar{t} = \Omega \frac{\bar{E}}{|E|} \; donde \; \Omega << r \tag{5.3}$$

que será tangente a la línea de campo que pasa por p_{nm}, ver figura 6.

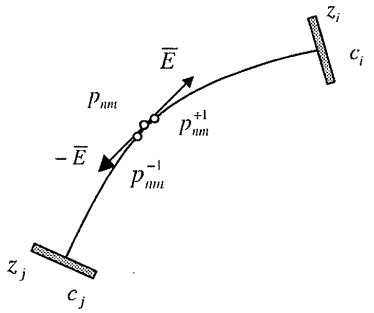


Fig. 5 – Cálculo de la línea de campo que pasa por p_{nm}

A partir del punto p_{nm} y tomando desplazamientos positivo y negativo en el sentido del vector desplazamiento $\bar{t}$, obtenemos dos nuevos puntos a los que volveremos a repetir el mismo proceso pero sólo para obtener un nuevo punto en su dirección de desplazamiento. El proceso terminará para cada una de las direcciones cuando se cumpla respectivamente a cada dirección de desplazamiento que

$$d(p_{nm}^{+}, m_k^{ij}) < r \; ; \; d(p_{nm}^{-}, m_k^{ij}) < r \tag{5.4}$$

sea n^+ el número de iteraciones efectuadas en el sentido positivo y n^- en el sentido negativo, entonces ponderamos la altura en el punto p_{nm} al valor

$$z_{nm} = \frac{z_i \sum_{k=1}^{n^-} \bar{t}_k + z_j \sum_{k=1}^{n^+} \bar{t}_k}{\sum_{k=1}^{n^-} \bar{t}_k + \sum_{k=1}^{n^+} \bar{t}_k} \tag{5.5}$$

6.- RESULTADOS

Seguidamente presentamos los resultados obtenidos de la aplicación del método propuesto basados en el cálculo del campo electroestático generado por distribuciones de carga.

El método ha sido aplicado a la generación de terreno partiendo de un mapa de elevaciones diseñado a tal efecto en 2D, que se muestra en Fig. 6, de tal modo que ha sido posible discretizar sus líneas de contorno.

La polaridad definida para las líneas de elevación ha sido definida positiva en la línea más externa, y alternante hacia el interior. Las alturas definidas en el mapa de elevaciones han sido respectivamente de 0, 70 y 30, empezado a contar desde la más externa. Para todas las pruebas se ha partido de un DTM inicial de altura nula y de una dimensión de 500x360 pixels.

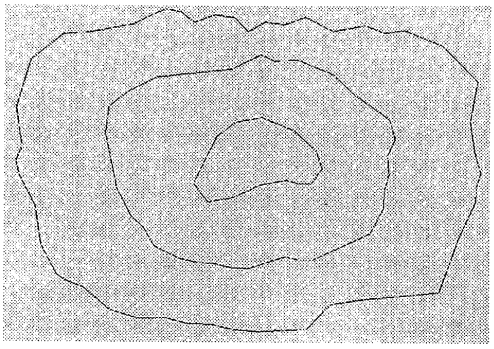


Fig. 6 – Mapa de elevaciones utilizado en las pruebas, con alturas 0, 70, 130.

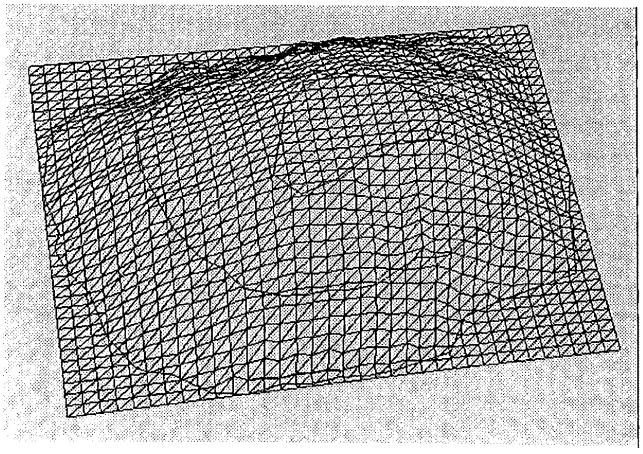


Fig. 7 – Vista de isolíneas y DTM de 500x360, con paso de 14.
Método generador utilizado: Distribución de carga continua.

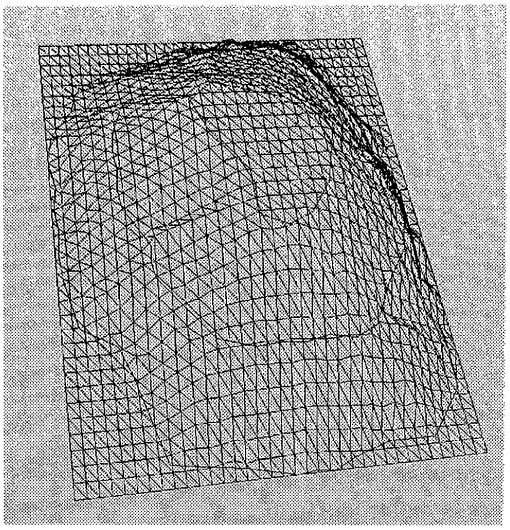


Fig. 8 – Vista de isolíneas y DTM de 500x360, con paso de 14. Método generador utilizado: Distribución de carga continua.

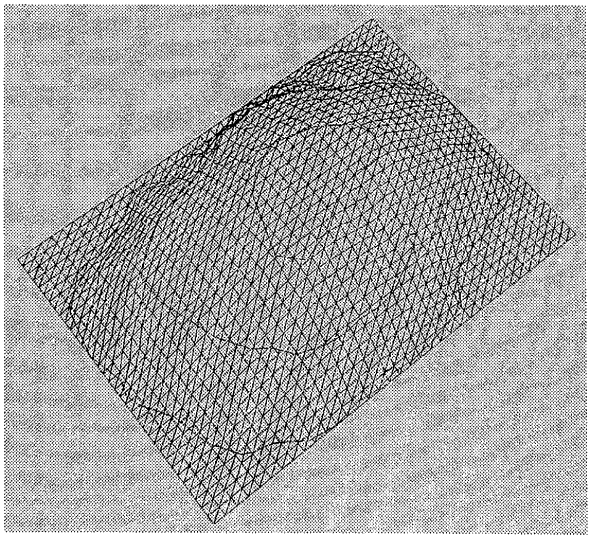


Fig. 9 – Vista de isolíneas y DTM de 500x360, con paso de 14. Método generador utilizado: Distribución de carga continua.

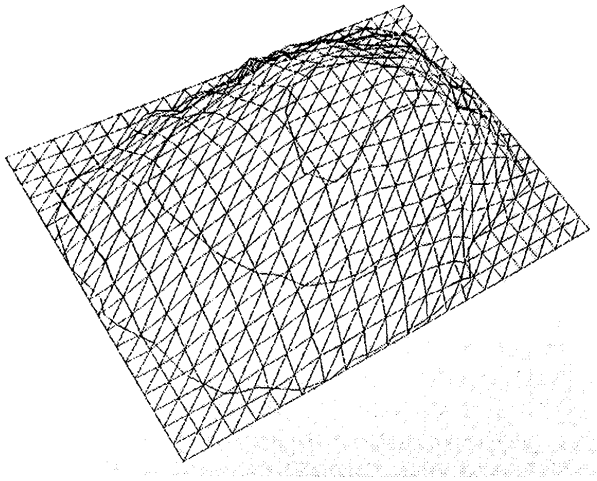


Fig. 10 – Vista de DTM interpolado de 500x360, paso de DTM 20.
Método generador utilizado: Distribución de carga continua.

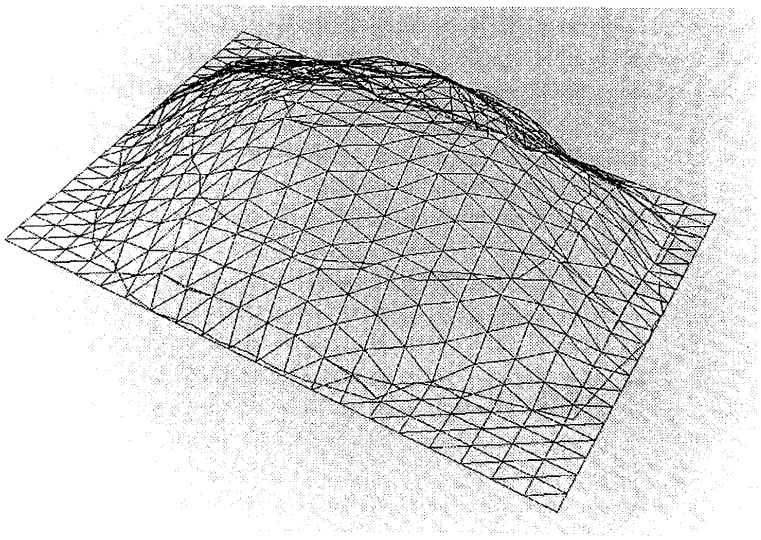


Fig. 11 – Vista de DTM interpolado de 500x360, paso de DTM 20
Método generador utilizado: Distribución de carga continua.

7.- CONCLUSIONES

Ha sido probada la factibilidad de generar MDT's a partir del diseño del perfil del terreno mediante mapas de elevación. Para ello ha sido utilizado un método de interpolación basado en el cálculo del campo electroestático generado por distribuciones de carga.

Los resultados obtenidos creemos que son muy satisfactorios y de posible uso en la enseñanza de sistemas de información geográfica, ya que su grado de dificultad conceptual no es excesivo. Y en cualquier caso con la herramienta de desarrollo adecuada basada en el método propuesto los alumnos podrán crear datos geográficos cuya visualización les proporcione entornos realistas.

7.- REFERENCIAS BIBLIOGRÁFICAS

[BURROUGH91], Burrough, P., "Principles of Geographic Information System for Land Resources Assessment"., Monographs on Soil and Resource Survey. Oxford University Press, Oxford England, 1991, pp. 1-300.

[CHANCE90], Chance, A., Newel, R. and Theriault, D., "An Object Oriented GIS: Issues and Solution", Proc. European Conference on Geographical Information System (EGIS). Annual Conference, Ultrecht, NL, 1990, pp. 179-188.

[COSMAN90], Cosman, M. A., Mathisen, A. E. and Robinson, J. A., "A New Visual System to Support Advanced Requirements", Proceedings, IMAGE V Conference, 1990, pp. 370-380.

[FALBY93], Falby, J. S., Zyda, M. J., Pratt, D. R., and Mackey, R. L., "Hierarchical DataStructures for Real-Time Three-Dimensional Visual Simulation", Computer and Graphics 17, 1993, pp. 144-151.

[FLORIANI95], De Floriani, L. and Puppo E., "Hierarchical Triangulation for Multiresolution Surface Description", ACM Transation on Graphics 14, 1995, pp. 363-411.

[SAMET84], Samet, Hanat, "The Quadtree and Related Hierarchical Data Structures", ACM Computing Surveys 16, 1984, pp. 187-200.

[SCHEE95], Schee, L. H., Jense, G. L., "Interacting with Geographical Information in a Virtual Environment", Proc. Joint European Conference on Geographical Information. The Hague, NL, 1995, pp. 151-156.

[SCHRODER94], Schroder, F. and Rossbach, P., "Managing the Complexity of Digital Terrain Models", Computer and Graphics 18, 1994, pp. 775-783.

EDICIÓN AUTOMATIZADA DEL MTN25: EXPERIENCIA DOCENTE DE UN PROYECTO CARTOGRÁFICO

María Guadalupe Rodríguez[1], Agustín Cabria[1]

[1]*Universidad de Alcalá Dpto de Matemáticas. Área de Cartografía, Geodesia y Fotogrametría*
e-mail: guadalupe.rodriguez@alcala.es
acabria@mfom.es

RESUMEN: En el presente artículo exponemos la experiencia docente referente a la elaboración de cartografía digital, de acuerdo con la metodología contrastada e implantada satisfactoriamente en el Instituto Geográfico Nacional para la producción del Mapa Topográfico Nacional 1:25000

1.- INTRODUCCIÓN.

En 1975 y de acuerdo con las directrices del Plan Cartográfico Nacional, comenzó a publicarse la serie cartográfica de hojas a escala 1:25.000 bajo la denominación MTN25. Esta serie nació con la filosofía de servir de complemento al MTN50 en las zonas que así lo requiriesen, aunque posteriormente, a raíz de la profunda evolución tecnológica experimentada en todos los ámbitos y ante las crecientes demandas sociales, se decidió el recubrimiento completo del territorio nacional a la escala 1:25.000 quedando así la serie constituida por unas 4.200 hojas. A partir de 1995 se comenzó a publicar también con formato digital.

En el marco del convenio suscrito entre el IGN y la Universidad de Alcalá, y dentro de la parte práctica de la asignatura Procesos Cartográficos, planteamos que algunos alumnos ejecutaran la fase de edición de una hoja piloto del MTN25, a partir del fichero de restitución numérica, y mediante el uso del menú informático programado por el IGN para ello. Esta práctica nos pareció muy interesante, ya que los alumnos podían experimentar con las últimas tecnologías de producción cartográfica digital, al mismo tiempo que manejaban los conceptos cartográficos básicos.

Asimismo, este adiestramiento en la elaboración de cartografía básica digital podría ser útil para algún alumno que en el futuro trabajase en el ámbito de la empresa privada, puesto que el IGN oferta de manera regular, mediante concurso público, la formación y edición de algunas hojas del MTN25.

2.- SOPORTE LÓGICO

Para la formación y edición interactivas del MTN25, el IGN diseñó una herramienta personalizada a partir de menús desplegables, programándose unas 1000 aplicaciones entre comandos de usuario, programas en lenguaje FORTRAN y rutinas MDL. Se utilizó como software de base el CAD MicroStation.

La programación se basó en las siguientes líneas conceptuales:

- *Facilidad de manejo.* No se necesitan conocimientos informáticos previos. Menús de trabajo con posibilidad de utilización a través de ratón/tableta y teclado.

- *Programación compilada.* La programación "blindada" asegura la uniformidad del trabajo y evita iniciativas no deseadas por parte del operador.

- *Entornos de trabajo amigables.* El aspecto de los ficheros en pantalla es muy parecido a la representación en papel, lo que facilita el trabajo interactivo del operador.

- *Compacidad.* En el proceso completo de una hoja del MTN25 únicamente se utilizan dos ficheros: uno contiene los usos del suelo, y el otro, el resto de la información.

- *Manipulación de los elementos orientada a su posterior carga en Base Cartográfica Numérica.* Todo elemento del mapa tiene codificación única y continuidad geométrica a lo largo de su recorrido.

3.- DESCRIPCIÓN DE LAS ETAPAS QUE COMPONEN EL PROCESO PRODUCTIVO DEL MTN25

La metodología aplicada en la producción del MTN25 se recoge en un plan general, que define los objetivos, tareas, situación y tiempos de cada estadio del proceso. En todas las fases del proceso se realiza una validación de la información tanto de entrada como de salida. De esta manera, se evitan el arrastre y acumulación de errores en etapas posteriores.

Las etapas que componen el proceso son:

a) Tratamiento previo de la información procedente de la restitución numérica

La denominada "fase previa" tiene como objetivo la obtención de un trazado con cierto "aspecto" de mapa, que sirva como soporte de trabajo y preparación de las fases siguientes.

La "fase previa" consiste, a grandes rasgos, en el paso de un fichero en 3 dimensiones (3D) a un fichero en 2 dimensiones (2D), la creación de recintos de usos del suelo a partir de los polígonos restituidos, la simbolización automática de ciertos elementos del mapa (puntos acotados, abrevaderos, etc.) y la colocación de los exteriores o información marginal. Simultáneamente, se detectan posibles anomalías existentes en el fichero: elementos con codificación incorrecta, falta de continuidad geométrica, solapamiento de elementos disjuntos, etc. Al mismo tiempo, se efectúa la recodificación de los elementos incorporados en la restitución, pasando de los códigos fotogramétricos al denominado "código mixto".

En la fase previa los ficheros 3D pueden también provenir de un proceso de digitalización. Esta posibilidad no la desarrollaremos, dado que no ha sido el objeto de la práctica.

b) Formación

Podemos definir la formación como el conjunto de tareas encaminadas a la selección y referenciación geográfica precisa de la información que conformará el contenido del mapa para una escala determinada. En el MTN25, la información geográfica se agrupa en los siguientes bloques temáticos:

- Altimetría
- Planimetría
  - Construcciones
  - Comunicaciones
  - Hidrografía
  - Usos del suelo
  - Toponimia
  - Divisiones Administrativas

Habitualmente, esta etapa consta de tres partes claramente diferenciadas:

1.- Estudio y preparación previa de la minuta en gabinete.
2.- Revisión en campo.
3.- Redacción de la minuta definitiva

c) Edición

Por edición se entiende la manipulación y procesamiento de la información contenida en la minuta para representarla según la simbolización y el nivel de generalización definidos en la norma cartográfica de la serie.

El trabajo de edición se realiza parcialmente de manera automática y en parte de manera interactiva, a través del menú de trabajo diseñado y programado específicamente para el MTN25.

Entre otras, se realizan las siguiente operaciones:

- *Tratamiento de altimetría*: representación de desmontes y terraplenes, escarpados, roquedos; rotulación de curvas de nivel directoras; etc.

- *Tratamiento de vías de comunicación*: simbolización de carreteras y ferrocarriles; resolución de intersecciones (enlaces/raquetas, nudos, pasos inferiores o superiores); etc.

- *Tratamiento de construcciones*: simbolización de edificaciones; adecuación de la estructura de cascos de población en función del nivel de generalización; arreglo de solapamientos con vías de comunicación; etc.

- *Tratamiento de la hidrografía*: creación de recintos a partir de polígonos de embalses, lagunas, etc.; simbolización de elementos puntuales, simbolización de elementos lineales (canales, acequias, etc.); etc.

- *Tratamiento definitivo de usos del suelo*

- *Tratamiento de la información marginal del mapa*

- *Rotulación de la toponimia*: selección correcta de fuentes y cuerpo en función de la categoría del topónimo, ubicación geográfica óptima (tanto aisladamente como en relación con los elementos geográficos circundantes); etc.

d) Control de Calidad

Todo el trabajo de las fases precedentes se plasma en un trazado con un aspecto muy aproximado al de la publicación impresa. Este trazado se somete de nuevo a una minuciosa revisión, efectuada por una unidad de trabajo físicamente aislada del entorno de producción.

El objetivo de esta etapa es doble. Por una parte, garantizar la homogeneidad del mapa y por otra parte, el asegurar el estándar de calidad de la serie, tanto a nivel de contenidos como de representación cartográfica.

Como paso previo a la filmación de positivos se obtiene la prueba digital de color, a través de un RIP que utiliza un algoritmo que realiza la separación de color.

e) Filmación de positivos

Completadas las fases anteriores, el paso final consiste en la rasterización de los ficheros vectoriales y filmación de fotolitos en fotoplotter láser.

4.- DESCRIPCIÓN DE LA PRÁCTICA

a) Introducción: Objetivo y material

Como ya hemos indicado, el objetivo de la práctica que desarrollamos con nuestros alumnos es realizar la "fase previa" y las tareas de edición de una hoja piloto del MTN25, mediante el uso del menú informático programado para ello.

Los alumnos disponen del fichero de restitución 3D y de la correspondiente hoja publicada (la cual sirve como minuta de trabajo), así como de documentación complementaria que les ayude a tomar decisiones, al no poder efectuar las correspondientes revisiones de campo.

b) Descripción del menú utilizado en la práctica

Disponemos de un menú de pantalla en el que las distintas opciones se van seleccionando con el cursor a través de unas casillas puestas en dos columnas. La columna de la izquierda contiene los temas principales de trabajo (comunicaciones, construcciones, etc.). La selección de cualquier tema hace aparecer en la columna de la derecha el submenú correspondiente, que recoge todas las herramientas necesarias para el tratamiento de los elementos geográficos de dichos temas. En general, se ha adoptado el criterio de escribir en mayúsculas las opciones que se corresponden con submenús y en minúsculas las órdenes a ejecutar.

El orden de las diferentes casillas del menú ayuda a realizar de manera secuencial las operaciones de edición.

c) Operaciones realizadas en la práctica

La secuencia de operaciones seguida se muestra en el apéndice Tabla 1:

5.- CONCLUSIONES

Como consecuencia de la secuencia de operaciones descrita en el apartado anterior, los alumnos habrán obtenido el fichero formado y simbolizado de una hoja del MTN25. Durante el proceso han tenido que tomar una serie de decisiones como son la selección de puntos acotados, la generalización de cascos urbanos, la revisión del acuerdo entre altimetría e hidrografía, etc. Estas operaciones requieren que alumno aplique los conocimientos cartográficos adquiridos, todo ello dentro de la norma cartográfica del MTN25.

En la planificación docente de la práctica, se tuvo especial cuidado en que no consistiese exclusivamente en el manejo de la aplicación informática. Para ello, al final de la práctica el alumno compara su trabajo con la hoja oficial correspondiente del MTN25, lo que permite una autoevaluación de su mapa.

Pedagógicamente, se pueden citar, entre otras, las siguientes ventajas:

- Se trata de una práctica real basada en un proceso actual de producción cartográfica, y no en una concepción teórica o abstracta.

- El alumno se familiariza con las técnicas más modernas de elaboración de cartografía digital.

- Obliga a contemplar de manera global los conceptos cartográficos básicos.

- Facilita la inserción del alumno en el mercado laboral, mayormente en empresas dedicadas a la elaboración de cartografía digital.

- Da conocer al alumno una escala cartográfica (1:25 000) diferente de las que ya conoce (escalas mayores).

6.- REFERENCIAS

Consejo Superior Geográfico. Instituto Geográfico Nacional. *"Normas cartográficas para la ejecución del MTN25"*. Madrid, 1985.

Torroja Menéndez, J. "Los antecedentes del Mapa Topográfico Nacional y la Real Academia de Ciencias". Revista Mapping, nº 11. Mayo 1993.

Instituto Geográfico Nacional. *"Formación y edición automatizada del MTN25"*. Madrid, 1997.

Instituto Geográfico Nacional. "Compendio de documentación para el MTN25". Madrid, 1997

Tabla I: Secuencia de operaciones

Tema	Descripci in	Operaciones
FASE PREVIA	-Fase previa a la edición	**PREVIO RESTITU:** - Fase previa de restitución
COMUNICACIONES	- Tratamiento de vías de comunicación. Clasificación de carreteras y ferrocarriles. Resolución de enlaces/raquetas, cruces, etc. -Colocación de hitos kilométricos de carretera y ferrocarril	**PREVIO RESTITU:** - Tratamiento autopistas - Tratamiento carreteras **NUEVOS COMUNIC** **TRATAMI COMUNIC**
ALTIMETRÍA	- Generalización de puntos acotados - Rotulación de curvas directoras - Representación de desmontes y terraplenes	**TEXTOS-OROGRAF:** - COTAS -Curvas **TRATAMI-RELIEVE** **NUEVOS-RELIEVE**
HIDROGRAFÍA	- Cierre de recintos - Clasificación de ríos, arroyos. - Representación de elementos con pattern - Colocación de hitos kilométricos de canales - Rotulación de curvas batimétricas - Acuerdo entre altimetría e hidrografía	**TRATAMI HIDROGRA:** - Cerrar automáticamente - Lanzar pattern **NUEVOS HIDROGRA** **NUEVOS OTROS**
CONSTRUCCIONES	- Clasificación y simbolización. - Representación de elementos con pattern - Tratamiento de cascos urbanos	**TRATAMI-CONSTRU** - Cerrar Automát. - Lanzar pattern **NUEVOS CONSTRU**
LÍMITES ADMINISTRATIVOS	- Clasificación y simbolización de líneas límite administrativas	**TRATAMI OTROS** - Límites Administ **NUEVOS OTROS**
USOS DEL SUELO	- Cierre de recintos. Clasificación	**PROCESO AUTOMAT**
EXTERIORES	-Tratamiento de la información marginal del mapa - Lanzamiento del marco y la cuadrícula U.T.M.	**MONTAJE** - Marco - MARGEN - LEYENDA **ENGANCH:** - MONO_ 25
ROTULACIÓN	-Rotulación de la toponimia del negro y azul	**TEXTOS** - CONSTRU - OROGRAF - HIDROGR -COSTA

ÁREA DE INFORMÁTICA EN ENSEÑANZA MEDIA

Responsable: Dña. Blanco Baños
E-mail: polglase@eco.uc3m.es
Responsable: Dña. Carmen Arriero
E-mail: mcarrier@platea.pntic.mec.es

LAS ACTITUDES HACIA EL USO DE ORDENADORES EN LA ENSEÑANZA SECUNDARIA EN FUNCIÓN DEL GÉNERO

Esther Ruiz Ben v. Marschall
Freie Universität Berlin/Universidad Complutense de Madrid (Fctad. de CC. Políticas y Sociología)

Descriptores: informática, enseñanza secundaria, actitudes, género.

I.- INTRODUCCIÓN

El desarrollo progresivo de la informática y la introducción expansiva de las tecnologías de la información durante las últimas décadas en la mayoría de los ámbitos sociales ha dado lugar, entre otros aspectos, a cambios drásticos en las demandas de cualificación del mercado de trabajo. Especialmente en aquellos sectores, en los que las tecnologías de la información han sido mayormente introducidas en España, es donde se observa un mayor crecimiento de ofertas de empleo incrementándose paralelamente la demanda de trabajadores/as con un medio o alto nivel de cualificación, procedentes principalmente del ámbito científico-técnico.

Sin embargo, la presencia de las mujeres en este tipo de estudios y en trabajos que demandan este perfil de cualificación no es igual respecto a la de los hombres. Según datos sobre la estructura sectorial de la actividad femenina relativos al año 1996 un 66% de las mujeres ocupadas se concentran en el sector servicios, un 11% en la industria y un 6% en la agricultura. Es decir, cuatro de cada cinco mujeres ocupadas se encuentran en el sector servicios experimentando los ámbitos de la "atención de criaturas y personas mayores, revaloración de los espacios públicos urbanos, conservación y valoración del patrimonio medio ambiental y cultural y desarrollo de la infraestructura de servicios local (...)"[i] un fuerte crecimiento de empleo femenino en los últimos años.

Además de esta sectorización y aunque el sector servicios presenta un gran potencial de acuerdo con los fuertes crecimientos de empleo que experimenta en relación con el resto de los sectores, otra de las características de la integración de las mujeres al mercado de trabajo en las últimas décadas es la concentración de su actividad. Es decir, la presencia de las mujeres en el mercado de trabajo se concentra en actividades determinadas, peor remuneradas, de mayor precariedad y con bajas exigencias en cuanto a la cualificación que demandan[ii], a diferencia de la situación de los hombres que se encuentran ocupados en un mayor número de ramas de actividad y continuan manteniendo el predominio de las ocupaciones de mayor prestigio y poder[iii]. Entre estas destacan las dedicadas a la producción técnica y especialmente las que se enmarcan en el ámbito de las tecnologías de la información y la comunicación[iv]. Las mujeres mantienen, sin embargo, en este campo un papel fundamentamente de usuarias.

No obstante, según afirman Carrasco et al. (1997), ya no se puede hablar de la "fuerza de trabajo femenino" como un grupo homogéneo, ya que existe un grupo de mujeres que con una alta cualificación y nivel de estudios desarrollan trabajos estables y de prestigio. Este aspecto supone que paralelamente a la segregación laboral entre los hombres y las mujeres que se manifiesta principalmente mediante la segmentación de la actividad de estas, tiene lugar un proceso de diferenciación dentro del propio grupo de las mujeres determinado de manera fundamental por su nivel de cualificación[v].

En el ámbito de las cualificaciones y las especialidades de estudios que eligen las mujeres se manifiesta asimismo una concentración en aquellos tipificados tradicionalmente como femeninos manteniéndose modelos de orientación profesional diferenciados en función del género. Esta tendencia ha quedado reflejada en el informe realizado por la Organización para la Cooperación y el Desarrollo Económico correspondiente al año 1997. En los ámbitos de la informática, las matemáticas, la ingeniería o la arquitectura, el número de mujeres diplomadas es mucho menor que el correspondiente a los hombres[vi].

En estudios de carácter científico-técnico en España la presencia masculina es predominante, si bien, no de manera tan destacada como en otros países occidentales[vii]. Concretamente en el caso de la Informática la tasa de feminización se encuentra, en general, entre un 20% y un 30%[viii].

En la enseñanza secundaria se refleja ya una tendencia a la segregación de las preferencias por determinadas materias[ix] y perspectivas profesionales en función del género[x]. La distancia de las chicas hacia los estudios de carácter científico-técnico en general se pone asimismo de manifiesto[xi], concretamente, en sus actitudes hacia la informática, como se ha indicado en ciertos estudios realizados con muestras de estudiantes de enseñanza secundaria en España[xii]. Esta etapa de la enseñanza, que coincide con la adolescencia resulta de especial importancia, ya que en ella, las alumnas y los alumnos toman sus primeras decisiones respecto a su trayectoria formativa y adquieren experiencias en las distintas especialidades de estudios que van conformando la base de posteriores decisiones vocacionales.

Los conocimientos en informática y las habilidades en el manejo de ordenadores constituyen actualmente una necesidad formativa en relación con las demandas de cualificación en el mercado de trabajo[xiii], pero además la importancia de la informática respecto a su influencia en amplios sectores sociales y a su capacidad innovativa se enmarca en el ámbito de la toma de decisiones en relación con la orientación del progreso, aspecto vedado tradicionalmente para las mujeres. Es precisamente de esta tradición marcada por el dominio masculino de la ciencia y la técnica como fuentes de poder, de donde parte la distancia de las mujeres hacia la informática, determinada por los estereotipos de género que han cristalizado socialmente de acuerdo con esta situación de desigualdad entre los hombres y las mujeres.

En el presente trabajo se aborda el estudio de la valoración de los alumnos y las alumnas de enseñanza secundaria en el manejo de ordenadores en función del género desde una perspectiva psicosociológica. Es decir, consideramos que el género constituye una categoría psicosocial, o como señala Barberá (1998) "(...) un factor estimular que opera tanto en la evaluación que se hace del comportamiento de los demás como respecto a la autovaloración de la propia conducta. (...) lo fundamental no es saber si hay o no diferencias comportamentales entre varones y mujeres, o entre rasgos de personalidad masculinos y femeninos, sino conocer las creencias, suposiciones y expectativas que determinan la aparición de tales comportamientos y rasgos. Se plantea que las creencias, suposiciones y estereotipos ejercen un poder comportamental considerable, que influye y crea conductas, actitudes, intereses y preferencias en las personas. Por eso, esta estrategia no delimita los conceptos de sexo y género, sino que los concibe como un sistema estimular global o una categoría psicosocial."[xiv] En este sentido, coincidimos con Shashaani (1993) al afirmar que el estudio de las diferencias de género debe tener en cuenta el proceso de socialización de los hombres y las mujeres. Es durante este proceso en el ámbito familiar, escolar y de las amistades donde se internalizan los estereotipos de género asociados, concretamente a la ciencia y la técnica y de modo específico a la informática.

1.- Base Teórica

Las investigaciones realizadas acerca de las actitudes de los/as adolescentes hacia la informática y los ordenadores en los últimos años revela diferencias en función del género en relación con la posesión de ordenadores, representada principalmente por los chicos (Nichols (1992), Culley (1986), Chen (1986); Levin & Gordon (1989); Faultisch-Wieland (1992), Funken (1993); Badagliacco (1990); Ogletree & Williams (1990) y su uso (Miura, 1987).

La mayor experiencia de los chicos en el uso extraescolar de ordenadores ha demostrado ser, además uno de los factores de mayor influencia sobre la determinación de las actitudes positivas de los chicos hacia la informática y los ordenadores (Chambers & Clarke (1987); Levin & Gordon (1989); Durndell, Glissov & Siann (1990), Shashaani (1997); Alonso & Gastaudi Gil (1992). En algunos casos, este tipo de experiencia que los chicos adquieren en el ámbito extraescolar constituye un medio de orientación de sus elecciones futuras de estudios universitarios, lo cual les ayuda a tener ideas más claras y determinadas sobre las opciones que tienen a su alcance en contraste con la situación más indefinida de las chicas en este mismo aspecto (Alemany (1992).

Otros estudios muestran que los chicos y las chicas emplean los ordenadores con distintos fines. Los chicos tienden a usar los ordenadores para programar más a menudo que las chicas, mientras en relación con el uso de aplicaciones informáticas, sobre todo, respecto al uso de programas de tratamiento de textos no se observan diferencias entre ambos sexos (Chen, 1986; Durndell et al., 1990). Theunert & Schorb (1992) hallaron en su investigación que mientras los chicos se mostraban más interesados en experimentar con los ordenadores y conocer su funcionamiento y características técnicas, las chicas solían orientarse principalmente hacia un uso concreto de estos. Esta tendencia fue, asimismo, demostrada en el estudio llevado a cabo por Gaicquintia et al. (1993).

En cuanto a las investigaciones desarrolladas en el contexto educativo, diversos estudios han mostrado la imagen masculina que se atribuye a los ordenadores, que se observa también en el ámbito extraescolar (Loyd et al. (1987). Los estereotipos de género relacionados con la informática y los ordenadores, que los/as alumnos/as de enseñanza secundaria adquieren a lo largo del proceso de socialización, por medio de sus experiencias sociales con miembros de su familia, en los procesos de interacción con los profesores o con las amistades (Noller y Paul (1991), Metz-Göckel et. al. (1991), Ritter (1996); Durndell, A. (1990); Measor (1984); Vredenburg et al. (1984); Levin y Gordon (1989), Colley, Gale y Harris (1994)) , asi como las actitudes y comportamientos que conforman durante estos procesos han sido considerados en varios estudios (Shashaani (1993); Alonso & Gastaudi (1992); Sanders (1984).

En su mayoría, estas investigaciones coinciden en señalar la existencia de actitudes más positivas hacia el uso de ordenadores en el colegio por parte de los chicos que de las chicas.
En España, en cuanto a la interacción entre los/as alumnos/as durante las clases de informática y su relación con el modo de usar los ordenadores en el aula, Alonso y Gastaudi Gil (1992) señalan, en referencia a la investigación-acción realizada entre 1986-88, a partir de las observaciones realizadas en las clases, el hecho de que las chicas no se atreven a usar los aparatos, pero si a adquirir conocimientos teóricos acerca de ellos. Los chicos, sin embargo, que cuentan con mayor experiencia con los ordenadores, segun afirman estas autoras, se agrupan alrededor del aparato y lo usan sin temor, sin esperar, al contrario que las chicas, instrucciones del profesorado, "acaparando", además, el uso del ordenador, sin dejar oportunidad a las chicas para ello[xv]. En cuanto a las opiniones expresadas por el profesorado entrevistado, el 99% consideró que los/as alumnos/as presentan puntos de partida diferentes en función del sexo

respecto al uso de tecnologías y el 100% afirmó que las chicas se encontraban menos motivadas que los chicos hacia el uso de tecnologías de la información.

Si bien en esta investigación-acción, que comentan Alonso y Gastaudi Gil, se pone de manifiesto la diferencia actitudinal de los/as alumnos/as hacia los ordenadores, en función del sexo, durante las clases de informática, la opinión de la mayor parte del profesorado que ha participado en el proyecto Atenea[xvi], es que tales diferencias no existen[xvii] [xviii].

La investigación llevada a cabo por Cabero Almenara et. al. (1993) analiza las actitudes hacia los ordenadores de 275 estudiantes de siete centros de E.G.B., 315 de tres centros de enseñanza media y 325 de cuatro universidades de Sevilla capital y su provincia y Huelva capital. En el estudio se considera el sexo como una de las variables influyentes sobre las actitudes de los/as estudiantes hacia los ordenadores junto con el nivel de estudios, presencia del ordenador en casa, uso del ordenador, asi como la formación previa en informática. Los resultados referidos a la influencia de la variable sexo sobre las actitudes de los/as estudiantes hacia los ordenadores, pusieron de manifiesto la existencia de diferencias significativas entre las actitudes de los alumnas hacia los ordenadores y las de las alumnos, presentando estos en el diferencial semántico empleado, medias significativamente superiores a las que mostraron las alumnas[xix]. Los autores de esta investigación argumentan que estas diferencias pueden deberse a factores socioculturales, afirmando que el uso y "dominio" de la tecnología "más o menos sofisticada" se orienta al hombre, mientras la mujer emplea tecnologías de carácter "doméstico". Es decir, el acceso "concedido de interacción con el medio, establece tales diferencias". Cabero Almenara et. al. señalan además que, las diferencias actitudinales de los/as estudiantes hacia los ordenadores en función del sexo, no son tan destacadas en la enseñanza básica, quizás por tener los chicos y las chicas oportunidades de acceso similar al ordenador en el colegio[xx].

De acuerdo con Eccles (1987), los padres y el profesorado juegan un papel muy importante como modelos y agentes socializadores en el proceso en el que los/as adolescentes desarrollan sus expectativas y conforman valores relativos a las diversas materias de estudio. El apoyo y refuerzo específico de estos agentes, asi como sus propios valores en relación con las distintas materias influyen sobre las actitudes y comportamientos de los chicos y las chicas. En este sentido, diversos estudios han confirmado que los padres muestran expectativas diferenciadas respecto a la trayectoria educativa y profesional de sus hijos e hijas, vinculadas a estereotipos de género (Shashaani (1993); Alonso & Gastaudi (1992).

El uso del ordenador por parte de los chicos y las chicas con diferentes miembros de la familia reflejan pautas de comportamiento relacionadas con los roles de género en el proceso de socialización que tiene lugar dentro del ámbito familiar, según señalan Colley et al. (1994). En el estudio realizado por estos autores se muestra que los miembros de la familia pertenecientes al mismo sexo comparten pautas de comportamiento similares en relación con los ordenadores. El tener un hermano que emplea el ordenador se mostró asociado a una menor ansiedad y mayor confianza en sí mismo/a para usar ordenadores por parte de ambos sexos, además de encontrarse vinculado con un interés más positivo en este aspecto por parte de las chicas. El uso del ordenador por parte del padre mostró un efecto positivo sobre las actitudes de los hijos hacia los ordenadores, mientras para las hijas fue el uso del ordenador con la madre el que presentó este mismo efecto. De este modo, Colley et al. (1994) sugieren "(...) cualquier examen de la experiencia con ordenadores en el hogar ha de tener en cuenta la influencia de otros miembros familiares como modelos de rol y fuente de oportunidades para aprender "[xxi].

En el contexto educativo, diversos estudios han prestado especial atención a los procesos de interacción que se desarrollan durante las clases de informática. La supuesta neutralidad de los centros mixtos ha sido cuestionada en paises como Alemania o Australia y Estados Unidos,

partiendo de los resultados de estas investigaciones que han puesto de manifiesto pautas estereotipadas de comportamiento en relación con el género durante los procesos de interacción que tienen lugar en las aulas de centros mixtos, especialmente en las clases de informática (Altermann-Köster et. al. (1990), Alonso y Gastaudi Gil (1992), Faultisch-Wieland (1992, Funken (1993); Jones & Clarke (1995)).

El comportamiento diferenciado de los/as profesores/as con los alumnos y las alumnas en función del género y el desequilibrio entre la experiencia extraescolar de los alumnos y las alumnas de enseñanza secundaria en el uso de ordenadores influye sobre las actitudes de las alumnas hacia la informática y los ordenadores reforzando su distancia hacia ellos (en España, la mayor parte del profesorado que imparte la asignatura de informática esta formada por hombres[xxii], confirmando la ausencia de roles femeninos en el ámbito de la informática). Por ello, diversos proyectos de investigación han sido desarrollados en Alemania con el fin de analizar los posibles beneficios del aprendizaje de la informática por parte de las alumnas en centros segregados y en cursos diseñados especificamente para este grupo (Alterman-Köster et al., 1990; Faultisch-Wieland, 1992).

2.-ESTUDIO EMPÍRICO SOBRE LAS ACTITUDES DE ALUMNOS Y ALUMNAS DE CENTROS MIXTOS HACIA LOS ORDENADORES EN FUNCIÓN DEL GÉNERO

2.1.- *Objetivos de la investigación*

Partiendo de las conclusiones de las investigaciones anteriormente comentadas, se ha diseñado el presente estudio, cuyo objetivo ha sido determinar si existen diferencias significativas entre la valoración, expectativas de éxito y la experiencia en el uso de ordenadores de los alumnos y las alumnas de enseñanza secundaria en centros mixtos, asi como entre las de las alumnas que estudian en centros mixtos y aquellas que estudian en centros segregados o de mayoría femenina en relación con el género. En este artículo nos limitaremos a presentar las conclusiones relativas a las diferencias entre alumnos y alumnas de enseñanza secundaria en cuanto a sus actitudes hacia los ordenadores y su uso.

Se ha partido de la hipótesis de que los alumnos tienen mayor experiencia extraescolar en el uso de ordenadores y valoran más positivamente que las alumnas de centros mixtos esta actividad, siendo estas diferencias significativas.

2.2.- *Muestra empleada*

La muestra seleccionada en este estudio se compone de 151 alumnos de centros mixtos y 274 alumnas de enseñanza secundaria, de las cuales 128 estudian en un centro de mayoría femenina (85%) (Colegio "Jesús Maestro"). En cuanto a los centros mixtos en los que estudian los alumnos y las alumnas, uno tiene carácter privado (Colegio "Arturo Soria") y dos público (IES "San Mateo" e IES "Cardenal Herrera Oria"). La edad de los/as estudiantes oscila entre los 14 y los 17 años. Durante el curso 1995/96 se pidió a estos grupos de estudiantes que rellenasen un

cuestionario en las clases de tutoría bajo la supervisión de los/as tutores/as o profesores/as de informática.

2.3.- *Descripción del Cuestionario*

El cuestionario empleado, que se basa en el empleado en un estudio realizado previamente en Pittsburg (USA) (v. Shashaanii, 1993) se compone de siete secciones relacionadas con los siguientes aspectos: información de carácter demográfico, relación con los padres, el estudio de la informática y perspectivas vocacionales en relación con esta materia, estereotipos de género asociados a los ordenadores, percepción de las actitudes de los padres, profesores y amigos hacia los ordenadores, experiencia en el uso de ordenadores y actitudes hacia los ordenadores. En este artículo no centraremos en el último aspecto citado, que se describe con mayor detalle a continuación.

2.2.1.- *Actitudes hacia el uso de ordenadores*

Esta sección incluye una escala de actitudes que se compone de items basados, en parte, en un estudio sobre las diferencias actitudinales respecto a los ordenadores entre chicos y chicas, estudiantes de enseñanza secundaria, en función del género, realizado en Estados Unidos por Shashaani (1993, op. cit.). Esta escala esta formada por 17 items, que tras el correspondiente análisis factorial ha quedado subdividida en cuatro subescalas: *interés* por los ordenadores (5 items), *confianza* en sí mismo/a para usar ordenadores (5 items), *conceptos acerca de los ordenadores* (4 items), *conceptos sobre el uso de ordenadores* (3 items).

La escala de cinco puntos, tipo Likert empleada para cada item, oscila entre el absoluto acuerdo, correspondiente al valor 5 y el absoluto desacuerdo, correspondiente al valor 1. Se emplearon items negativos y positivos con el fin de comprobar la fiabilidad de las respuestas.

Los valores de las preguntas con carácter negativo fueron recodificados antes de realizar el análisis factorial. Posteriormente se llevó a cabo un análisis de consistencia interna de cada una de las subescalas resultantes del análisis factorial, cuyos resultados fueron: *interés por los ordenadores* (α de Crombach = .8681); *confianza en sí mismo/a respecto al uso de ordenadores* (α de Crombach = .7531); *conceptos sobre los ordenadores* (α de Crombach = .5397); *conceptos sobre el uso de ordenadores* (αde Crombach = .5305).

2.3.-*Análisis de los datos*

Para analizar las diferencias respecto a la valoración del uso de ordenadores de los alumnos y las alumnas de centros mixtos se han realizado cuatro t-test para muestras independientes (uno para cada subescala).

3.- RESULTADOS

3.1.- *Valoración del uso de ordenadores*

Tabla 1 **Valoración del uso de ordenadores - Alumnas y Alumnos de centros mixtos (puntuaciones medias)**

	Alumnas	Alumnos	t
Interés por los ordenadores	2. 93	3. 37 (***)	6. 52
Confianza en sí mismo/a para usar ordenadores	4. 25	4. 73 (***)	4. 62
Conceptos sobre los ordenadores	3. 78	4. 05 (***)	3. 73
Conceptos sobre el uso de ordenadores	3. 56	3. 68	1. 43

p< 0.05*; p< 0,01**; p< 0,001***

Como se puede ver en la Tabla 1, existen diferencias significativas entre las puntuaciones medias correspondientes a la valoración del uso de ordenadores de los alumnos y las alumnas de centros mixtos, excepto en relación con los conceptos acerca del uso de ordenadores, según los resultados del t-test. Los alumnos muestran puntuaciones más elevadas que las alumnas en las cuatro subescalas, es decir, valoran más positivamente el uso de ordenadores. Más concretamente, en relación con el item relativo a la importancia de los conocimientos de informática para conseguir un trabajo, asi como el correspondiente a la utilidad de los ordenadores en el ámbito laboral, los alumnos mostraron puntuaciones medias significativamente más elevadas que las alumnas.

4.- DISCUSIÓN

Los resultados del estudio empírico que hemos comentado anteriormente apoyan la hipótesis de que existen diferencias significativas entre las actitudes de los alumnos y las alumnas de enseñanza media hacia el uso de ordenadores. Los alumnos demostraron mayor interés por los ordenadores que las alumnas de centros mixtos, mayor confianza en sí mismos para usar ordenadores y conceptos más positivos que sus compañeras acerca de los ordenadores.

Estos resultados coinciden con los hallados en las investigaciones realizadas por Chen (1986.), Shashaani (1993), Busch (1995) y Sanders (1984). Sin embargo, al contrario de la tendencia encontrada en estos estudios, las alumnas mostraron, según los resultados de nuestra investigación, una confianza positiva en sí mismas para usar ordenadores. Este aspecto ha sido también resaltado por Lightbody & Durndell (1993)[xxiii], que lo denominaron "síndrome puedo, pero no quiero" (can, but I don't want to" syndrome). En este sentido, Durndell et al. (1995) comentan que, dado que las chicas valoran los ordenadores de acuerdo con su utilidad y no como objetos interesantes per se, incluso aquellas que muestran confianza en sí mismas para usarlos, no encuentran esta disciplina lo suficientemente atractiva.

Por otro lado, según nuestros resultados, los alumnos tienen conceptos más positivos sobre los ordenadores que las alumnas de centros mixtos, especialmente en lo que se refiere a la utilidad de los ordenadores para facilitar las tareas laborales. Sin embargo, en relación con la utilidad de los ordenadores en la vida cotidiana no se hallaron diferencias significativas entre ambos grupos, resultados que coinciden con los de Shashaani (1997), pero no con los del estudio de Wu y Morgan (1989).

Según Eccles (1994), la decisión de elegir una determinada opción de estudios o profesional se halla influida por las expectativas de éxito del sujeto respecto a cada opción, así como por el valor subjetivo que atribuye a la tarea, es decir el valor que cada persona da a cada opción, que se relaciona con el valor de incentivo (compensación implícia en la realización de la tarea) y con el valor de utilidad, relativo a las facilidades que una determinada tarea proporciona respecto a las propias metas a largo y corto plazo. Asimismo, dicho valor que cada persona atribuye a las distintas opciones que tiene a su alcance se encuentra vinculado con el coste implícito o las consecuencias percibidas por la persona acerca de la realización de la tarea, es decir, el tiempo y el esfuerzo que se prevee invertir en ella, o el miedo al fracaso.

Según los resultados de nuestra investigación, las alumnas de centros mixtos expresaron percibir una menor capacidad propia para usar ordenadores y para aprender a usarlos. Es decir, sobre la base del modelo de Eccles (1994), el valor subjetivo que las alumnas de centros mixtos atribuyen al uso de ordenadores es menor que el que muestran los alumnos. Asimismo, ya que las alumnas de centros mixtos no mostraron tanto interés y diversión respecto al uso de ordenadores y considerando además, que expresaron más miedo e inseguridad en este aspecto, se puede deducir que el coste del uso de ordenadores para ellas es mayor que para sus compañeros.

Sobre la base de estos resultados, se sugiere la consideración por parte de la comunidad educativa de la existencia de diferencias en la valoración y experiencia en el uso de ordenadores en función del género. El obviar esta situación, da lugar a la reproducción de roles estereotipados por parte de los/as estudiantes de enseñanza secundaria que se reflejan posteriormente en los perfiles profesionales de los hombres y las mujeres vinculados a roles tradicionales marcados por estereotipos de género.

REFERENCIAS

[i] García Sanz, B. (1997) "Mujeres y Empleo (1976-1996)" Madrid, Instituto de la Mujer, p. 45

[ii] Según García de Cortázar "(...) La lista de los sectores de preferente inserción para varones y mujeres coincide en pocos casos. Se aprecia un distinto patrón de ocupación sectorial según el sexo de los trabajadores. Puede hablarse de sectores masculinos (Construcción, Industrias manufactureras, Transporte) y sectores femeninos (Educación, Sanidad, Servicios domésticos, Confección)
De las doce ramas consideradas, nueve pertenecen al sector servicios, predominando aquellos que acogen actividades de asistencia colectiva o personal (Salud, Educación, Personal doméstico) y los que estan vinculados al Sector Público. De los tres restantes, uno corresponde a la Agricultura y dos a actividades industriales (Alimentación y Textil) en las que son habituales las condiciones de precariedad en el empleo y salarios bajos. (...) Estructura Laboral de las mujeres españolas. (...) Sectores débiles, ocupaciones escasamente cualificadas, actividades en proceso de fuerte transformación tecnológica son el ámbito profesional en el que desarrolla sus quehaceres una parte importante de las mujeres ocupadas." en: García de León, M.A.; García de Cortázar, M.; Ortega, F. (Coord.) (1996) "Sociología de las mujeres españolas" Madrid, Editorial Complutense, p. 129.

[iii] Carrasco, C.; Alabart, A.; Mayordomo, M.; Montagut, T. (1997) "Mujeres, Trabajos y Políticas Sociales. Una aproximación al caso español" Madrid, Instituto de la Mujer, Colección Estudios, No. 51.

[iv] En relación con las tasas de paro de las diplomadas y licenciadas en función de la especialidad, Guerrero Serón señala que "(...) Si se desmenuzan esas tasas de paro por áreas universitarias, vemos que son algo más reducidas entre las diplomadas (5,2%) y licenciadas (15,5%) en Ciencias Sociales, mientras se disparan entre las diplomadas en Ciencias Naturales y en Ingeniería y Tecnología (38,1% y 33%) y las licenciadas en Humanidades y Ciencias de la Salud (34,5 y 27,4%, respectivamente)." Guerrero Serón, A. (1996) "Nivel educativo de la Población Femenina española como proceso de construcción social" en: García de León, M.A.; García de Cortázar, M.; Ortega, F. (Coord.) (1996), op. cit., p. 87.

[v] según indican Carrasco et al. (1997), op. cit. : "(...) en ciertos sectores se desarrollan empleos femeninos cualificados que dan la posibilidad a ciertos grupos de mujeres de acceder a profesiones intelectuales, científicas y técnicas pero, por otra parte, se constata una feminización masiva y generalizada de empleos de poca cualificación. (...)" p. 115.

[vi] "(...) In the fields of mathematics, computer science, engineering and architecture, however, women earn far fewer university-level qualifications than men." en: Organización para la Cooperación y el Desarrollo Económico (1997) "Education at a Glance. OECD Indicators" p. 338.

[vii] Según datos obtenidos por la Organización para la Cooperación y el Desarrollo Económico en países como Finlandia, Gran Bretaña, Canadá o Nueva Zelanda, las diferencias entre las tasas de obtención de diplomas en Matemáticas e Informática de los hombres y las mujeres en 1991 era bastante superior a las que se observabann en España (ver: OCDE (1994) *"Análisis del Panorama Educativo"* p. 169). En Alemania un 10% de mujeres estudian Informática: Kretschmann, M. (1998) "Frauen wollen keine "isolierten Computer-Freaks" sein" en: *Der Tagesspiegel*, Di. 28. Jul., p. 30.

[viii] Alemany, C. (1992) "Yo no he jugado nunca Electro-L. Alumnas en enseñanza Superior Técnica" Madrid, Instituto de la Mujer, Colección "Estudios" , No. 31, p. 25.

[ix] Mirandes, Josefina & Tarín, Rosa (1991) *"Diferents expectatives del professorat sobre les capacitats científiques dels nois i de les noies"* (pendiente de publicación), citado en: Solsona, Nuria (1995) "Una mirada no sexista a las clases de Ciencias Experimentales" Institut de Ciénces de l'Educació, Universitat Autónoma de Barcelona, Colec. "Cuadernos para la Coeducación", p. 73.

[x] Gaviria Soto, J.L. (1993) "El sexo y la clase social como determinantes de los intereses profesionales en el último curso de secundaria" Madrid, Instituto de la Mujer, Colección "Estudios", No. 38.
En este estudio acerca de las preferencias profesionales de los/as estudiantes de último curso de enseñanza secundaria, las alumnas se mostraron interesadas sobre todo por profesiones del ámbito del derecho, la medicina, las ciencias sociales y las letras, mientras los alumnos tendían a optar por el campo de la física y la química en mayor proporción que las alumnas.

[xi] En cuanto a la elección de opciones en Bachillerato y en el Curso de Orientación Universitaria (COU), Guerrero Serón comenta: "(...) Tanto el análisis de la elección de opciones en BUP y COU, como el desglose de los datos de estudios universitarios, de primero y segundo ciclos, muestran una persistente **adscripción genérica** de lo que se da en llamar "ciencias" y "letras", tecnología y humanidades, a los géneros masculino y femenino, respectivamente. En 3° de BUP, las chicas representan el 53, 9% del alumnado de la Opción A, conocida como "Letras", mientras que solo son el 44, 9% de la Opción B, la de "Ciencias". En COU, donde las alumnas constituyen el 54,3% del alumnado, solo representan el 35,2% de la Opción A, la Científico-Tecnológica, frente al 55,9% de la B (Bio-Sanitaria), el 56,6% de la C (Ciencias Sociales) y el 57,9% de la D (Humanístico-Linguística)." en: Guerrero Serón, A. (1996) "Nivel educativo de la Población Femenina española como proceso de construcción social" en: García de León, M.A.; García de Cortázar, M.; Ortega, F. (Coord.), op. cit., p. 83.

[xii] Alonso, I. & Gastaudi, P. (1992) "Guía para el uso no sexista de las nuevas tecnologías" Madrid, Instituto de la Mujer.
Cabero Almenara, J. (Coord.) (1993) "Investigaciones sobre la Informática en el centro" Barcelona, PPU.

[xiii] (...) At a time when technological innovation is becoming an increasingly important component of industries´ ability to compete in the global marketplace and when employment is growing in high-technology, science-based sectors, closing the gender gap can be an important policy objective to ensure a sufficient supply of the required skills and knowledge." en: Organización para la Cooperación y el Desarrollo Económico (1997) "Education at a Glance. OECD Indicators" p. 338.

[xiv] Barberá, E. (1998) "Psicología del Género" Barcelona, Editorial Ariel, Colección Psicología. p. 61.

[xv] "(...) En las observaciones de aula para analizar las actitudes de chicos y chicas en relación con el uso de microordenadores y programas, se registra con frecuencia el hecho de que los niños no suelen esperar instrucciones, ya que directamente manipulan el teclado; las niñas, en la mayoría de los casos, suelen escuchar las instrucciones o asegurarse de la tarea que tienen que realizar, antes de manipularlos.
Otro hecho habitual es que los chicos acaparen los microordenadores y no dejen a las chicas tiempo para utilizarlos.(...)" en: Alonso, Isabel; Gastaudi, Paz, op. cit. (1992), p. 97.

[xvi] Proyecto "Atenea" del Ministerio de Educación y Ciencia, para la introducción de los ordenadores en las Enseñanzas Básica y Media, que comienza en el curso 1985/86.

[xvii] Ministerio de Educación y Ciencia(1994) "Estudio sobre el funcionamiento de los Programas de Nuevas Tecnologías" Madrid, Ministerio de Educación y Ciencia.

[xviii] "(...) *La actitud de las alumnas en actividades con ordenadores ?ha sido diferente a la de los alumnos?*
No.....254 (72%)
Poco diferente..... 65 (19%)
Bastante diferente.... 21 (06%)
Muy diferente..... 1 (00%) (...)" en: Ministerio de Educación y Ciencia, op. cit. (1994), p. 47.

[xix] "(...) En consecuencia, concluimos que los varones tienen actitudes más positivas hacia la informática que las hembras.

Dato que se refleja claramente al comparar las medias obtenidas por ambos grupos, superiores en los varones a las hembras.(...)" en: Cabero Almenara, Julio, et. al., op. cit. (1993), p. 257.

[xxxx] "(...)Las actitudes de los alumnos se presentan de forma diferente en función del sexo: los alumnos de sexo masculino tienden a tener actitudes más positivas y altas que los del sexo femenino. Tales diferencias creemos que pueden explicarse por causas socioculturales. Tradicionalmente la utilización y dominio de cualquier tecnología, más o menos sofisticada, es destinada al hombre, reservando a la mujer el dominio de otras tecnologías que podríamos considerar domésticas. En definitiva el nivel de oportunidad concedido de interacción con el medio, establece tales diferencias. Diferencias que cuando nos situamos en el nivel educativo de E.G.B., no se dan de forma tan pronunciada, posiblemente porque en este, y debido a la inserción del medio en la escuela, las oportunidades sean similares y no discriminatorias en función del sexo de los alumnos.
Por otra parte, aunque aqui no lo hemos contrastado, el tipo de interacción que las mujeres realizan con los ordenadores son, por lo general, cualitativamente diferentes a la de los hombres, y nos atreveríamos a decir que menos creativos y más necesarios. (...)" en: Cabero Almenara, Julio, et. al., op. cit. (1993), p. 276-77.

[xxi] "(...) any examination of home experience with computers should take into account the influence of other family members as role models and as providers of opportunities for learning" en: *Ibid.*, p. 135.

[xxii] Ministerio de Educación y Ciencia (1994) "Estudio sobre el funcionamiento de los Programas de Nuevas Tecnologías" Madrid.

[xxiii] Lightbody, P. & Durndell, A. (1993) *"Senior school pupils´ career aspirations: is gender an issue?"* Paper presented at BPS London Conference, December.

BIBLIOGRAFÍA

-Alberdi, I. (1985) "Coeducación y Sexismo en la Enseñanza media" en: Instituto de la Mujer "La Investigación en España sobre Mujer y Educación" Madrid, Instituto de la Mujer.

-Alemany, C. (1992) "Yo no he jugado nunca con Electro-L. Alumnas en Enseñanza Superior Técnica" Madrid, Instituto de la Mujer Colección Estudios.

-Alonso, I. & Gastaudi, P. (1992) *"Guía para el uso no sexista de las nuevas tecnologías"* Madrid, Instituto de la Mujer.

-Altermann-Köster, M.; Holtappels, H. G.; Kanders, M.; Pfeiffer, H.; de Witt, C. (1990) "Bildung über Computer?" Institut für Schulentwicklungsforschung der Universität Dortmund, Juventa.

-Badagliacco, J. M. (1990) "Gender and race differences in computing attitudes and experience" en: *Social Science Computer Review, Vol. 8, No. 1, pp. 42-64.*

-Barberá, E. (1998) "Psicología del Género" Barcelona, Editorial Ariel, Colección Psicología.

-Cabero Almenara, Julio (Coord.); Alba Riesco, José María; López-Arenas González, José Manuel; Pérez Díez de los Ríos, José Luis (1993)"Investigaciones sobre la Informática en el centro" Barcelona, PPU.

-Carrasco, C.; Alabart, A.; Mayordomo, M.; Montagut, T. (1997) "Mujeres, Trabajos y Políticas Sociales. Una aproximación al caso español" Madrid, Instituto de la Mujer, Colección Estudios, No. 51.

-Chambers, S.M. & Clarke, V.A. (1987) "Is inequity cumulative?: The relationship between Disadvantages Group Memebership and Students´ Computing Experience, Knowledge, Attitudes and Intentions" en: *Journal of Educational Computing Research, No. 3, pp. 495-517.*

-Chen, M. (1986) "Gender and Computers. The beneficial effects of experience on attitudes" en: *Journal of Educational Computing Research, Vol. 2, No. 3, pp. 265-282.*

-Colley, Ann M.; Gale, Mathew T.; Harris, Teri A. "Effects of Gender Role Identity and Experience on Computer Atitude Components" en: *Journal of Educational Computer Research, Vol. 10, No. 2, pp. 129-137 (1994)*

-Culley, L. (1986) "Gender Differences and Computing in Secondary Schools" Loughborough: Department of Education, Loughborough University of Technology.

-Durndell, A. (1990) "The Persistence of the Gender Gap in computing" en: *Computers & Education, Vol. 16, No. 4, pp. 283-287.*

-Durndell, A.; Glissov, P.; Siann, G. (1990) "Gender differences and computing in course choice at entry into higher education" en: *British Educational Research Journal, Vol. 16, No. 2, pp. 149-162.*

-Eccles, J.S. (1994) "Understanding women's educational and occupational choices. Applying the Eccles et al. Model of Achievement-Related Choices" en: *Psychology of Women Quarterly, No. 18, p. 587.*

-El País *"Documentación e Informática de Gestión, carreras con más salidas"* Martes, 9 de Junio 1998, No. 767, p. 32.

-García de León, M.A.; García de Cortázar, M.; Ortega, F. (Coord.) (1996), "Sociología de las mujeres españolas" Barcelona, Anthropos.

-Guerrero Serón, A. (1996) "Nivel educativo de la Población Femenina española como proceso de construcción social" en: García de León, M.A.; García de Cortázar, M.; Ortega, F. (Coord.) (1996), "Sociología de las mujeres españolas" Barcelona, Anthropos.

-Faultisch-Wieland, H. (1992) "Mädchenbildung und neue Technologien. Das hessische Entwicklungs- und Forschungsprojekt" en: Bundesminister für Bildung und Wissenschaft (Hrsg.) (1992) "Mädchen und Computer. Ergebnisse und Modelle zur Mädchenförderung in Computerkursen" Bad Honnef: Bock.

-Fauser, R. & Schreiber, N. (1990) "Was erwarten Jugendliche und Erwachsene von informationstechnischer Bildung?" Universität Konstanz.

-Funken, C. (1993) "Ist die Koedukation ein Fortschritt? Empirische Befunde zu Geschlechtsunterschieden im Informatikunterricht" en: *Computer und Unterricht, No. 9, pp. 43-48.*

-Gaicquintia, J.B.; Bauer, J.A.; Levin, J. (1993) "Beyond Technology's Promise" Cambridge, Cambridge Univ. Press.

-García Sanz, B. (1997) "Mujeres y Empleo (1976-1996)" Madrid, Instituto de la Mujer

-Gaviria Soto, J.L. (1993) "El sexo y la clase social como determinantes de los intereses profesionales en el último curso de secundaria" Madrid, Instituto de la Mujer, Colección "Estudios", No. 38.

-Jones, T. & Clarke, V.A. (1995) "Diversity as a determinant of attitudes: a posible explanation of the apparent advantage of single-sex settings" en: *Journal of Educational Computing Research, Vol. 12, No. 2, pp. 51-64.*

-Levin, T. & Gordon, C. (1989) "Gender and Computer Experience on Attitudes toward Computers" en: *Journal of Educational Computing Research, No. 5, pp. 69-88.*

-Lightbody, P. & Durndell, A. (1993) *"Senior school pupils´ career aspirations: is gender an issue?"* Paper presented at BPS London Conference, December.

-Loyd, B.H.; Loyd, D.E.; Gressard, C.P. (1987) "Gender and Computer experience as factors in the computer attitudes of middle school students" en: *Journal of Early Adolescence, No. 7, pp. 13-19.*

-Measor, L. (1984) „Gender and the sciences. Gender-based conceptions of school subjects" en: Harmmersley, M. y Woods, P. (Eds.) (1984) „Science in the Schools: The Sociology of Pupil culture" Milton Keynes, The Open Univ. Press.

-Metz-Göckel, S.; Frohnert, S.; Hahn-Mausbach, G.; Kauermann-Walter, J. (1991) „Mädchen, Jungen und Computer: Geschlechtsspezifisches Verhalten beim Umgang mit Computer" Frankfurt a. M., Opladen.

-Ministerio de Educación y Ciencia (1994) "Estudio sobre el funcionamiento de los Programas de Nuevas Tecnologías" Madrid, Ministerio de Educación y Ciencia.

-Miura, I. T. (1987) "The relationship of computer self-efficacy expectations to computer interest and course enrollement in college" en: *Sex Roles, Vol. 16, No. 2, pp. 303-311.*

-Nichols, L. M. (1992) "The influence of Student Computer-Ownership and In-Home Use on Achievement in an Elementary School Computer Programming Curriculum" en: *Journal of Educational Computing Research, Vol. 8, No. 4, pp. 407-421.*

-Noller, P.; Paul, G. (1991) „Jugendliche Computerfans. Selbstbilder und Lebensstyle- Eine empirische Untersuchung" Frankfurt a. M., Juventa.

-Ogletree, S.M. & Williams, S.W. (1990) "Sex and sex-typing effects on computer attitudes and aptitude" en: *Sex Roles, Vol. 23, Nos. 11/12, pp. 703-712.*
Organización para la Cooperación y el Desarrollo Económico (1997) "Education at a Glance. OECD Indicators"

-Sanders, J.S. (1984) "The Computer: Male, Female or Androgynous?" en: *The Computing Teacher, Vol. 11, No. 8, pp. 31-34.*

-Shashaani, L. (1993) "Gender-Based Differences toward Computers" en: *Computers & Education, Vol. 20, No. 2, pp. 169-181.*

-Shashaani, L. (1997) "Gender differences in Computer Attitudes and Use among College Students" en: *Journal of Educational Computing Research, Vol. 16, No. 1, pp. 37-51.*

-Theunert, H. & Schorb, B. (1992) "Zur pädagogischen Arbeit mit Computern" en: Bundesminister für Bildung und Wissenschaft (Hrsg.) (1992) "Mädchen und Computer"

-Vredenburg, K.; Flett, G. L.; Krames, L.; Pliner, P. (1984) *"Sex differences in attitudes, feelings and behaviors towards computers"* Paper presented at the 92nd Annual Convention of the American Psychological Association, Toronto, Canada

-Wu, Y. & Morgan, M. (1989) "Computer Use, Computer Attitudes and Gender: Differential Implications of Micro and Mainframe Usage among College Students" en: *Journal of Research on Computing in Education, Vol. 22, No. 2, pp. 214-228.*

ÁREA DE T.I Y GESTIÓN EMPRESARIAL

Responsable: Dr. Ferran Virgós
(Catedrático EITT, asesor EUI UA)
E-mail: fvirgos@correu.andorra.ad

LA INFORMATICA DE GESTION PARA EMPRESARIALES COMO ASIGNATURA INTEGRADA EN EL PROYECTO CAMPUS EXTENS DE LA UIB

Adelaida Delgado Domínguez

UIB (Universitat de les Illes Balears)
e-mail: dmiadd0@ps.uib.es, adelaida@ipc4.uib.es

RESUMEN: En el presente artículo se presenta la metodología utilizada para la impartición de la asignatura de Informática de Gestión, del 3er curso de la Diplomatura de Empresariales de la *Universitat de les Illes Balears*, siguiendo el modelo semipresencial adoptado en el proyecto denominado Campus Extens para que la asignatura pueda ser seguida por alumnos distribuidos geográficamente en las sedes universitarias de las Islas Baleares.

1.- EL PROYECTO CAMPUS EXTENS

La utilización de las nuevas tecnologías en la enseñanza universitaria ofrece un amplio espectro de posibilidades que constituyen un reto para la enseñanza convencional. Respondiendo a este reto la Universidad de las Islas Baleares puso en marcha el curso 1996/1997 el proyecto Campus Extens <http://ce.uib.es:9798> que pretende explotar las cualidades de las redes de telecomunicaciones para mejorar las oportunidades de acceso a los estudios universitarios desde cualquiera de las islas y favorecer el autoaprendizaje.

Los objetivos de dicho proyecto son:

Que los alumnos tengan acceso a los servicios educativos del campus desde cualquier lugar, de forma que puedan desarrollar personal y autónomamente acciones de autoaprendizaje y toma de decisiones (selección de contenido y materiales, itinerario, ritmo, a quién recurrir para solicitar ayuda, ...).

Que los alumnos utilicen las nuevas tecnologías de la información y la comunicación. El profesor pone a su disposición, en el servidor web de Campus Extens, materiales didácticos (contenidos de elaboración propia o bien enlaces a materiales en la web, actividades, etc.) y además el alumno dispone de herramientas de comunicación como el correo electrónico (para tutorías, subscripción a listas de distribución, intercambio de opiniones, etc), foros de discusión y chats.

Fomentar métodos interactivos de enseñanza en la universidad. Los alumnos tienen la posibilidad de participar e intercambiar información desde cualquier sitio y en cualquier momento, permitiendo a cada participante trabajar a su propio ritmo y tomarse el tiempo necesario para leer, reflexionar, escribir y revisar antes de compartir las cuestiones o información con los otros.

Para conseguir tales objetivos, en Campus Extens se propone un sistema mixto de enseñanza, combinando actividades de presencia continuada con actividades de autoaprendizaje, de tal manera que los alumnos de Menorca e Ibiza-Formentera puedan acceder a titulaciones ofrecidas en la UIB sin desplazarse a Mallorca y con unas condiciones de calidad de enseñanza similares a las de los estudiantes de la UIB.

Los profesores de Campus Extens tienen una reducción de entre el 25 y el 30% de horas lectivas durante las cuales los alumnos han de trabajar los contenidos a partir de los materiales y actividades sugeridadas por el profesor (a disposición de los alumnos a través de Internet).

2.- CONTENIDO DE LA ASIGNATURA DE INFORMATICA DE GESTION Y SITUACIONES DE APRENDIZAJE

La asignatura se ha dividido en 6 módulos, predominando la formación en nuevas tecnologías, en particular en Internet como canal de comunicación, como mecanismo empresarial para obtener información externa, como medio de publicación y difusión de información corporativa, como soporte de nuevos modelos de negocio y de transacciones comerciales y como tecnología a aplicar en el seno de la empresa para constituir Intranets. Además se dedica un módulo a los Sistemas de Información para la Empresa donde se describen y se presentan ejemplos reales y algunos casos de estudio de Sistemas de Procesamiento de Datos (DPS), Sistemas de Información para Gestión (MIS), Sistemas de Ayuda a la Toma de Decisiones (DSS) y Sistemas Expertos (ES). En este módulo se ven también algunas herramientas de análisis de sistemas.

Las estrategias y situaciones de aprendizaje, planteadas por el profesor para esta asignatura en concreto, son las siguientes:

Clases teóricas (videoconferencia). En estas sesiones el profesor desarrollará, fomentando la participación del alumno, los diversos conceptos reflejados en el temario de la asignatura. Estas clases se impartirán de forma presencial en Mallorca, y simultáneamente a través de videoconferencia en las extensiones de Menorca e Ibiza. Los alumnos dispondrán de un cuadernillo de soporte a los contenidos, en formato impreso, y también disponible en formato PDF en la sede web de la asignatura. En las aulas de videoconferencia hay un ordenador conectado a Internet y el profesor actúa como realizador, alternando la proyección de la pizarra, la de su persona y la del ordenador (pudiendo utilizar presentaciones en PowerPoint, conexiones a webs, *demos* de programas, etc).

Clases prácticas supervisadas. Estas clases se llevarán a cabo de forma presencial, con ordenadores y conexión a Internet y el alumno se ejercitará en el manejo de diversas herramientas y habilidades. Estas clases tendrán lugar en Aulas Informáticas. Los enunciados de las prácticas así como los materiales necesarios para llevarlas a cabo estarán disponibles en la red. En las extensiones de Menorca e Ibiza habrá un profesor asociado para supervisar tales prácticas y además periódicamente el profesor responsable de la asignatura se desplazará a las islas para tutorizar personalmente parte de tales practicas.

Sesiones de autoaprendizaje. En estas sesiones el alumno, mediante los materiales elaborados para tal fin y haciendo uso de las telecomunicaciones, participará en los foros de

discusión, analizará los diversos casos de estudio propuestos, se ejercitará en el uso de las diferentes aplicaciones de Internet y en la búsqueda de información en la WWW. El alumno dispondrá de una guía de estudio de la asignatura donde se darán orientaciones para el estudio autónomo, proporcionando esquemas, proponiendo actividades e indicando los recursos necesarios para llevar a cabo el autoaprendizaje.

Tutorías presenciales y electrónicas. Para las tutoriás electrónicas los alumnos disponen de una cuenta de correo electrónico específica para la asignatura la cual está integrada en la herramienta de teleeducación utilizada, el WebCT <http://www.webct.com>, la cual se comentará más adelante.

Subscripción a listas de distribución de interés para la asignatura y participación en foros de discusión. En cada módulo el profesor propondrá un tema a debate. La participación en estos foros de discusión tendrán un peso en la evaluación de la asignatura. En caso de que algún alumno no haya participado a lo largo del cuatrimestre lectivo, tendrá como alternativa, durante los meses de verano, el recopilar noticias, extraídas de listas de distribución, relacionadas con los contenidos de la asignatura.

Trabajo individual o en grupo, fuera del horario de clase. Este trabajo propuesto consiste en la elaboración y publicación en un servidor de un documento web cuya realización implica la búsqueda de información en la WWW. Los alumnos pueden elegir entre 3 modalidades de trabajo:

Creación de un catálogo de recursos de información electrónica (páginas web, bases de datos, listas de distribución) de interés para una empresa (ficticia) o colectivo de un determinado sector para satisfacer las posibles necesidadades de información sobre su MICROENTORNO y su MACROENTORNO.

Desarrollo teórico de un tema relacionado con la asignatura a partir de la documentación obtenida en Internet, incluyendo los recursos electrónicos como referencias.

Creacion de un portal de información sobre un tema de libre elección.

3.- LA SESE WEB DE LA ASIGNATURA DE INFORMATICA DE GESTION

Los alumnos acceden a la asignatura a través de una cuenta personal que se les facilita en el momento en que se matriculan. Además también disponen de una cuenta de correo electrónico y de una cuenta para acceder desde casa (actuando en este caso la UIB como proveedor de acceso a Internet).

La página inicial de cada asignatura se divide en 2 áreas, un área de visualización y otra área de trabajo.

a) **Área de visualización:** Esta zona es donde se visualizan los contenidos y coincidirá con lo que verán los alumnos matriculados de la asignatura. Se ofrece un menú que contiene las siguientes opciones:

Guía Práctica del estudiante

Aquí se dispone de un manual que explica cada una de las opciones del menú prinicipal.

Correo

Este servicio del WebCT permite el intercambio de mensajes electrónicos entre los alumnos matriculados en la asignatura y con el profesor.

Tablón de Anuncios

Esta utilidad permite disponer de un espacio virtual donde enviar mensajes que puedan ser vistos por todos los alumnos de la asignatura. Se utiliza por ejemplo para dar un aviso a toda la clase, para suscitar el debate de alguna cuestión (foro de discusión), etc. Los alumnos también lo utilizan entre ellos para manifestar su opinión, preguntarse cosas, solicitar apuntes, etc.

Conversación Electrónica

Con esta opción se accede a la utilidad de chat del WebCT a través de la cual se puede conversar e intercambiar opiniones por escrito y en tiempo real con los participantes.

Calendario

Consiste en un calendario que el profesor puede utilizar para realizar la temporalización de la asignatura, de manera que los alumnos puedan saber qué parte del temario se va a desarrollar cada uno de los días de clase. El calendario constituye también una agenda personal.

Programa

En el programa se incluye la descripción de la asignatura, los objetivos, el índice del temario, las estrategias didácticas y los criterios de evaluación. Constituye el *cuaderno descriptivo de la asignatura.*

Módulos

Aquí se accede a los diversos módulos de la asignatura, los cuales a su vez pueden descomponerse en temas. Cada módulo tiene un menú (ver Fig. 1) desde donde se enlaza a la presentación del módulo, sus objetivos, los contenidos (apuntes en formato hipertextual), test de autoevaluación y enlace al glosario. Para esta asignatura se han añadido 2 botones más: uno para los enunciados de las prácticas supervisadas y otro para especificar qué actividades concretas se han de desarrollar de forma autónoma.

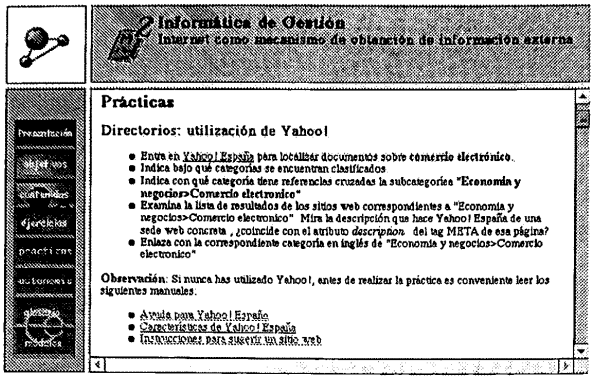


Fig 1. - Estructura de un módulo de la asignatura

Materiales

Los materiales están clasificados en:

- Documentos Electrónicos
- Enlaces a Internet
- Materiales de Referencia
- Biblioteca (enlace vía *telnet*)
- Glosario

Utilidades

La opción incluida por defecto como utilidades es el cambio de clave de acceso a la asignatura. Se ha añadido el enlace al *plug-in* de Acrobat Reader para poder leer documentos en formato pdf y un visor de powerpoint para ejecutar las presentaciones utilizadas en clase por el profesor.

Agora

Agora es la Aplicación de Gestión de la Organización Académica a través de la cual el alumno puede obtener toda su información acadèmica. El profesor puede emplear esta aplicación para calificar y publicar las actas de manera electrónica.

b) **Área de trabajo:** corresponde al área inferior de la pantalla donde se encuentran un menú de opciones y utilidades de la herramienta de teleeducación denominada WebCT (ver Fig. 2). Esta zona no es visible por los alumnos.

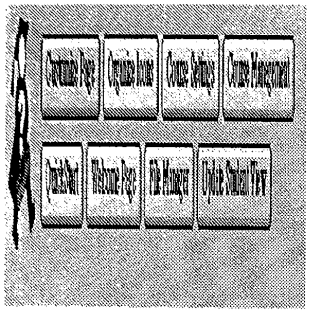


Fig 2. - Menú Principal de la herramienta WebCT para profesores

WebCT es una herramienta para la distribución, desarrollo y control de cursos a distancia sobre Internet. El profesor posee una cuenta de diseñador para poder definir la estructura de la asignatura, decidir qué herramientas *(WebCT Tools)* instala o qué páginas web va a incorporar. Con esa cuenta se tiene acceso a datos de los alumnos de la asignatura *(Student Management)*, como por ejemplo saber las veces que cada alumno ha entrado en la sede web de la asignatura, cuándo lo hizo por primera vez, cuándo fue la última vez que entró, cuál es la dirección de correo electrónico y el *login* de cada alumno, cuántos mensajes han enviado o leído, etc. De las altas de alumnos, profesores y altas de asignaturas no se encarga el profesor sino el *Servei de Càlcul* de la UIB con una cuenta de administrador.

Además WebCT posee utilidades como el calendario, gestión de foros de discusión, correo privado para la asignatura, chats. Posee también un gestor de ficheros *(File Manager)* que permite visualizar el árbol de directorios, crear carpetas, actuar como cliente *ftp* para enviar los materiales al servidor, etc. Otras posibilidades que ofrece WebCT es la de crear exámenes a partir de una base de datos de preguntas que pueden ser de 4 categorías diferentes: elección múltiple (test), emparejamiento de palabras o frases cortas, desarrollo, o respuesta corta. Los comentarios que el profesor desee efectuar en la corrección de cada pregunta podrán ser leídos por los alumnos. En el caso del test, el alumno puede recibir la corrección de manera inmediata. A partir de las respuestas se pueden obtener estadísticas del grupo.

4.- SITUACION ACTUAL Y PERSPECTIVAS

La asignatura de Informática de Gestión se pondrá en funcionamiento en la modalidad *Illes Menors* durante el curso 2000/2001. Será entonces cuando se lleven a cabo las videoconferencia, los desplazamientos periódicos a las islas y la distribución de los cuadernillos impresos. Durante el presente curso, 1999/2000, se ha ofrecido la asignatura solamente a los alumnos con sede en Mallorca, pero ya siguiendo las directrices generales de Campus Extens en cuanto a la reducción de horas lectivas en favor de actividades autónomas, la creación de la sede web de la asignatura incluyendo el cuaderno descriptivo, materiales tales como las transparencias en formato powerpoint, enlaces a Internet, los enunciados de las prácticas con los correspondientes hipervínculos a los materiales necesarios, etc.

En base al cuestionario que se les ha pasado a los alumnos de Campus Extens, el mayor inconveniente del proyecto es la insuficiencia de equipos informáticos para acceder a la web, mientras que la ventaja más señalada es la de disponer de materiales adecuados en la red para poder preparar la asignatura.

LA LEGISLACIÓN SOBRE SEGURIDAD DE DATOS EN INFORMÁTICA DE GESTIÓN. IDEAS PARA UNA GUÍA PARA LA DOCENCIA.

Juan Vicente Oltra Gutiérrez

Escuela Universitaria de Informática
Universidad Politécnica de Valencia
correo-e: jvoltra@omp.upv.es

RESUMEN: En todas las organizaciones empresariales e instituciones públicas existe información de carácter personal, que reside en los sistemas informáticos y recibe un tratamiento automatizado. Esta información, que puede referirse a clientes, empleados, proveedores y colaboradores, está protegida en la legislación española y esto debe formar parte del bagaje de todo futuro profesional informático que pretenda desarrollar su carrera elaborando aplicaciones que manipulen tales datos. En la presente comunicación se presentan ideas para una guía que acerque al alumno la presente legislación, en un marco temporal breve.

1.- INTRODUCCIÓN.

La protección de datos funde el espíritu de protección de las leyes junto con las medidas técnicas y organizativas. Esta aparente *tierra de nadie*, donde ingenieros de organización, abogados e informáticos parecen tener que pugnar, quien tiene la principal competencia es, como veremos en la presente comunicación, el informático, buscando un fin: la humanización de la Informática, la defensa de *la persona*.

A priori parece que el empleo del sentido común aunado con la más elemental deontología profesional bastaría para lograr este fin. No es así[1], al menos en la totalidad de los casos, por lo que el legislador regula aspecto de la informática, ampliando la protección otorgada por la Constitución[2] con otras leyes que la complementan y refuerzan de forma que es obligación tanto ética como legal el cumplirla y hacerla cumplir.

La evolución de la Informática en los últimos años y su integración con las Telecomunicaciones son uno de los fenómenos que más han influido en el vertiginoso cambio social en el que estamos inmersos, con una globalización propiciada por fenómenos como el de Internet.

[1] Aparentemente, con la principal Ley al respecto, la Ley de Protección de Datos de Carácter Personal, nos encontramos con una ley que intenta poner *"puertas al campo"*. La expresión *"si no nos detiene la Etica, si al menos el miedo"* sea tal vez el argumento no escrito más expresivo de la citada ley.

[2] Artículo 18.4: *"la ley limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos"*.

El crecimiento tecnológico ha permitido que el tratamiento se realice de forma masiva y a grandes velocidades, desde cualquier lugar del planeta[3].

Al desarrollar las leyes de protección de datos a través de los reglamentos definidos aparecen, claramente especificadas, una serie de medidas a adoptar en las organizaciones, bien sean públicas o privadas. Este tipo de medidas son de carácter tanto técnico como organizativo, y no tienen otra misión que garantizar los derechos de las personas de las cuales se poseen los datos. Estas medidas a adoptar implican un conocimiento de la realidad informática de la organización, conocimientos que pertenecen al informático *de carrera,* apoyado en conocimientos de gestión para lograr que la integración de las medidas resulte poco gravoso para la organización y su contexto.

2.- LEGISLACIÓN VIGENTE MÁS RELEVANTE.

Dando complemento a la Ley de Leyes, y apoyada posteriormente por convenios y directivas de la Unión Europea, se desarrolla por primera vez el concepto de *protección de datos*, en la Ley Orgánica 5/1992, de 29 de octubre, donde además se crea un organismo independiente que velará por la aplicación de la ley. Este organismo será la Agencia de Protección de Datos. No será esta la ley definitiva y aparecerá posteriormente la Ley Orgánica 15/1999, de 13 de diciembre, que mejorará la anterior y ampliará el ámbito de la protección no solamente al honor e intimidad sino que incluirá las libertades públicas y derechos fundamentales, utilizando para ello el término "privacidad".

La actividad moderna se ha creado una dependencia de los sistemas informáticos que la hacen vulnerable, por falta de seguridad física, por falta de seguridad lógica y por falta de seguridad jurídica, muchas veces producida, por insuficiencia de medios, o utilización de recursos no apropiados, y es ahí donde los reglamentos que desarrollan las leyes de protección de datos, hacen especial hincapié.

Estos reglamentos desarrollan los documentos de seguridad como la herramienta de trabajo fundamental para la regulación de los sistemas de información y su posterior control como sistemas que cumplen y respetan las leyes y los derechos de las personas a las que pretenden servir, que no controlar. El reglamento por excelencia es el Reglamento de medidas de seguridad de los ficheros automatizados que contengan datos de carácter personal, del que más adelante nos ocuparemos.

Otra legislación relevante, como apuntábamos, son convenios y directivas de la Unión Europea e instrucciones que ahondan en temas concretos como los servicios de información sobre solvencia patrimonial y crédito, seguros, accesos a edificios... etc.
La que consideramos esencial y que no debe faltar en el bagaje de un Ingeniero Técnico Informático especializado en la creación de aplicaciones de gestión (no olvidemos que la practica mayoría manipulan datos personales) son los apuntados:

- Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.
- Reglamento De Medidas De Seguridad De Los Ficheros Automatizados Que Contengan Datos De Carácter Personal

[3] *"Empieza a no contar el tiempo ni el espacio (...). La Informática (...) puede llegar a convertirse en un instrumento de presión y control de masas"* Manual de Derecho Informático. M.A.Davara Rodriguez. Ed. Aranzadi, 1997, pg.23

3. ASPECTOS BÁSICOS DE LA LEY ORGÁNICA DE PROTECCIÓN DE DATOS DE CARÁCTER PERSONAL.

Inicialmente se trató de proteger siempre que el tratamiento de los datos fuera susceptible de ser manipulado no solo por personal no autorizado, sino de forma automatizada. Sin embargo, las nuevas Directivas, y en nuestro caso la Ley Orgánica 15/1999 de 13 de diciembre, que sustituye y deroga a la LORTAD (Ley Orgánica de Regulación del Tratamiento Automatizado de Datos de carácter personal) 5/1992 de 29 de octubre, ya hacen hincapié en que no solo se protegerán los ficheros automatizados, sino todos aquellos ficheros con datos de carácter personal y los tratamientos, sean o no automatizados.

Por lo tanto, tenemos por un lado los datos que referencian a personas, por otro lado un tratamiento de dichos datos para generar información que identifique de manera unívoca a las personas, y por último el acceso o utilización fraudulenta de dicha información. Estos elementos son los que nos llevan a la creación del concepto de "protección de datos[4]"

El ámbito de aplicación de la ley son los datos de carácter personal en soporte físico que los haga susceptibles de tratamiento, y a toda modalidad de uso posterior de estos datos por los sectores público y privado[5].

Hace la Ley especial hincapié, y por tanto es preciso hacer notar esto al alumno, en la seguridad de las instalaciones donde se encuentran los sistemas que almacenan y procesan los datos. Por ello el responsable del fichero y, en su caso, el encargado del tratamiento deberán adoptar las medidas técnicas y organizativas necesarias para[6]:

- garantizar la seguridad de los datos personales
- evitar su alteración, pérdida, tratamiento o acceso no autorizado teniendo en cuenta el estado de la tecnología, la naturaleza de los datos almacenados y los riesgos a que están expuestos.

a) Principios de la protección y momentos del tratamiento de datos.

El discente debe tener presente los principios de la protección de datos, como piedra basal sobre la que asentar el resto de conocimientos. Estos, según el Ministro de Justicia son de tres tipos:

1) principios de carácter general: de proporcionalidad, de vinculación al fin, de exactitud, de transparencia y el secreto del responsable

2) principios singulares: de recogida, de tratamiento y de cesión

---

[4] Se protege a la persona titular y no al dato en si mismo.

[5] Especialmente se protegen los datos relativos a ideología, religión y creencias, donde nadie puede ser obligado a declarar sobre ello. También se protegen los de afiliación sindical. Todos estos datos se solicitarán con consentimiento expreso y por escrito. Como excepción encontramos los ficheros de partidos políticos, sindicatos, iglesias, asociaciones y fundaciones con finalidad política, filosófica, religiosa o sindical, en cuanto a los datos de sus afiliados o miembros. Los referidos a raza, salud y vida sexual se podrán pedir con consentimiento expreso o si lo marca alguna Ley; y los datos referidos a infracciones solo podrán obtenerse por las Administraciones Públicas competentes ejerciendo sus funciones.

[6] Para ello la Ley se apoya en el Real Decreto 994/1999, sobre medidas de seguridad.

3) principios especiales: afectan a datos sensibles, ideología, religión, creencia, raza, salud y vida sexual.

Estos principios, de cara a articular una docencia coherente, deben completarse con elementos más contrastables dado su perfil, que serían:

Principio de pertinencia de los datos, de acuerdo con el ámbito y la finalidad para las que se hayan obtenido.

Principio de exactitud y actualización, de forma que los datos reflejen con veracidad la situación real del titular, complementados con los principios de congruencia y de racionalidad, en relación a la necesidad de garantizar que los datos no pueden ser tratados, ni utilizado el resultado de su tratamiento en caso de que ya lo hubieran sido, nada más que en aquellos casos en que sea totalmente necesario y adecuado a la finalidad para la que fueron tomados.

Principio de consentimiento[7] del interesado para el tratamiento automatizado.

También resultaría poco coherente no establecer una distinción temporal, según el momento por el que "transitan" esos datos. Los momentos del tratamiento de datos serían:

- La toma de datos
- El tratamiento automatizado de datos, incluyendo el que pueden ser cruzados y relacionados en forma automática junto con otros datos
- La utilización y, si se da el caso, cesión[8] de datos.

b) Derechos de las personas

De igual manera, procede explicar, aun de manera sucinta, al alumno, cuales son los derechos que deben ser respetados:

- Derecho de autodeterminación
- Derecho de información y acceso del titular
- Derechos de rectificación y cancelación
- Derechos de impugnación
- Derecho a exigir responsabilidad por daño
- Otros: consulta al Registro General de protección de datos.

---

[7] Excepto:
- cuando una Ley lo prevea
- cuando sean datos de fuentes accesibles al público
- cuando exista una libre aceptación de relación jurídica
- cuando los datos se requieran por parte de Fiscales, Tribunales, Defensor del Pueblo, Tribunal de Cuentas y órganos autonómicos con funciones análogas.
- en las actividades propias de las Administraciones Públicas descritas en el artículo 21 de la Ley Orgánica 15/1999.
- en los datos referidos a la salud en caso de urgencias o estudios epidemiológicos.

[8] Las garantías en el caso de cesión, no son necesarias si a los datos se les aplica un proceso de disociación, proceso por el cual los datos no identifican a personas en forma alguna.

Destacando por su singular importancia, los derechos de acceso, rectificación y cancelación.

c) Exclusiones:

Obviamente, interesa delimitar que campo es aquel donde los alumnos tendrán que emplear la legislación, siendo por lo tanto necesario que sepan cuales son las excepciones. Se excluyen del ámbito de aplicación los ficheros personales, de materias clasificadas, sobre investigación del terrorismo y de formas graves de delincuencia organizada -aunque de los cuales se notificará su existencia a la Agencia de Protección de Datos.-

Otros ficheros que se rigen por sus disposiciones específicas son: ficheros electorales, ficheros con fines estadísticos, informes personales de los militares profesionales, el Registro Civil, el Registro Central de Penados y Rebeldes y una novedad con respecto a la anterior Ley, los ficheros de imágenes: la videovigilancia.

4. ASPECTOS BÁSICOS DEL REGLAMENTO DE MEDIDAS DE SEGURIDAD DE LOS FICHEROS AUTOMATIZADOS QUE CONTENGAN DATOS DE CARÁCTER PERSONAL

El cumplimiento del reglamento de seguridad no es tan sólo una obligación legal, sino que, a su vez, redunda en beneficio del informático y de la organización donde desempeña su trabajo, pues genera unas necesidades de organización saludables para la institución en que se aplique.

Los objetivos principales son la confidencialidad de los datos, su integridad y la disponibilidad de los mismos en el ámbito adecuado.

El reglamento de seguridad crea tres niveles de seguridad en función de la información que almacenen los ficheros. Estos tres niveles, denominados básico, medio y alto, podemos desglosarlos de la siguiente manera en una primera aproximación:

- Nivel Básico:
    - Todos los ficheros con datos personales
- Nivel medio:
    - Infracciones Administrativas y Penales
    - Hacienda Pública (Administración Tributaria)
    - Servicios Financieros
    - Evaluación de la personalidad del indivíduo
- Nivel Alto:
    - Ideología[9], Religión y Creencias
    - Origen Racial, Vida Sexual y Salud
    - Policiales

[9] Recordemos que este reglamento apareció para complementar la extinta LORTAD. En ella, los datos sindicales no eran mencionados, no así en la nueva Ley, que los coloca en le rango que merecen, siendo pues de necesaria aplicación en su caso a pesar de no ser expresamente referidos por el reglamento.

Se trata de niveles "envolventes", p.e.: el nivel alto incluye las medidas necesarias para los niveles medio y básico. Dado que las medidas citadas en el nivel básico deberán ser cumplidas siempre, y que estadísticamente, serán estos ficheros los más comunes, parece oportuno centrar la atención del alumno en ellas, sin desdeñar las complementarias para los niveles de seguridad más elevados.

El elemento más importante que aparece es el *documento de seguridad*, que debe mantenerse permanentemente actualizado y revisado siempre y cuando se produzcan cambios relevantes en el sistema de información o en la organización del mismo. En él encontramos información tan significativa como los recursos protegidos, las medidas, normas, procedimientos, reglas y estándares encaminados a garantizar el nivel de seguridad exigido por el reglamento, las funciones y obligaciones del personal, la estructura de los ficheros y la descripción del Sistema de Información, así como los procedimientos de notificación, gestión y respuesta ante las incidencias, y los procedimientos de realización de copias de respaldo[10] y recuperación de los datos.

Como vemos, unos contenidos prolijos[11] que sirven al tiempo de guía para la correcta gestión de los datos.

En el nivel medio resulta destacable la creación de la figura del responsable de seguridad, que por perfil coincide con el responsable de informática o alguien de su equipo. Se hace mención de dos registros: el de incidencias, viejo conocido de los informáticos como cuaderno de bitácora de un S.I. y el novedoso registro de entrada y salida de soportes.

En el nivel alto se observan unas medidas exigidas encaminadas a una intensa actividad controladora de los movimientos de datos.

a) Adaptación al documento de seguridad

Prácticamente todas las medidas que se pueden disponer quedan definidas en el documento de seguridad, sin embargo aparecen al margen de las medidas organizativas y técnicas otras que podríamos denominar *de sentido común*[12] que se solucionan implicandose directamente en la formación de los usuarios.

b) Implantación y su impacto

Se realizará de manera progresiva, yendo del nivel básico a los superiores. De esta manera, los cambios organizativos se verán como algo positivo que no supone apenas esfuerzo llevar a cabo y que sufre evoluciones para mejorar. Se podrían detectar dos niveles de impacto: técnico (donde se precisa el apoyo de los conocimientos de otras asignaturas *tecnológicas*) y personal, donde se puede encontrar ciertas reticencias para colaborar (donde se precisaría ayuda de conocimientos del campo *Comportamiento Organizacional*)

[10] Como dato curioso, el reglamento fija una temporalidad mínima exigible a las copias de seguridad, algo que parece escapar de la *ley* para caer en la *gestión*.

[11] ¡y sólo citamos el nivel básico!

[12] No obtener datos para asuntos particulares, no emplear soportes magnéticos de cuya procedencia no se esté seguro, no hacer mal uso de las aperturas y cierres de sesiones de trabajo...

5. CONCLUSIONES

Hay que tener en cuenta que no sólo los problemas legales y técnicos, sino también organizativos, en cuanto a cambio en la operativa diaria y en la forma de actuar de las organizaciones para cumplir esta legislación.

El alumno no debe infravalorar el impacto de este problema a la hora de establecer prioridades. La realización del diagnóstico de adecuación al mismo conlleva también problemáticas adicionales, con el fin de evitar las sanciones en un entorno tan sujeto a necesidades de gestión de configuración y de gestión del cambio como son los sistemas de información.

6. BIBLIOGRAFÍA

BARRIUSO, C. Interacción del derecho y la informática. Dykinson. Madrid, 1996 ✸ DAVARA, M.A. Manual de derecho informático. Aranzadi. Pamplona, 1997. ✸ GRIMALT, P. La responsabilidad civil en el tratamiento automatizado de datos personales. Comares, Granada, 1999 ✸ Herran, A. La violación de la intimidad en la protección de datos personales. Dykinson. Madrid, 1999 ✸ OLTRA, J.V. Ética e informática, protección de datos y otros aspectos. SPUPV, Valencia 2000; Impacto legal de la informática en las organizaciones. SPUPV, Valencia, 1999 ✸ VVAA. Protección y seguridad de datos (Vol I y II). Cuatrecasas, Madrid 2000

ACERCAMIENTO DEL DERECHO Y LA DEONTOLOGÍA PROFESIONAL AL ALUMNO DE INGENIERÍA TÉCNICA INFORMÁTICA. UN CASO: ASPECTOS LEGALES DE LA INFORMÁTICA.

Juan Vicente Oltra Gutiérrez

Escuela Universitaria de Informática
Universidad Politécnica de Valencia
correo-e: jvoltra@omp.upv.es

RESUMEN: En la presente comunicación se propone un modelo para la docencia del derecho y la ética informáticas sobre discentes que serán Ingenieros Técnicos Informáticos, un reto por la previsible animadversión del alumno a una asignatura con organizadores tan dispares a sus otras materias ("tecnológicas"). Se estudia el caso particular de la asignatura "Aspectos Legales de la Informática" impartida en los últimos años en la Escuela Universitaria de Informática (E.U.I.) Universidad Politécnica de Valencia (U.P.V.)

1. INTRODUCCIÓN

Tradicionalmente, los estudios de informática fueron enfocados desde un prisma puramente tecnológico, ausente[13] en gran medida en ellos un entronque con el entorno: el lugar donde van a ejercer su profesión, sea una empresa privada o una administración pública.

La puesta en marcha en el curso 93/94 de los nuevos planes de estudio en la E.U.I., permitió dar aires nuevos a la titulación, respetando los conocimientos puramente informáticos pero abriendo la puerta a la adquisición por parte del alumno de conocimientos de materias antes vedadas, bien como asignaturas optativas o de libre elección, bien como temas dedicados en asignaturas troncales.

Este puede ser el caso de la asignatura "Aspectos Legales de la Informática" (aparecida en el curso 97/98). En esta asignatura se trata de acercar al futuro profesional los elementos básicos para facilitarle un correcto desarrollo dentro de los cauces legales. Siempre con la idea de incorporar al alumno, una vez egresado, en un estado de derecho[14], avanzamos un paso al frente recordando al viejo refranero castellano cuando nos dice "*hecha la ley, hecha la trampa*" o, en otras palabras, respetar el texto de la ley no sirve para nada si se conculca su espíritu. Y la mejor garantía de que no se haga un uso torticero de la ley es incorporar la guía natural para evitar desmanes: la ética que, de esta manera, adquiere carta de naturaleza en la asignatura.

[13] O al menos con escasa relevancia
[14] Que de soporte a una sociedad moderna y progresista

El reto que supone la asignatura podría resumirse en cómo lograr el interés por parte de un alumno a priori hostil a este tipo de conocimientos, a pesar de su conciencia[15] de la necesidad de añadir estos a su bagaje.

No se trata de conseguir una asignatura "fácil", (en argot *maría*) sino de convertir en una más una asignatura humanista *-poesía-* entre un aluvión de asignaturas tecnológicas *-libro de teoremas*[16]-.

2. LA ASIGNATURA "ASPECTOS LEGALES DE LA INFORMÁTICA" (A.L.I.) EN LOS PLANES DE ESTUDIO DE LA E.U.I. DE LA U.P.V.

Dentro de la estructura de la titulación, aparecen tres tipos de asignaturas: obligatorias, optativas y de libre elección. La asignatura que nos ocupa es de la categoría "libre elección". Los créditos de libre configuración permiten al alumno una gran flexibilidad[17] para configurar su curriculum.

Su carga docente asignada es de 4,5 créditos (3 teóricos y 1,5 prácticos). Por tanto la carga lectiva será de 3 horas semanales, correspondiendo 2 horas a teoría y 2 horas a prácticas en semanas alternas. Resultan en total, considerando cuatrimestres de 15 semanas 30 horas de teoría y 15 de prácticas, una dotación exigua para unas materias tan extensas, problema al que se ha intentado dar solución, expuesta en el punto 4.

Una vez ya tenemos establecida la extensión (continente) pasemos a hablar del diseño del programa (contenidos). Para concretarlos, se estableció una serie de criterios (veremos su aplicación en el punto 4).

- La significación epistemológica debe ser ajustada y abarcar los conceptos fundamentales de la disciplina, recogiendo desde los enfoques globales hasta los procedimientos operativos[18]. Los criterios academicistas/tecnológicos deben equilibrarse[19].

- El criterio curricular debe contemplar la situación del alumno, su formación reducida sobre la materia (siendo por tanto sus organizadores previos un tanto rudimentarios), dedicando un tiempo adicional a reforzarlos, para que la enseñanza de nuevas informaciones no estén carentes de sentido y no supongan un rechazo. En resumen: relacionar con lo ya visto y es más, preparar para lo que se va a ver en otras asignaturas.

- Los criterios de amplitud y profundidad deben ser adecuados y estar en consonancia con el tiempo disponible dominando los conceptos básicos. Sería un enfoque pedagógico[20], la formación no puede ser aprovechada sino por aquel que está adaptado. Lo importante aquí pasa de nuevo por la selección de materias de interés para el desarrollo curricular del alumno y la distribución que se hace de ese tiempo entre ellos.

[15] De esta consciencia se da cuenta en el punto 2.
[16] Evidentemente, sin trocar a la poesía en un teoréma
[17] Esta flexibilidad va acompañada de mayores dificultades. En el caso de tratarse de una asignatura, aparecen pequeñas diferencias con sus "hermanas mayores", como por ejemplo la que se da al no fijar el centro una fecha de examen como para las obligatorias u optativas, quedando esta tarea pendiente del acuerdo conseguido entre los alumnos, lo cual no suele ser fácil. Pero dejemos la mera anécdota para calibrar lo esencial.
[18] p.e. desde los principios generales de protección de datos personales, hasta medidas técnicas concretas a aplicar para el cumplimiento del reglamento de medidas de seguridad (994/1999)
[19] Ver apartado de métodos (punto 4)
[20] Barbero, M. "Preparar profesionales y directivos de TIC". Convergencia IT (órgano de Dintel), Vol nº1. 1999. Madrid.

3. EL DERECHO Y LA ÉTICA EN EL DESARROLLO PROFESIONAL DE UN I.T.I.

Los alumnos que cursan la asignatura han tenido contactos esporádicos en asignaturas previas con conceptos relacionados con el mundo del derecho[21], pero adolecen de una visión amplia que les permita situarse en una actitud óptima para recibir los conocimientos que se les trata de transmitir. Con respecto a la ética, la situación es mucho más precaria[22]. Esta situación la dan ellos mismos por sentada. Una encuesta que se pasa el primer día de curso da en una gran mayoría de respuestas en desacuerdo con la pregunta: "*Creo que tengo bases suficientes para entender sin ningún problema un curso sobre aspectos legales de la informática, incluyendo ética y leyes*".

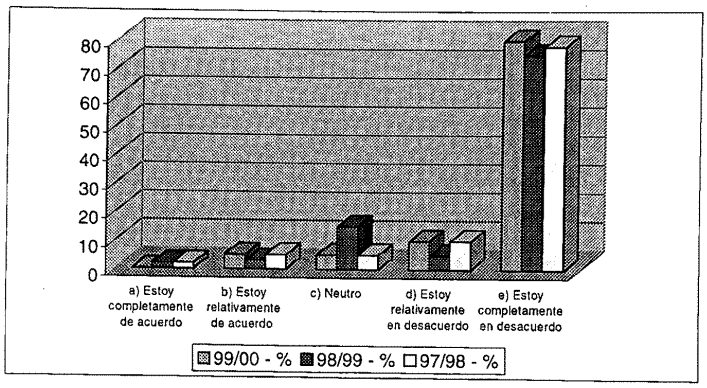


Gráfico 1. Respuestas a encuesta (1/2)

Y sin embargo, reconocen la gran importancia que supone en su formación estos conocimientos, lo que se observa en la proporción inversa recogida para la afirmación[23] "*El conocimiento de unos mínimos fundamentos legales y éticos son necesarios para el desarrollo de mi profesión futura*".

[21] trámites para crear una empresa, mínimos para desenvolverse en una empresa virtual, ...

[22] Esto es parte de un problema común a todas las titulaciones técnicas: el alejamiento de la sociedad, de los aspectos humanísticos de ésta, y empieza a querer solucionarse, siendo pionera en cierta medida la experiencia que supone la presente asignatura.

[23] Obviamente, hay que tener en cuenta que se trata de una asignatura no obligatoria, por lo tanto todos los alumnos que la cursa, y por ende, que rellenan la encuesta, se han matriculado libremente.

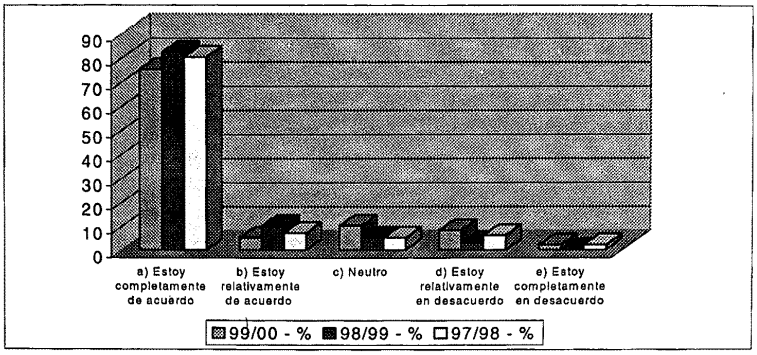


Gráfico 2. Respuestas a encuesta (2/2)

Aún así, el objetivo marcado para un público de informáticos no puede pasar por hacer un estudio exhaustivo de la legislación existente o de la ética como disciplina, sino conocer, con respecto al derecho, por una parte las leyes más importantes que como informáticos han de marcar su quehacer profesional, y por otra parte, en los casos concretos en que así se requiera, dar a conocer los medios técnicos necesarios para la aplicación de dichas leyes. Con respecto a la ética, se trata de enfocar la responsabilidad que tiene su fin en la empresa, desde su origen en el individuo, por lo que se les trata de procurar las herramientas básicas con que puedan construir su propio marco ético. Ética[24] entendida como la ciencia del comportamiento humano, no sólo ligada al concepto de ley, sino que como el profesor De Miguel[25] nos recuerda, en muchos casos se superpone.

4. DIDÁCTICA / DINÁMICA DE LAS CLASES. MÉTODOS

El proceso de enseñanza ha de ser en principio eficaz y también eficiente. Con esta posición de salida, debemos comenzar por identificar que es lo que deben y que pueden aprender los alumnos, para posteriormente, centrarnos en cual es el proceso de aprendizaje y las estrategias a seguir[26].

Tradicionalmente, el alumno memorizaba el conocimiento y el profesor comprobaba el grado de similitud entre la repetición y el original. En este modelo el docente trata de enseñar lo que sabe[27].

Hoy, se describe el aprendizaje como un proceso que implica la adquisición o reorganización de las estructuras mentales mediante el procesamiento[28] de la información que se recibe. La preocupación del profesor se orienta a enseñar lo que el alumno puede asimilar, los

[24] CORTINA, A. Ética de la empresa. Trotta. Madrid, 1998

[25] De MIGUEL, E. Introducción a la gestión (management). (Vol I), SPUPV. Valencia, 1999.

[26] "Descuidar la atención a los métodos con la intención de dedicarse a los contenidos es un falso camino; por que los métodos – sin perder su función instrumental – pueden impedir, si no son adecuados, la transmisión de cualquier contenido", de Rafael Gómez Pérez en el prologo de "Los Métodos en la Enseñanza Universitaria" (Pujol y Fons, EUNSA. Pamplona, 1981)

[27] GOOD, T. "Psicología educativa contemporánea". McGraw-Hill. México 1996.

[28] Procesar significa *clasificar o "archivar" conocimientos y crear conexiones cruzadas entre los conocimientos*

conocimientos que están dentro de su "zona de desarrollo próximo"[29]. Se busca que la persona aprenda a pensar, a reflexionar y se prepare para aprender toda la vida.

Dentro de este "cognoscitivismo" se siguen en distintos momentos de la docencia de la asignatura dos planteamientos: el expositivo y el experimental. No se debe abandonar el aprendizaje de memoria, ya que se basa en la conexión de conocimientos y para unirlos es preciso tenerlos.

El sistema es más lento que la simple enumeración de conocimientos, por lo que se debe renunciar a ciertos contenidos para centrarnos en lo fundamental. Para esta selección nos apoyamos en un estudio del programa de las otras asignaturas que cursa el alumno, no sólo para evitar redundancias (escasas), sino para poder establecer conexiones adecuadas entre unos y otros conocimientos.

Un esquema del programa a seguir, para la parte teórica de la asignatura, sería el siguiente[30]:

I. Introducción (1. Presentación de la asignatura-1-);

II. Ética (2. Ética empresarial -2-; 3. Ética informática -2 -);

III. Derecho informático (4.Introducción al derecho informático -2-; 5. Protección de datos -5-; 6. Protección jurídica del software y las BBDD -4-; 7. Contratación informática. Contratación electrónica -2-; 8. El delito informático -5-; 9. Otros 'Dinero de plástico,...' -2-);

IV. Informática y trabajo (10. Informática y mundo laboral -2-; 11. Informática y ley en la administración pública -1-; 12. Peritajes y arbitrajes informáticos -2-)

a) Método expositivo.

Se trataría del método empleado en la parte teórica de la asignatura. El profesor actúa como transmisor de un cuerpo fijo de contenido al alumno (receptor), preocupándose por presentar el material de forma que aliente a los estudiantes a darle sentido, relacionándolo con lo que ya conocen. Se les establece relaciones de los contenidos de la asignatura con otras ya cursadas invitándoseles a cursar asignaturas optativas y de libre elección donde pueden profundizar sobre determinados puntos, etc.

Conseguir el interés en una materia que, por el contraste ofrecido con el resto de las asignaturas puede crear un estado inicial de rechazo, supone un reto que a ser superado con herramientas y con imaginación. A la exposición con el apoyo de pizarra y presentaciones asistidas por ordenador se suman los vídeos, que permiten establecer un diálogo o una discusión dirigida y otras muchas posibilidades: por poner un ejemplo, resulta interesante para el alumno apoyar la exposición del tema referente a delitos informáticos con exhibiciones de páginas web o correos en grupos de noticias donde se viole la ley[31].

---

[29] GOOD, T. "Psicología educativa contemporánea". McGraw-Hill. México 1996

[30] Para la implementación de este temario se siguieron diversos textos que figuran en la bibliografía de la presente comunicación. Se sigue el formato Unidad Temática (tema-duración-, tema-duración-,...)

[31] En este orden de cosas, para la realización de consultas (tutorías), se ha dispuesto de un mecanismo adicional al tradicional: el empleo de chat. Esto es algo que, de nuevo, atrae la atención del alumno, máxime cuando le encuentra un sentido práctico a una herramienta considerada por él como meramente lúdica.

El papel principal lo desarrolla el profesor que selecciona el contenido a enseñar y evalúa el aprendizaje, buscando en todo momento una mayor participación por parte del alumno.

b) Método experimental.

Se trataría del método empleado en la parte práctica de la asignatura. Se permite a los estudiantes que usen los conocimientos mientras llevan a cabo una investigación, solucionando problemas, tomando decisiones... razonando en suma. El profesor da oportunidades para que los estudiantes aprendan por medio del "descubrimiento guiado", que hagan suyo el conocimiento nuevo. Se busca crear un conocimiento generado, para que el alumno pueda usarlo en situaciones nuevas. Se trata de evitar ese conocimiento sin posibilidad de aplicación general, que el estudiante es capaz de identificar y aplicar si se le pregunta directamente, o se encuentra en situaciones muy similares a las de la clase, pero es incapaz de acceder a él y aplicarlo en las múltiples situaciones de la vida. Partiendo de la base de una necesaria base de conocimientos, lo que permite "activar" la máquina, la idea madre es que "los alumnos sólo aprenden, es decir, establecen nuevas conexiones, cuando se equivocan"[32]; pues cuando aciertan lo que hacen es reforzar las conexiones existentes. En este caso, la responsabilidad del aprendizaje la tiene el alumno.

Las herramientas posibles son múltiples: irían desde la simulación hasta la discusión dirigida. El método "Rol" supone una variante muy apreciada por los alumnos (siguiendo la máxima *jugando aprendo*). Una práctica que suele tener un grado de aceptación muy alto por parte del alumnado es la búsqueda de recursos en Internet (páginas web, grupos de noticias, foros de charla por lista de distribución o canales de chat en la red irc-hispano[33]).

Las catorce horas de prácticas se dividen en siete sesiones como queda:

1. Introducción (Búsqueda guiada de recursos)
2. Ética (Discusión dirigida sobre un caso real)
3. Protección de datos (Método del caso, variante "Rol")
4. Delito informático (Búsqueda de recursos en Internet)
5. Protección jurídica del Sw (Discusión dirigida sobre un caso real)
6. Mundo laboral (Simulación -Creación empresa-)
7. Peritajes y arbitrajes (Elaboración de un peritaje).

5. CONCLUSIONES Y PERSPECTIVAS

Es bueno que el alumno reciba conocimientos "humanísticos", eso es algo que desde instancias superiores (Informe Bricall) se reconoce. Pero el cambio puede resultar muy duro, traumático para el alumno. Si empleamos herramientas que permitan *convertir* el vehículo de transmisión de conocimientos en algo habitual, conocido para el discente, y al tiempo, buscamos captar el

[32] BENEJAM, P. ¿Cómo aprenden los alumnos universitarios?. Instituto de Ciencias de la Educación de la Universidad Politécnica de Valencia. Valencia, 1997.

[33] Los recursos encontrados después se clasifican por categorías.

interés del alumno, no tan sólo buscando el saber por el saber, sino caminando hacia el saber hacer, mostrándole su utilidad en el mundo profesional y enseñándole a razonar con un sustrato básico, no sólo enriqueceremos su formación profesional sino que ayudaremos en su formación personal.

Ahondando en esa *sensación de proximidad* citada ut supra e intentando también llegar no sólo de nuevas formas a un mismo "público" sino ampliar este, en el curso académico 00/01 esta previsto impartir la asignatura no sólo en modo presencial, sino también en teleformación. El reto esta vez es transformar las herramientas citadas previamente en sus respectivas variantes *a distancia*.

7. BIBLIOGRAFÍA BÁSICA

BARBERO, M.. "Preparar profesionales y directivos de TIC". Convergencia IT (Dintel), Vol nº1.Madrid, 1999 ✸ BARRIUSO, C. Interacción del derecho y la informática. Dykinson. Madrid, 1996 ✸ BENEJAM, P. ¿Cómo aprenden los alumnos universitarios?. Instituto de Ciencias de la Educación de la Universidad Politécnica de Valencia. Valencia, 1997. ✸ CORTINA, A. Ética de la empresa. Trotta. Madrid, 1998 ✸ DAVARA, M.A. Manual de derecho informático. Aranzadi. Pamplona, 1997. ✸ GOOD, T. Psicología educativa contemporánea. McGraw-Hill. México 1996. ✸ MATEO, R. Medios de comunicación educativa. SPUPV. Valencia, 1982 ✸ MESO, K. Y ÁLVAREZ, B. Derecho en Internet. Anaya, Madrid, 1998 ✸ OLTRA, J.V. Ética e informática, protección de datos y otros aspectos. SPUPV, Valencia 2000; Impacto legal de la informática en las organizaciones. SPUPV, Valencia, 1999 ✸ ORTIZ, J.M. La hora de la ética empresarial. Mc Graw Hill. Madrid, 1995 ✸ PUJOL, J. y FONS, J.L. Los Métodos en la Enseñanza Universitaria. EUNSA. Pamplona, 1981

UNA EXPERIENCIA EN LA APLICACIÓN DE LA INFORMATICA A LA ENSEÑANZA DE CALCULO FINANCIERO

Pedro Carvajal Molina[1], Susana Blanco García[2], Rocío Gallego Losada[1], Raquel Garrido Avia[1], Piedad Tolmos Rodriguez-Piñero[1]

[1]*Dpto. Economía Financiera y Contabilidad, Univ. Rey Juan Carlos*

e-mail: carvajal@poseidon.fcjs.urjc.es

[2]*Dpto. Economía Financiera y Contabilidad I, Univ. Complutense Madrid.*

RESUMEN: El objetivo de esta ponencia, como su nombre indica, es mostrar nuestra experiencia sobre la utilización de medios informáticos en la enseñanza universitaria de **Matemáticas Financieras** y las posibilidades de ahorro de tiempo y potenciación de los aspectos conceptuales, a nuestro juicio los realmente importantes, que este uso permite al reducir los procesos meramente calculísticos o de operatoria mecánica.

1. INTRODUCCION

Fruto de la colaboración entre la Universidad Rey Juan Carlos y el Instituto Madrileño para la formación, surgió un curso titulado "Informática y finanzas aplicadas a las Pyme" que tuve la fortuna de coordinar.

En dicho curso estabamos condicionados por la presencia de veinte estudiantes de las mas diversas formaciones previas: Derecho, Economía, Políticas, Ciencias de la Información, Relaciones Laborales... pero todos con un afán común: reciclarse en los conocimientos básicos demandados por las Pyme para sus departamentos de administración.

Doscientas horas y un programa que incluía:

a) Microsoft Office 97.

b) Herramientas de redes y comunicación.

c) Creación de Paginas Web.

d) Seguridad del Sistema de Información.

e) Calculo Financiero.

f) Fuentes de Financiación.

g) Gestión de nóminas.

h) Impuestos.

i) Contabilidad

Como puede verse poco tiempo para tantos contenidos.

Centrándonos en el apartado de Cálculo financiero, a él solo le podíamos dedicar una semana, veinte horas, en la que nos propusimos dar a los participantes una formación lo mas similar posible a la que obtiene uno de nuestros diplomados en empresariales, el problema es que con ellos disponemos de todo un cuatrimestre, con cinco horas a la semana y así y todo nos quedamos cortos de tiempo.

Nuestra única solución era cambiar la tradicional clase de: explicación de conceptos en la pizarra, con la posterior presentación de formulas y realización de problemas por parte de los alumnos mediante el uso de calculadora, para mediante la repetición automatizar los procedimientos de cálculo; por la explicación y realización directa de los contenidos mediante el ordenador. Para ello elegimos como programas básicos dos: Scientific y Excel. El primero por su claridad expresiva, y el segundo por su facilidad de manejo y por ser uno de los más extendidos en las pequeñas y medianas empresas de nuestro país.

Antes de continuar una puntualización, nosotros, con todos los respetos, no nos consideramos una academia que enseñe a manejar Excel, es decir nuestro interés no se centraba en explicar el comando "VNA (tasa;valor 1;valor 2;....)", nuestro desafío, en este caso concreto, era mostrar el significado de la actualización de capitales y la procedencia de las expresiones que nos permitían llegar hasta él, independientemente del paquete informático utilizado.

2. DESARROLLO DE LOS CONTENIDOS

Básicamente los contenidos que quisimos abarcar con nuestro programa de operaciones financieras incluían:

a) Fundamentos financieros: capital financiero, equivalencia financiera, leyes financieras, magnitudes derivadas, operaciones financieras, etc.

b) Leyes de capitalización simple: concepto, cálculo de cada una de sus variables, tantos equivalentes.

c) Descuento Simple: concepto, descuento simple comercial, descuento simple racional.

d) Capitalización compuesta: concepto, cálculo de cada una de sus variables, tantos equivalentes en capitalización compuesta, tantos efectivos, TAE, nominales

e) Descuento compuesto: a tanto de capitalización, a tanto de descuento.

f) Aplicación a productos financieros: cuentas corrientes, libretas de ahorro, letras, pagarés, cheques, transferencias, recibos bancarios...

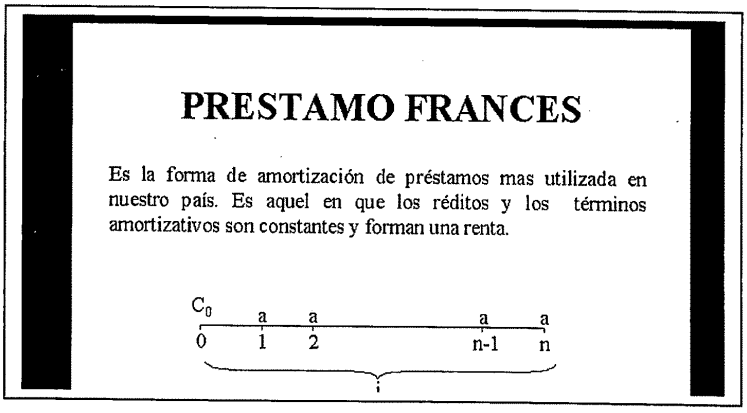
g) Rentas: concepto, caso general, valoración, estudio detallada de los diferentes tipos de rentas. Aplicación a los alquileres, valor final de una nómina, rentas vitalicias, planes de pensiones, valoración de bonos...

h) Préstamos: francés, americano, cuota de amortización constante, cuadros de amortización de préstamo, aplicaciones como los prestamos a tipo fijo, variable o mixto, análisis de solvencia, garantías, carencias, comisiones y plazos

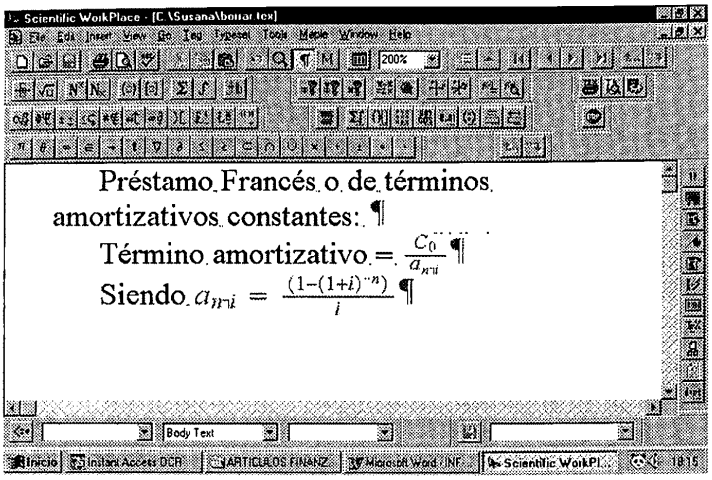
Y como los anteriores otros cuatro temas que podríamos enumerar, pues todos los desarrollamos apoyándonos en los dos programas anteriormente indicados, aunque preferimos detenernos aquí para no hacer demasiado tediosa su indicación. Tomemos mejor este último tema indicado, el de los préstamos y utilizando los archivos generados para el curso mostremos, en algunas pantallas, como desarrollamos el tema.

3. PRESTAMOS.

En primer lugar utilizamos unas cuantas pantallas realizadas en Power Point para introducir las cuestiones a las que nos íbamos refiriendo



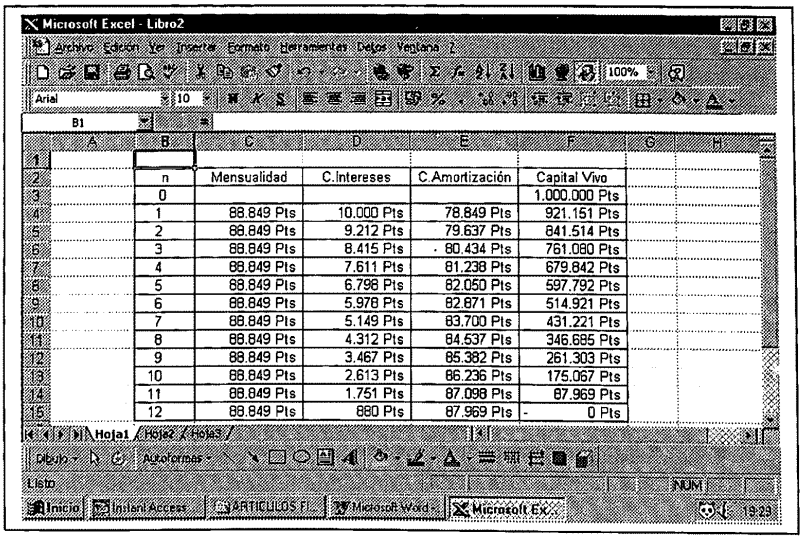
Después indicabamos la formulación para que pudieran introducirla los componentes del curso dentro de su programa Scientific , y así poder ir practicando con diferentes tipos de interés, variando el principal de los prestamos o el número de periodos en que se querían amortizar.



Una vez expresadas, y entendidas, todas las formulas usadas en cada tipo de préstamo, dejábamos un tiempo para que pudieran "jugar" con ellas y sorprenderse de los enormes cambios que provoca en la cantidad que debemos pagar en concepto de mensualidad, o anualidad un simple cambio de medio punto en los tipos de interés.

Finalmente nos pasábamos a Excel para poder crear de forma sencilla cuadros de amortización de préstamos, esas expresivas tablas que nos entrega el banco a la hora de formalizar una operación de este tipo, donde se nos indica las cantidades que hemos de pagar cada mes, cuanto va destinado a cubrir intereses, a amortizar, vemos la evolución de la deuda pendiente etc.

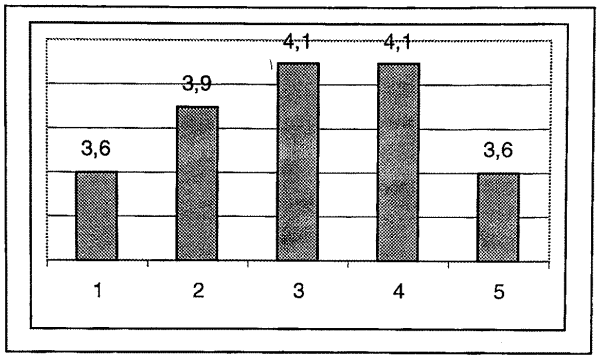
En el ejemplo siguiente les propusimos crear el cuadro de amortización de un préstamo de un millón de pesetas, amortizable en un año mediante pagos mensuales constantes y tipo de interés también constante del 12% nominal anual.



n	Mensualidad	C.Intereses	C.Amortización	Capital Vivo
0				1.000.000 Pts
1	88.849 Pts	10.000 Pts	78.849 Pts	921.151 Pts
2	88.849 Pts	9.212 Pts	79.637 Pts	841.514 Pts
3	88.849 Pts	8.415 Pts	80.434 Pts	761.080 Pts
4	88.849 Pts	7.611 Pts	81.238 Pts	679.842 Pts
5	88.849 Pts	6.798 Pts	82.050 Pts	597.792 Pts
6	88.849 Pts	5.978 Pts	82.871 Pts	514.921 Pts
7	88.849 Pts	5.149 Pts	83.700 Pts	431.221 Pts
8	88.849 Pts	4.312 Pts	84.537 Pts	346.685 Pts
9	88.849 Pts	3.467 Pts	85.382 Pts	261.303 Pts
10	88.849 Pts	2.613 Pts	86.236 Pts	175.067 Pts
11	88.849 Pts	1.751 Pts	87.098 Pts	87.969 Pts
12	88.849 Pts	880 Pts	87.969 Pts	0 Pts

4. CONCLUSION.

El explicar las operaciones financieras dejando que los alumnos se entretuvieran planteando directamente diferentes supuestos, mas o menos reales, pensamos que fue una fuente de ahorro de tiempo y captación de interés muy positiva y perfectamente aplicable a la enseñanza universitaria. Los resultados expresados en el "análisis cuantitativo" desarrollado por el Instituto para la Formación de la Comunidad de Madrid podrían resumirse en:



Puntuando máximo 5, donde:

1 Representa Metodología.
2 Material Didáctico.
3 Equipamiento.
4 Centro de formación.
5 Profesorado.

El problema básico que nos limita a la hora de incluir estas prácticas en un curso normal de matemática financiera se centra en un único hecho: tenemos mas de seis mil alumnos y una única aula de informática con cuarenta ordenadores útiles.

EVOLUCIÓN DE CONTENIDOS EN LA ASIGNATURA MICROINFORMÁTICA PARA ECONOMISTAS

Pilar Arques, Otto Colomina, Patricia Compañ

Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
e-mail: {arques, otto, patricia}@dccia.ua.es

RESUMEN: En el presente artículo se presenta la situación actual y la posible evolución de la asignatura de Microinformática, que se imparte en las Licenciaturas de Economía y Administración y Dirección de Empresas en la Universidad de Alicante. Incluimos los resultados de una encuesta realizada en el presente curso académico que muestra el nivel de conocimientos previos del alumnado, y en base a ella nos planteamos la evolución de la asignatura en cursos sucesivos.

1.- INTRODUCCIÓN.

La implantación de las tecnologías de la información en el mundo laboral ha supuesto que los docentes debamos incluir una serie de contenidos informáticos necesarios para desarrollar cualquier actividad laboral. Hoy en día, es casi imposible encontrar una empresa que no utilice ordenadores para gestionar su contabilidad, controlar la nómina de sus empleados o realizar cualquier informe. Aunque no todos los empleados de la empresa van a tener que manejar los programas que realizan estas tareas, es conveniente que se tengan unos conocimientos mínimos que luego puedan ser ampliados en la empresa mediante cursillos específicos o simplemente de forma autodidacta por la misma persona.

En este artículo vamos a tratar sobre el contenido actual y la posible evolución de la asignatura *Microinformática*, de las Licenciaturas de Economía y Administración y Dirección de Empresas en la Universidad de Alicante.

2.- SITUACIÓN ACTUAL.

En el actual curso académico, se ha realizado una encuesta sobre una población de 494 alumnos, mediante la que podemos analizar al alumnado de Microinformática [Arques99].

Mediante estas encuestas los alumnos valoran sus conocimientos de las distintas partes de la asignatura: Windows, Procesador de Textos, Hoja de Cálculo e Internet, así como de la importancia de la asignatura en su carrera.

Después de analizar los resultados llegamos a la siguientes conclusiones:

- Los conocimientos iniciales del grupo actual sobre Sistema Operativo, Procesador de Textos e Internet son superiores a los conocimientos del grupo en el año anterior sobre las mismas materias.

- El conocimiento de la aplicación de hoja de cálculo sigue siendo escaso.

- En su mayoría, al igual que el año anterior, encuentran la asignatura de Microinformática muy importante para su carrera.

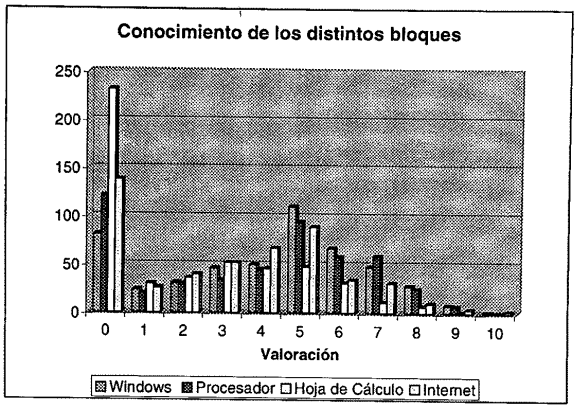
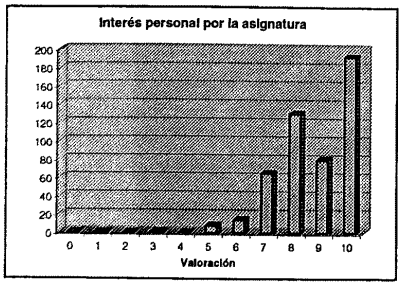
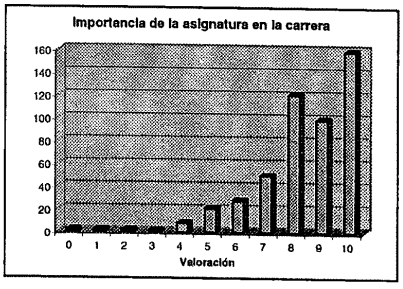


Ilustración 1. Valoración de los conocimientos previos.



Interés personal por la asignatura.



Importancia de la asignatura en su carrera.

Desde nuestro punto de vista, cuando se plantea impartir docencia de informática se debe tener en cuenta la carrera en la que se va a impartir (además, por supuesto, de los descriptores del Plan de Estudios).

En el caso de Economía, teniendo en cuenta las directrices generales y reflexionando sobre la utilización del ordenador que van a hacer en un futuro estos estudiantes, parece interesante insistir en hojas de cálculo. Además, pensamos que introducir unos contenidos básicos de Internet es muy aconsejable dado el impacto social que ha producido. Los resultados de las encuestas también apoyan estas conclusiones, los alumnos consideran que las hojas de cálculo en concreto, son fundamentales en su posterior actividad profesional y les gustaría que la asignatura tuviera más créditos para poder profundizar más en esa herramienta.

3.- EVOLUCIÓN DE CONTENIDOS

Debido al impacto de la informática en todos los aspectos económicos y sociales, debemos analizar la situación actual de la asignatura y la evolución de sus contenidos.

Actualmente, esta asignatura está enfocada hacia el aprendizaje básico de unas herramientas muy concretas, que son sistema operativo, procesador de texto, hoja de cálculo e Internet. No obstante, el enfoque ha cambiado con respecto al curso anterior, la asignatura se orienta más a profundizar en temas de hoja de cálculo, ya que un amplio porcentaje del alumnado sabe manejar el sistema operativo Windows, y un procesador de textos ([Arques99]).

También hay que tener en cuenta, la evolución de las herramientas; cuando empezamos a impartir esta asignatura, se impartía como sistema operativo, MS-DOS y Windows 3.1, como procesador de textos, Word Perfect 5.1, y como hoja de cálculo Lotus 1-2-3. Y por supuesto no impartíamos Internet. Actualmente, utilizamos el sistema Operativo Windows 98, y para las herramientas de procesador de textos y hoja de cálculo, utilizamos el Microsoft Office 2000. Con esto queremos decir, que vamos evolucionando y adaptando el temario, a medida que cambiamos a una versión más actualizada.

Por todo ello, no podemos dejar de lado, las ideas para un enfoque futuro de la asignatura. Nuestras propuestas para años posteriores son las siguientes:

a) Simulación de procesos económicos y decisionales.

El funcionamiento de una hoja de cálculo no se limita a la simple utilización de operadores matemáticos. El alumno debe conocer también otras herramientas avanzadas para resolver problemas de decisión en el ámbito de la empresa ([Carbonell99]).

Con frecuencia, estamos interesados en conocer cómo afecta el cambio de uno o varios datos de entrada a los resultados, es decir, queremos experimentar con los datos a fin de responder a la pregunta ¿qué pasa si...?. Mediante los **escenarios** de Excel podemos crear un modelo que nos permita solucionar este problema. Además facilitan la creación de informes con los distintos datos de entrada y sus correspondientes valores de salida.

Otro tipo de problemas a resolver son los problemas de optimización con restricciones. Se caracterizan por tener un único objetivo a optimizar (maximizar, minimizar o igualar a) que depende de uno o más valores de entrada a determinar y está sujeto a una o varias restricciones.

La herramienta **solver**, de Excel, utiliza técnicas de programación lineal y de programación no lineal para resolver estos problemas.

Mediante la herramienta **buscar objetivo** de Excel se pueden resolver problemas "hacia atrás", es decir, encontrar el valor que debe tener una celda, *valor de entrada*, para lograr determinado resultado en otra celda, *valor objetivo*.

b) Búsqueda selectiva de información.

Actualmente se estima que hay entre 200 y 320 millones de documentos disponibles en la World Wide Web, y la cifra se duplica cada año. Esta abundancia de información ha hecho que mucha gente compare la red con una gran biblioteca en la que se puede localizar cualquier tipo de datos. No obstante, la naturaleza descentralizada de Internet, que ha sido precisamente la que ha hecho posible este crecimiento, dificulta la búsqueda de información en la misma. En Internet, a diferencia de las bibliotecas no hay un "catálogo central", ni tampoco hay un sistema de clasificación aceptado por todos. En definitiva, dicen que **en Internet es fácil buscar, pero difícil encontrar** ([Cazorla99]).

Para solucionar este problema, el alumno debe conocer cuáles son las principales herramientas de búsqueda de información en Internet. Además, debe dominar los lenguajes documentales (operadores lógicos) que permiten realizar búsquedas más especializadas y selectivas.

Por último, es interesante que el alumno conozca herramientas avanzadas como los sistemas de metabúsqueda, que permiten lanzar la búsqueda de manera simultánea en varios buscadores para aprovechar los beneficios de cada uno de ellos.

c) Comercio electrónico y *e-business*

Aunque las grandes empresas llevan bastante tiempo realizando transacciones económicas a través de redes telemáticas (mediante sistemas de tipo EDI), en los últimos años la popularización de Internet ha permitido que empresas de cualquier tamaño realicen, a través de esta red, transacciones con los consumidores y con otras empresas. Esto ha dado lugar al auge del *comercio electrónico* ([Kosiur97]). Cuando el comercio electrónico se integra con los sistemas informáticos de uso interno de la empresa, tenemos lo que se ha dado en llamar *e-business*.

Evidentemente, la formación de los futuros licenciados en Economía y Administración de Empresas no puede permanecer al margen de estas cuestiones, como lo demuestra el número creciente de cursos en comercio electrónico y *e-business* que se vienen impartiendo en estas carreras en distintas universidades[34].

Desde el punto de vista de nuestra asignatura, creemos que una formación adecuada en estos aspectos requiere conocer dos tipos de tecnologías:

- **Tecnologías para gestión de la información:** Asímismo debe saber cómo pueden integrarse estas herramientas en la propia gestión de la empresa, dando lugar a tecnologías de red como las intranets o las extranets. En la asignatura, estos contenidos quedan parcialmente cubiertos en el bloque temático sobre Internet.

[34] Como ejemplo, en la dirección http://dossantos.cbpa.louisville.edu/ISNET/Ecomm/ se puede acceder a una lista de más de 35 cursos que ofrecen distintas universidades (aunque principalmente extranjeras).

- **Tecnologías específicas para e-business:** dentro de este apartado, cabría distinguir dos tipos de herramientas: las que se encargan de los aspectos relativos a gestión de la información, como las bases de datos accesibles vía web (por ejemplo, un catálogo electrónico), y por otro lado las que dan soporte a los aspectos económicos de las transacciones en red. Dentro de estas últimas estarían los sistemas de pago en Internet (puntos de venta "virtuales", SET, dinero electrónico,...) e íntimamente ligados a ellos, los sistemas de seguridad en Internet (criptografía, firmas digitales, firewalls...).

En cuanto al nivel de profundización en la materia, si bien el alumno no puede (debido a los lógicos problemas de tiempo) ni necesita conocer todos los detalles de estas tecnologías (ya que no es un informático), sí debe tener una visión general de todas ellas y ser capaz de comprender y evaluar adecuadamente los beneficios que puedan representar para su empresa.

En este sentido, y dentro del temario de la asignatura, se podría plantear la inclusión de un tema relativo a *tecnologías para gestión de la información en e-business*, en el que se tratarían herramientas hardware y software como las intranets y los catálogos electrónicos, y un tema sobre *medios de pago y seguridad en Internet*. Además de los conceptos teóricos necesarios, desde el punto de vista práctico sería interesante que el alumno pudiera aprender a colocar en Internet o en una intranet la información que él mismo ha generado mediante herramientas ofimáticas y, además, que adquiriera familiaridad con algunas aplicaciones básicas de seguridad, como el correo electrónico encriptado o los documentos firmados digitalmente (a través de herramientas estándar y no demasiado complicadas de utilizar, como por ejemplo PGP para Windows, o los mecanismos de seguridad incorporados en los propios navegadores).

Como se puede apreciar en estas propuestas, estamos dejando de lado las cuestiones básicas de manejo de sistema operativo y de procesador de textos. Actualmente, se están implantando asignaturas en colegios e institutos que cubren esos temarios, con lo que los estudiantes llegarían a la universidad con una buena base de aplicaciones ofimáticas.

4.- CONCLUSIONES

La informática aplicada a la gestión empresarial es una disciplina en rápida evolución, y por ello el temario de la asignatura debe estar en consonancia con esta evolución. La asignatura de microinformática comenzó siendo una introducción al manejo básico de un ordenador, pero a medida que los alumnos van adquiriendo conocimientos previos de herramientas ofimáticas en la enseñanza primaria y secundaria, se hace necesaria una adaptación de contenidos. Por ello, hemos planteado la introducción de temas como la búsqueda selectiva de información en Internet, comercio electrónico y simulación de procesos económicos y decisionales en Excel. Estamos seguros de que en años posteriores los contenidos deberán seguir evolucionando y adaptándose a las necesidades de nuestros futuros alumnos.

5.- BIBLIOGRAFÍA

[Arques99] P.Arques, M. Cazorla, P. Compañ, R. Molina. *Docencia de la informática en las titulaciones de Económicas*. Actas JENUI 99.

[Arques00] P. Arques. Proyecto docente para Modelos Abstractos de Cálculo y Microinformática. 2000.

[Cazorla99] M.A. Cazorla, O. Colomina, P. Compañ. *Internet para universitarios*. Servicio de Publicaciones de la Univ. de Alicante, 1999.

[Carbonell99] L. Carbonell, P. Compañ, M. Marhuenda, S. Ramón. *Hoja de cálculo para economistas*. Servicio de Publicaciones de la Univ. de Alicante, 1999.

[Kosiur97] D. Kosiur. *Understanding Electronic Commerce*. Microsoft Press, 1997.

"INFORMATICA APLICADA A LA GESTION", ASIGNATURA DE LOS PLANES DE ESTUDIO DE INGENIERIA TECNICA EN INFORMATICA DE GESTION Y DE CIENCIAS EMPRESARIALES.

Ramon Segret Sala[1], Carlos Fernando Cabañero Pisa[2]

[1]*Profesor de los Estudios de Informática y Multimedia de la UOC*
e-mail: rsegret@campus.uoc.es

[2]*Profesor de los Estudios de Economía y Empresa de la UOC*
e-mail: ccabanero@campus.uoc.es

Resumen: En el presente artículo se presenta la experiencia de la preparación y docencia de la asignatura "Informática aplicada a la gestión", cuya característica más innovadora es su impartición a un grupo mixto de alumnos pertenecientes a las carreras de Ingeniería técnica en informática de gestión y de Ciencias empresariales, de la Universitat Oberta de Catalunya (UOC).

1.- INTRODUCCIÓN

La asignatura "Informática aplicada a la gestión" es común a los planes de estudio de la Ingeniería técnica en informática de gestión, de los Estudios de Informática y Multimedia, y de la diplomatura de Ciencias empresariales, de los Estudios de Economía y Empresa, ambos de la Universitat Oberta de Catalunya (UOC).

En ambas carreras es una asignatura obligatoria, de 6 créditos, situada en el último semestre. Se trata pues de una asignatura finalista que requiere los conocimientos que se adquieren al cursar las asignaturas de los semestres anteriores.

2.- OBJETIVOS

Al diseñar los contenidos de esta asignatura desde los dos estudios involucrados nos planteamos como objetivo principal proporcionar a los estudiantes un medio que permitiera simular el entorno real que se van a encontrar en el mundo laboral.

Actualmente, en cualquier organización, se toman decisiones partiendo tanto de los conocimientos empresariales como de los tecnológicos, mayormente informáticos y telemáticos. Por esta razón, estas decisiones se discuten en grupos de trabajo interdisciplinares, que cada vez más son virtuales, ya que sus miembros pueden estar trabajando en distintas localidades.

a) Enfoque de la asignatura

Todo ello nos llevó a diseñar una asignatura con el siguiente enfoque:

- orientada a la resolución de casos,
- agrupando estudiantes de empresariales y de informática en una misma aula virtual,
- que los estudiantes resolvieran los casos en grupos interdisciplinares,
- grupos que evidentemente serían virtuales en virtud de la metodología propia de nuestra universidad.

Resumiendo, podemos decir que *Informática aplicada a la gestión* es una asignatura:

- basada en casos
- y en su resolución en grupos virtuales interdisciplinares,
- donde los estudiantes deben, por una parte, poner en práctica los conocimientos adquiridos durante toda la carrera y, por otra, buscar aquellos que les falten para la resolución del caso a través de lecturas complementarias, búsquedas bibliográficas y búsquedas en Internet,
- y todo ello con el objetivo principal de simular el trabajo real en una empresa o institución.

b) Beneficios para los estudiantes

Esta asignatura proporciona a los estudiantes la oportunidad de profundizar en los conocimientos aprendidos durante la carrera y de adquirir una experiencia práctica.

Les permite, también, darse cuenta de que las tecnologías de la información y comunicación constituyen oportunidades estratégicas para las empresas e instituciones.

Pero lo más importante es que aprenden a trabajar en grupo interdisciplinar, de forma virtual, mientras descubren las estrechas relaciones que existen hoy en día entre la tecnología informática y la estrategia empresarial, adquiriendo con todo ello una experiencia que les va a ser de gran valor para su vida profesional.

3.- ESTRUCTURA DE LA ASIGNATURA

La asignatura es semestral. Durante el semestre se trabajan uno o dos casos, según la longitud y complejidad de los mismos.

Junto con el caso proporcionamos una guía de la resolución del mismo, que ayuda al estudiante a planificar su trabajo y no deja que el proceso de resolución quede tan abierto. Este tipo de ayuda metodológica no es en absoluto un elemento extraño en un entorno virtual de aprendizaje, como el de la UOC. Recordemos que alumnos y profesores no coinciden ni en el espacio ni en el tiempo.

Para esta asignatura la guía es, fundamentalmente, un conjunto de pistas para desarrollar el caso, agrupadas en unos hitos que los estudiantes deben ir superando secuencialmente. Cada hito se compone de los objetivos a conseguir, las instrucciones de cómo trabajarlo, unas actividades que sirven de ayuda para avanzar en la resolución del caso y, finalmente, unos ejercicios que debe resolver. En el último hito, el ejercicio es, evidentemente, la resolución final del caso propuesto.

A todo ello hay que añadir las lecturas complementarias y les referencias a artículos, libros o webs de Internet. A través de la lectura y estudio de estos documentos los estudiantes adquieren conocimientos específicos y de actualidad que les van a facilitar la correcta resolución del caso mediante decisiones basadas en informaciones contrastadas.

4.- COMPONENTES DE LA ASIGNATURA

Los componentes a disposición de los estudiantes para seguir esta asignatura son: el Plan docente, el material de la asignatura, los documentos de ayuda y el soporte de trabajo en grupo. Los estudiantes acceden a todos estos componentes a través de su conexión al campus virtual de la Universidad.

El Plan docente es una herramienta pedagógica que tienen todas las asignaturas de la UOC. Es el primer documento que lee el estudiante antes de empezar cualquier asignatura. Entre otras cosas es una agenda con la temporalización del estudio de la asignatura y las fechas de las pruebas de evaluación.

Los materiales de la asignatura son variados. Siguiendo lo expuesto anteriormente, en *Informática aplicada a la gestión* los estudiantes disponen del caso, de la guía con los hitos y de un conjunto de lecturas complementarias, referencias bibliográficas y direcciones Internet.

Confeccionamos unos documentos de ayuda para introducir al estudiante en el espíritu del trabajo en grupo y sus requerimientos específicos. Así mismo les hemos proporcionado una guía para la formación de grupos y otra para el uso del producto informático empleado para el trabajo en grupo.

Finalmente, respecto al trabajo en grupo, mencionaremos que existe un grupo amplio, coincidente con todos los alumnos del aula, que sirve para que todos ellos debatan cuestiones de interés general. A medida que el curso progresa se forman los grupos de trabajo propiamente dichos. Idealmente estos grupos son de 4 personas, 2 informáticos y 2 de empresariales. Para cada grupo de trabajo formado, se crea el soporte correspondiente en el producto informático que utilizamos.

Actualmente usamos el servidor Domino, de Lotus, al cual acceden los estudiantes con un navegador desde su PC.

5.- EVALUACION

Nos planteamos evaluar lo siguiente:

- las habilidades en combinar conocimientos y técnicas para resolver los casos planteados,
- la capacidad para adquirir los conocimientos que les pudieran faltar,
- la actitud de trabajo en grupo, y
- la habilidad para trabajar en un grupo virtual.

Todo ello conduce a un modelo de evaluación que da mucho más peso al trabajo diario que a un examen final. Por esto esta asignatura no tiene un examen final en el mismo sentido que la mayoría de las otras asignaturas que se cursan en la UOC, sino que lo que se evalúa es:

- el trabajo diario de cada estudiante,
- la actitud del estudiante, sobre todo cuando trabaja en grupo, y
- los ejercicios de cada hito.

6.- DESARROLLO DEL CURSO

a) Contenido

En el curso desarrollado durante el semestre Marzo a Julio de 2000 se ha propuesto un solo caso a los estudiantes, para su trabajo y resolución. El caso presenta una librería que se plantea la conveniencia o no de estar presente en Internet y en qué medida le interesa estar. Se trata de un caso en que los estudiantes deben analizar cuestiones de estrategia, de márqueting y de análisis de costos y beneficios, proyectados a cinco años. Combinando estos factores tienen que llegar a una conclusión razonada.

La parte informática del caso está en la identificación de los distintos elementos del sistema de información que va a permitir la solución Internet y en su correcta valoración en relación al análisis de costos y beneficios mencionado antes.

b) Consultoría

Siguiendo el modelo de la Universidad, un profesor consultor ha seguido día a día el avance de la asignatura por parte de los estudiantes. Ha respondido a preguntas puntuales, ha encauzado el debate sobre el caso y el desarrollo del mismo y ha proporcionado informaciones de actualidad relevantes.

Finalmente ha propuesto los diferentes ejercicios y ha evaluado a los estudiantes.

c) Temporalización

Se previó una semana a principio de curso para que los estudiantes aprendieran el uso de la herramienta de soporte del trabajo en grupo. Se entró luego en la primera parte del caso, que los estudiantes trabajaron de forma individual, aunque tuvieron a su disposición el espacio común proporcionado por la herramienta, para poder consultar y debatir.

Seguidamente se les dio otra semana para que formaran los grupos de trabajo tal como se detalla en el punto siguiente. A partir de este momento los estudiantes trabajaron en grupos independientes la segunda y última parte del caso.

d) Estudiantes

En el semestre de referencia hemos tenido 23 estudiantes matriculados, aproximadamente la mitad provenientes de los estudios de Ingeniería técnica informática y la otra mitad de los de la diplomatura de Ciencias empresariales.

A la hora de crear los distintos grupos de trabajo, se constituyeron cinco grupos, tres de cinco estudiantes y dos de cuatro. Se repartieron los estudiantes de informática y los de empresariales entre

todos los grupos, de forma que los dos colectivos estuvieran representados lo más uniformemente posible.

7.- RESULTADOS OBSERVADOS

De los cinco grupos que se formaron, cuatro han seguido el curso y uno no. Los estudiantes de los grupos que han seguido han obtenido calificaciones, en general, muy buenas.

La opinión general de los estudiantes es positiva. Consideran que tanto los contenidos del curso como, sobre todo, la oportunidad de compartir un espacio de trabajo común con estudiantes de otras disciplinas han enriquecido satisfactoriamente sus conocimientos y habilidades. El desarrollo de curso se ha correspondido con el objetivo principal de la asignatura, que era proporcionar un escenario de trabajo que simule el entorno real que se encontrarán en el mundo laboral.

Donde han encontrado ciertas dificultades es en la metodología de trabajo, es decir, en la forma concreta de trabajar en grupo. Mencionan los estudiantes que les ha faltado formación sobre cuestiones tales como la forma de comunicarse entre ellos y de coordinarse el trabajo. Encuentran a faltar la posibilidad de una sincronización eficaz del grupo para, en determinados momentos, tomar decisiones de forma ágil y rápida. Como veremos ésta es un tarea pendiente para el próximo curso.

Es interesante constatar la forma cómo han trabajado los estudiantes, que ha coincidido bastante en los diferentes grupos. Un primer rasgo es que han trabajado siguiendo unos ciclos que se corresponde con los distintos hitos del caso. En cada hito se observa que uno de los miembros del grupo se otorga el rol de coordinador y en unos primeros mensajes al resto de compañeros marca las prioridades, las tareas y los plazos.

Posteriormente suelen aparecer dos borradores de documentos, uno elaborado por uno de los estudiante de empresariales del grupo e intentado dar una primera versión de respuesta a las preguntas planteadas en el hito, pero centrándose en los temas propios de su carrera, y el otro, que es obviamente lo mismo, pero confeccionado por uno de los informáticos y trabajando los temas informáticos.

A partir de este punto los miembros del grupo que aun no han intervenido se ven forzados a hacerlo y suelen hacerlo de formas variadas. O bien debaten los borradores publicados, exponiendo sus puntos de vista, o bien intentan escribir algún borrador alternativo si sus ideas son muy dispares de las expresadas hasta aquel momento, o bien, a partir de la primera versión del documento, la recogen, amplían y crean una segunda versión.

Finalmente alguien del grupo queda asignado para la confección del documento final, que tras una ronda de debate y opiniones, se le da la versión definitiva y se entrega al consultor.

El consultor tiene en esta asignatura un papel aun más importante, si cabe, que el que tiene en las otras asignaturas de las carreras. Ya se ha mencionado con anterioridad sus distintas actuaciones a lo largo de este curso.

8.- FUTURO

La buena acogida que ha tenido esta asignatura nos ayuda en la decisión de seguir ofreciéndola en próximo semestre con el mismo enfoque, intentando, esto sí, introducir algunas mejoras. Estas se van a concentrar, fundamentalmente, en las áreas de la metodología del trabajo en grupo y de las herramientas utilizadas.

En la primera vamos a reforzar la formación de los estudiantes para que sepan trabajar en grupo de forma más eficiente. Pensamos desarrollar los puntos siguientes:

- Dificultades normales en un entorno de trabajo en grupo y consejos prácticos para superarlas.
- Explicación clara de cual debe ser la actitud y disponibilidad temporal de los miembros del grupo.
- Lista de roles y/o tareas que hay que llevar a cabo para resolver el caso de forma cooperativa en grupo.
- Criterios de distribución de estos roles y tareas entre los miembros del grupo.

En cuanto a las herramientas nuestra idea es complementar la capacidad asíncrona actualmente utilizada para desarrollar el trabajo en forma cooperativa, con mecanismos que les permitan comunicaciones síncronas.

9.- BIBLIOGRAFIA

Marta C. Rosatelli y John A. Self, Supporting Distance Learning from Case Studies. En: S. P. Lajoie and M. Vivet (editores), Artificial Intelligence in Education. IOS Press, 1999.

Montse Guitert *et al.*, El treball en equip a les assignatures de Multimedia i Comunicació i Estructura de la Informació a l'entorn del Campus Virtual de la UOC (Projecte Bellver). Documento interno de la Universitat Oberta de Catalunya, 1999

CONTENIDOS Y OBJETIVOS DE LA ASIGNATURA DE INFORMÁTICA APLICADA A LA GESTIÓN DE LA EMPRESA.

José Manuel Rodríguez García ,Miguel Angel Pérez Toledano.

Departamento de Informática

Área de Lenguajes y Sistemas Informáticos

Escuela Universitaria Politécnica

Universidad de Extremadura

E-mail: { jmrodri, toledano }@unex.es

RESUMEN: Con la aparición de los nuevos planes de estudios donde se dejaban de ofertar asignaturas de menos de 4,5 créditos, las asignaturas troncales Informática de Gestión I e Informática de Gestión II (ambas de 3 créditos), recogidas en los planes de estudio de la Diplomatura en Ciencias Empresariales impartida en Cáceres y perteneciente a la Universidad de Extremadura, han dejado de impartirse, y han dado pie a la aparición de una nueva asignatura Informática Aplicada a la Gestión Empresarial y cuyo contenido explicamos en el trabajo.

1.- INTRODUCCIÓN.

La asignatura de Informática Aplicada a la Gestión de la Empresa, es una asignatura troncal de 6 créditos (3 de teoría y otros tantos de prácticas) que se empezará impartir el curso 2000/2001 dentro los planes de estudio de la Diplomatura de Ciencias Empresariales, y que viene a cubrir el hueco que en los anteriores planes de estudios ocupaban las asignaturas troncales de Informática de Gestión I y II. No obstante, la nueva propuesta no es una mezcla ordenada de los contenidos que se cubrían con anterioridad, sino que es bastante más ambiciosa e intenta dar una visión lo más amplia posible de la realidad informática en la que nos movemos, haciendo especial hincapié en facilitar herramientas que ayuden a resolver problemas planteados en los diferentes sistemas de información en los que se moverán los actuales alumnos y futuros profesionales.

Esta línea argumental está deriva de la realidad económica de la región, cuyo tejido productivo está constituido esencialmente por pequeñas y medianas -las menos- empresas, lo que nos lleva a formar profesionales con un amplio registro de capacidades informáticas, en oposición a entornos con empresas de mayor tamaño medio en los que es posible y deseable una mayor especialización.

Antes de entrar en lo que abarcará Informática Aplicada, vamos a intentar describir el entorno de trabajo de la asignatura. El número de alumnos oscila cada año sobre 250 nuevos matriculados. No existe ninguna otra asignatura de informática, optativa o de libre elección, que complemente los conocimientos adquiridos, aunque se está trabajando para encontrar recursos que nos permitan ofertar

una asignatura optativa de informática básica que se impartiría antes, y que serviría para normalizar los dispares conocimientos que presentan los alumnos que acceden a esta asignatura. Sería recomendable ofertar otra asignatura optativa que se impartiría, con posterioridad a la de Informática Aplicada a la Gestion de la Empresa, y que profundizaría en el desarrollo de soluciones más complejas para los más interesados.

Por ultimo, es interesante hacer constar que la realidad económica de la región,

2.- CONTENIDOS.

El temario de la asignatura está dividido en cuatro bloques con diferentes objetivos:

a) Los temas 1, 2 y 3 están centrados en asimilar el concepto de sistema, e ir profundizando en los sistemas de información como base para conocer el entorno de trabajo dentro de las organizaciones, y al mismo tiempo aprender a clasificar y cuantificar la información.

b) Los temas 4 y 5 pretenden enseñar al alumno la forma de afrontar el desarrollo de soluciones informáticas, dándoles a conocer las diferentes metodología de trabajo así como las ventajas e inconvenientes de cada una de ellas y las herramientas que utilizan.

c) Los temas de bases de datos pretenden ofrecer al alumno una visión sencilla de cómo utilizar las herramientas estudiadas en prácticas para desarrollar soluciones aprovechando los conocimientos de los temas anteriores y completados con la visión de las bases de datos relacionales como forma para modelar los datos.

d) El último tema está pensado para conocer Internet como herramienta de trabajo y aprender los servicios que ofrece, la forma de utilizarlos y el funcionamiento en general de la World Wide Web.

El programa teórico de la asignatura es el siguiente:

Parte Teórica

Tema 1° Introducción a los sistemas

Objetivos
1.1 Introducción
1.2 Concepto de sistema
1.3 Clasificación de los sistemas
1.3 Organizaciones y empresas como sistemas
1.5 Sistemas de información
1.6 Perspectiva general de los sistemas

Tema 2° Introducción al análisis de sistemas

Objetivos
2.1 Introducción

2.2 ¿Por qué una metodología?
2.3 ¿Quiénes intervienen en el desarrollo de sistemas?
2.4 Sistemas transacionales y sistemas de tiempo real
2.5 Estilos de interfaz hombre-máquina
2.6 El centro de procesos de datos
2.7 El software. Características

Tema 3° Sistemas y subsistemas en la empresa
Objetivos
3.1 Introducción
3.2 División en módulos
3.3 Sistema de almacenamiento
3.4 Sistema de aprovisionamiento
3.5 Sistema comercial
3.6 Sistema financiero
3.7 Sistema contable
3.7 Sistema de personal
3.9 Sistema de producción

Tema 4° Metodologías de análisis y desarrollo
Objetivos
4.1 Introducción
4.2 Ciclo de vida
4.3 Metodología Warnier-Orr
4.4 Ciclo de vida clásico
4.5 Diseño en espiral
4.6 Prototipos
4.7 Metodología SSADM
4.8 Metodología MERISE
4.9 Métrica 2.1

Tema 5° Otras herramientas del ciclo de vida
Objetivos
5.1 Introducción
5.2 Las encuestas
5.3 Grafos PERT
5.4 Diagramas GANTT

Tema 6° Modelado de datos
Objetivos
6.1 Introducción
6.2 El modelo Entidad Relación
6.3 Reglas para la construcción de diagramas Entidad Relación

Tema 7° Introducción a las bases de datos
Objetivos
7.1 Bases de datos, ¿qué son?
7.2 Perspectiva histórica
7.3 Componentes de una base de datos
7.4 Características de las bases de datos
7.5 Tipos de bases de datos

7.6 Arquitectura interna de un sistema de base de datos

Tema 8°. El modelo relacional
Objetivos
8.1 Introducción
8.2 Estructura del modelo relacional
8.3 Claves
8.4 Reglas de integridad
8.5 Lenguajes de manipulación

Tema 9°. Introducción a Internet.
Objetivos
9.1 Introducción
9.2 Conceptos básicos: arquitectura, protocolos, direcciones y dominios.
9.3 Servicios en Internet.
9.4 Funcionamiento de la Word Wide Web.

En cuanto a la parte de prácticas, esta se impartirá en un aula de 25 ordenadores con sistemas operativos Windows 98, conectados entre si mediante una red Ethernet, y con un servidor con sistema operativo Windows NT Server. Los grupos de trabajo siguiendo las directrices del departamento de Informática serán de 30 alumnos, y los contenidos a trabajar serán desarrollados con las siguientes herramientas:

Parte Práctica

Herramientas de cálculo: Excel
Herramienta de gestión de Base de datos: Access
Gestión de proyectos: MsProject
Estudio de un sistema automatizado de información
Navegador: Netscape

3.- OBJETIVOS.

La evolución técnica que se ha producido a lo largo de los últimos decenios ha hecho de la informática un pilar básico en la existencia humana. El ámbito empresarial no podía sustraerse a este empuje y, como consecuencia de ello, tanto empresas como organizaciones en general están profundamente determinadas por esta evolución técnica. Así, por ejemplo, es difícil encontrar algún puesto de trabajo en las áreas administrativas y comerciales de las empresas que no tengan, de una u otra manera, soporte informático.

La informática, en el ámbito de organizaciones y empresas, se inserta en un marco más amplio: el de los sistemas de información. Los sistemas de información, verdaderas columnas vertebrales de las organizaciones, han sido, a su vez, influenciados por el empuje de la técnica informática, convirtiéndose, en muchos casos, en entornos complejos en los que es difícil manejarse y para los que se requieren conocimientos especializados.

No obstante, no solamente la informática es importante desde el punto de vista de las aplicaciones verticales, de aplicaciones de gestión de sistemas de información. También, en la circunstancia actual, los futuros profesionales de la gestión de empresas deben conocer las

posibilidades que la informática les brinda como herramienta. Aplicaciones como por ejemplo la simulación mediante ordenador, el proceso de textos o las presentaciones mediante programas especializados, pueden ser una buena muestra de la importancia que la informática tiene en las empresas.

A la vista de las consideraciones anteriores, los objetivos que se persiguen alcanzar en la asignatura son los que a continuación se exponen:

a) Dotar al alumno de una visión general de los sistemas de información y de su importancia en organizaciones y empresas.

b) Mostrar al alumno algunas de las diferentes metodologías que existen para el análisis de sistemas de información.

c) Ayudar al alumno en la comprensión de la metodología de análisis de sistemas estructurada.

d) Capacitar al alumno en el uso de herramientas de gestión y planificación de proyectos para sistemas de información.

e) Proporcionar al alumno un lenguaje que permita el intercambio de ideas con las áreas de conocimiento tecnológico informático y le ayude tanto en el desarrollo de soluciones como en el modelado de datos.

f) Facilitar al alumno el conocimiento de diferentes herramientas informáticas que permitan su desarrollo profesional.

g) Introducir al alumno en la utilización de Internet como herramienta de trabajo cotidiano.

4. BIBLIOGRAFÍA DE TRABAJO

Apuntes de la asignatura.

Sistemas de Bases de Datos. C. J. Date. Ed. Addison Wesley Hispanoamericana 1993

Sistemas y Tecnologías de la Información para la Gestión. Ignacio Gil Pechuán. Ed. MacGraw-Hill 1997

Introducción al Análisis de Sistemas. I.I Hawryuzkiewycz. Ed. Anaya 1990

Análisis y diseño de sistemas. Keneth E. Kendall, Julie E. Kendall Ed. Prentice-Hall Hispanoamericana S.A. 1991

Fundamentos de Bases de Datos. Henry F. Korth, Abraham Silberschatz. Ed. Mac Graw-Hill 1998.

La Estructuración de las Organizaciones. Henry Mintzberg Ed. Ariel 1984.

Sistemas de Información Administrativa. Robert G. Murdick, John Munson. . Ed. Prentice Hall 1988

Simulación. Aplicaciones prácticas en la empresa. Leandro Pardo, Teófilo Valdés. Ed. Diaz de Santos S.A. 1987

Análisis y Diseño Detallado de Aplicaciones Informáticas de Gestión. Mario G. Piattini, José A.Calvo Manzano,Joaquín Cevera, Luis Fernández. Ed. RA-MA 1996

Ingeniería del software. Un enfoque práctico. Roger S. Pressman. Ed. MacGraw-Hill 1994

Análisis Estructurado Moderno. Edward Yourdon. Ed. Prentice Hall, 1993

5.- CONCLUSIONES.

En la exposición de los contenidos y objetivos de la asignatura de Informática Aplicada a la Gestión de la Empresa hemos intentado eliminar todas referencias a temas de informática básica con la esperanza de complementar sus contenidos con alguna asignatura optativa que le preceda y que nos permitirá avanzar en los objetivos planteados en el descriptor de la asignatura, como son los sistemas de información en las organizaciones y el aprendizaje en el desarrollo de pequeñas soluciones informáticas que sirvan para automatizar los sistemas de información en los que se moverán los futuros diplomados en Empresariales.

BIBLIOGRAFÍA.

Bibliografía de trabajo, citada anteriormente, y que no vamos a repetir.

- J.M. Rodríguez. Los estudios de informática en los curricula de los economistas. Una propuesta alternativa. IV Jornadas sobre la Enseñanza Universitaria de la Informática. Libro de actas del congreso, pp 512-515. Andorra 1998.
- J.M.Torralba et all. La formación en tecnología de la información en las titulaciones de LADE en la comunidad Valenciana. V Jornadas sobre la Enseñanza Universitaria de la Informática. Libro de Actas, del congreso, pp 417-422. Zaragoza 1999.
- Virgós Bell, F., Segura Casanovas, J. Propuesta metodológica para la definición de contenidos de formación en TI/SI para titulaciones del área de em,presa. V Jornadas sobre la Enseñanza Universitaria de la Informática. Libro de Actas, del congreso, pp 423-428. Zaragoza 1999.

ENSEÑANZA DE LA LEGISLACIÓN PARA EL PROFESIÓNAL INFORMÁTICO.

Rosalía Peña

[1] *Departamento C.C. Sanitarias y Medico sociales.Universidad de Alcalá de Henares*
e-mail: ccrpr@uah.es

RESUMEN: Se discute la necesidad y objetivos de acercar al futuro informático a las leyes directamente relacionadas con su ejercicio profesional. Se plantea el contexto y dificultades de esta docencia. En consecuencia se propone una metodología. Se desarrolla un tema concreto dentro de la asignatura de Protección de la Información en las Ingenierías Técnicas de Informática y Telecomunicaciones. Se estudia el efecto de esta docencia sobre el alumnado evaluando una encuesta.

1.- INTRODUCCIÓN.

La generalización del uso del ordenador e Internet tiene fuertes implicaciones sociales y la legislación tanto nacional como supranacional esta en pleno desarrollo en la actualidad por lo que no existe una conciencia social de su importancia ni, incluso, de su existencia, es necesario exponer al futuro informático a unos principios de Ética y Legislación profesional.

Esta enseñanza podría abordarse desde dos aproximaciones diferentes: dedicar asignaturas específicamente a estas materias o insertar sus distintos aspectos en la docencia de las materias tradicionales con las que están directamente relacionados. Somos partidarios de la visión integrada porque entendemos que, ligado a su contexto tecnológico, el alumno asume mejor el dilema ético y sus obligaciones legales.

2.- DEFINICIÓN DE OBJETIVOS.

El objetivo general de la inclusión de los aspectos éticos y legales dentro de la asignatura de Protección de la Información es provocar un cambio de actitud sobre su responsabilidad profesional.

El primer paso es proporcionar la capacidad de saber reconocer la existencia de un dilema ético y las repercusiones de la decisión tomada para abordarlo. [Rachels93, Shepard97] y es nuestro objetivo fundamental en cuanto a los aspectos éticos.

Adicionalmente se pretende que el futuro informático sepa la existencia de unas leyes que afectan su trabajo y a grosso modo, conozca su contenido. No se desea convertirlo en un especialista en legislación, sino, capacitarle para detectar la necesidad de un mayor estudio concreto, o asesoramiento, en una situación determinada. En este sentido no es necesario que memorice la redacción concreta de

la ley, ni por supuesto, el número del artículo que desarrolla cada aspecto de la misma. Pero sí que tenga conocimiento de su existencia, acceso fácil a la fuente original y a su contenido global, familiarizarle con el lenguaje del entorno para que, en caso de necesidad, acceda a sus detalles concretos y excepciones.

3.- METODOLOGÍA Y PLANIFICACIÓN.

En nuestro plan de estudio la asignatura de Protección de la Información tiene 45 horas lectivas, de las cuales hemos dedicado unas 28 horas a conocimientos técnicos, del orden de 9 a legislación y unas 7 a discusiones abiertas sobre el papel de la ética en la protección de la información profesional. Aunque Ética y Legislación son materias distintas, por supuesto, la presentación de una ley puede dar pié a una discusión ética, y en una primera toma de contacto están muy relacionadas, por tanto las horas de unas y otras no están totalmente delimitadas.

En nuestra asignatura incluimos como "información a proteger", todo "el saber hacer", el know-how, de la empresa, además de sus datos y software desarrollado por ella, esto implica cubrir un amplio abanico de leyes.

Los alumnos llevan tres años dedicados casi en exclusiva a materias técnicas, cuya metodología y contenido es muy distinto del de estas dos vertientes de nuestra asignatura. El profesor también está mas acostumbrado a explicar materias donde el objetivo fundamental es la adquisición de unos conocimientos concretos mas que fomentar un cambio de actitud en el alumno.

Para el estudio de la legislación se facilita al alumno los textos legales en un formato uniforme, más grato de leer que el de la fuente original, así como la referencia completa a la fuente original. En la mayoría de los casos se ha entresacado del texto de la ley (Ley del IVA, Ley de la Propiedad Intelectual, Código penal) sólo los artículos directamente relacionados con la profesión informática y aquellos imprescindibles para comprenderla. En las legislaciones que abordan íntegramente temas de nuestro interés (como la LOPDP, o la norma de dominios) se han eliminado los artículos transitorios. Se menciona la existencia de artículos relativos a problemáticas muy especializadas pero no se entra en su contenido.

Las clases se imparten sobre material de presentación que facilite la síntesis y estructuración de los aspectos mas relevantes, suscitando discusiones y aportaciones de los alumnos. El alumno dispone de copia de este material, a fin de facilitar la tarea de toma de apuntes para que pueda tomar parte activa en la sesión

En el apartado 5 de este documento se presenta el material de apoyo utilizado para el desarrollo del Reglamento de medidas de seguridad.

4.-TEMARIO.

Se ha pretendido seleccionar el conjunto mínimo de materia legal que afecta directamente a la profesión informática, de forma diferenciada del resto de ámbitos de trabajo. Es decir, aunque somos conscientes de que un profesional informático puede tener que lidiar, en un momento determinado, con la contratación de personal o la creación de una empresa, no vamos a tratar temas legales de ámbito general, restringiéndonos, por el contrario a los que se refieren al uso del ordenador, tratamiento de la información, gestión del software, seguridad y comercio electrónico, por considerar que estos temas

despiertan mas la atención, son mas novedosos en la legislación tanto nacional como internacional y mas desconocidos, por el momento, en los ámbitos de asesoramiento jurídico.

La materia legal se ha estructurado en los siguientes temas (con indicación de la planificación temporal):

Tema 1.- Protección de la privacidad 3,5 h
- Constitución española (Art. 10 y 18)
- Ley Orgánica Regulación del Tratamiento Automatizado de Datos
- Directivas Comunitarias
- Reglamento de medidas de seguridad
- Ley Orgánica de Datos Personales

Tema 2.- Protección del Software y de las Bases de Datos 1,5 h
- Ley de la propiedad intelectual
- Ley de patentes

Tema 3.- Documentos y comercio electrónico 1,5 h
- Normas de aplicación de facturación telemática
- Normas de Dominios
- Firma digital y certificación de clave
- Comercio electrónico.

Tema 4.- Delitos Informáticos 2 h
- Código penal español

Tema 5.- Ciberlegislación 1 h
- Legislación para Internet. (41 países involucrados en borrador actual)

5.- EJEMPLO DESARROLLO DE UN TEMA: REGLAMENTO DE MEDIDAS DE SEGURIDAD.

En este apartado se presenta el material de apoyo utilizado para la docencia del Reglamento de seguridad. Esta normativa se imparte tras el estudio de la LORTAD y normativa comunitaria relativa a privacidad. A efectos de proporcionar una visión global en este documento, se incluyen trasparencias que introducen los temas anteriores. Como puede apreciarse se ha intentado facilitar el posicionamiento crítico del alumno, resaltando puntos especialmente conflictivos y reflexiones sobre el efecto de la perdida de privacidad. Se presenta la terminología empleada por la ley y los niveles de seguridad que esta contempla. Se pretende facilitar la adquisición de aspectos de la ley, especificando de forma estructurada los requisitos establecidos en cada uno de los niveles de seguridad.

Legislación:

- Convención de Schengen: 1985 para la supresión gradual de controles en las fronteras. España se adhiere 1988/90.
- LORTAD: Ley Orgánica para la Regulación del Tratamiento Automatizado de Datos personales 5/1992 BOE 263, de 31/10/92
  Reglamento medidas Seguridad. RD994/99. BOE 25/06/99
- DIRECTIVAS COMUNITARIAS 95/46/CE protección personas físicas/libre circulación datos (ámbito mercado común, mas ligera que LORTAD y 97/66/CE telecomunicaciones.
- LOPD: Ley Orgánica de protección de datos de carácter Personal 15/99 BOE 14/12/99

3

Reglamento medidas seguridad

- Ambito
- Personas implicadas
- Tipos de medidas
- Niveles de seguridad
- Documento seguridad
- Control acceso
- Identificación y autenticación
- Gestión y distribución de soportes
- Telecomunicaciones
- Notificación y gestión incidencias
- Copias de respaldo
- Auditoría

4

RMS: Ambito

- Establece medidas TECNICAS Y ORGANIZATIVAS
- Afecta a: Fich.Automat., locales, hard.soft, personas
- Seguridad acceso red ≡ acceso local ?? (art.5)
- Tratamiento fuera ubicación necesita autorización

5

RMS: Personas implicadas

- Afectado: Persona física titular de datos
- Responsable del FICHERO: Persona física o jurídica, pública o privada que decide fin,contenido, uso y trato.(a3 LORTAD)
- Encargado del TRATAMIENTO: Persona física o jurídica determina fines y medios (art2. DC 95/46/CE)
- Responsable de SEGURIDAD: Persona(s) responsables de coordinar y controlar medidas de seguridad (reg seguridad)
- Usuarios (trabajadores del sector)

6

RMS: Tipos de medidas

- TECNICAS
  - Control acceso lógico
  - Id+Autentificación
  - Gestión soportes
  - Copias respaldo
  - Telecominicaciones
- ORGANIZATIVAS
  - Documento de seguridad
  - Función/oblig.del personal
  - Responsable seguridad
  - Control acceso físico
  - Distribución soportes
- Notificación y gestión incidencias
- Auditoría

7

RMS: Niveles seguridad requeridos

Alto

Medio

Básico
- Identificativos
- Características personales
- Circunstancias Sociales
- Académico-profesional
- Empleo y carrera
- Información comercial
- Transacciones
- SI NO => PERFIL

Obtener perfil: infracciones penales y administrativas

Hacienda Financieros Solvencia

Ideología religión salud vida sexual

Fines policiales

RMS: Documento de seguridad

- De obligado cumplimiento
- Contenido:
  - detalle de recursos protegidos
  - medidas, normas, reglas, estándares
  - funciones y obligaciones del personal
  - Estructura ficheros y programas que los tratan
  - Procedimiento Not, gestión y respuesta incidencias
  - Política copias respaldo
- Control verificación cumplimiento
- Revisión periódica, adecuación a normativa
- Responsable de seguridad

RMS: Control acceso físico

- acceso a locales sólo personal autorizado

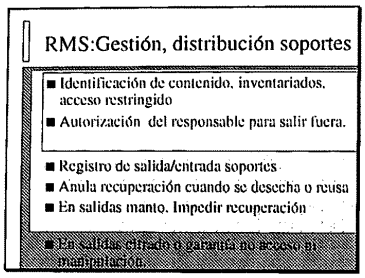
RMS: Control acceso lógico

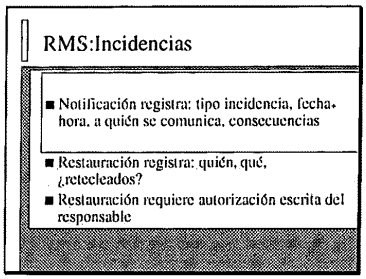
- Sólo a datos necesarios
- Control soft de quién, a qué, para qué
- Especifica quien derecho a conceder derechos

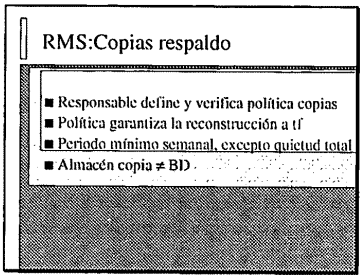
- accesos concedidos/denegados
- qué registro accedido por quién
- no desactivar diario
- conservar diario 2 años
- informe periódico del diario

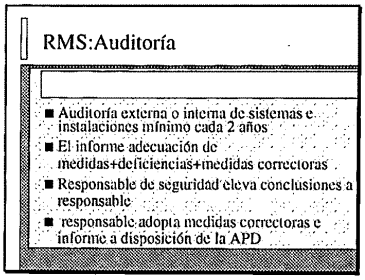
RMS: Identificación y Autentificación

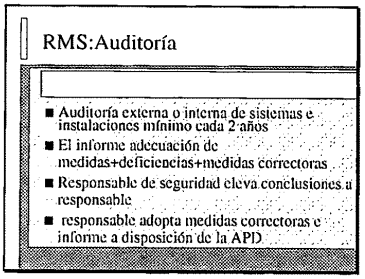
- Matriz accesos y proced. Autenticación en ds.
- Procedimiento de asignación, distribución y almacén de contraseña en ds.
- Cambio periodico de contraseña.
- Almacén ininteligible contraseñas vigentes
- Identificación inequívoca y personalizada
- Limitación de nº accesos no autorizados

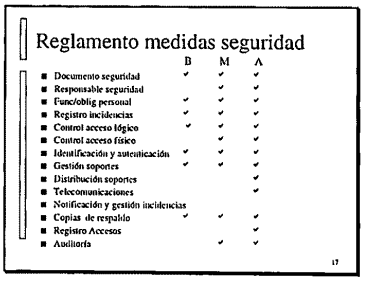










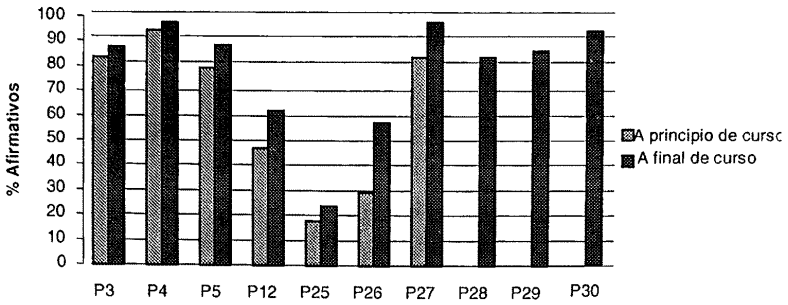


6.- RESULTADOS DE LA EXPERIENCIA.

Para evaluar la incidencia de la docencia de aspectos éticos y legales de la profesión se analiza la experiencia realizada con la asignatura Protección de la Información impartida a alumnos de 3er curso de Ingeniería Técnica Informática y 3er curso de Ingeniería Técnica de Telecomunicaciones de la Universidad de Alcalá en el segundo semestre del curso 98-99.

a) grupo muestral: Se pasaron dos encuestas en horas de clase, una al principio (54 alumnos) y otra al final del semestre (43 alumnos). Entendemos que la disminución en el número de alumnos en la segunda corresponde a la tónica general, en todas las asignaturas, con la proximidad de los exámenes. La edad media de los alumnos es de 22 años y hay un reducido número de mujeres.

b) Análisis de resultados: La figura I refleja los cambios en la valoración que hacen los alumnos sobre el efecto social del uso del ordenador.



P3 El uso del ordenador en un determinado entorno ¿Puede condicionar las relaciones interpersonales dentro del mismo? No[], Si[]
P4 El empleo del ordenador no es ni bueno ni malo intrínsecamente, pero puede tener efectos nocivos o beneficiosos en la sociedad. FALSO[], VERDADERO[].
P5 ¿Crees que todo profesional, antes de abordar un proyecto, tiene la obligación de evaluar si éste va a resultar beneficioso o perjudicial al entorno social al que va destinado (mas concretamente a las personas que lo componen)? NO[], Si[]
P12 ¿Existen aspectos éticos o legales relacionados con el uso del ordenador pero no relacionados con la protección de la información? NO[], Si[], ¿cuales?
P25 ¿Las relaciones con tu familia serían distintas si no tuvieras ordenador en casa? Si[], No[]
P26 ¿Las relaciones con tus padres y hermanos serían distintas si no tuvieras TV en casa? Si[], No[]
P27 ¿Puede resultar peligroso, o hacer algún daño a alguien el que divulguen sus datos? Si[], No[]
P28 ¿Estas ahora mas atento y receptivo a noticias relacionadas con manipulación ilegal de información, efectos sobre la sociedad, etc? Si[], No[]
P29 ¿Piensas que probablemente tus reflexiones afectarán a la forma de enfrentar los dilemas que se puedan presentar en tu ejercicio profesional? Si[], No[]
P30 ¿Has adquirido una idea general del estado actual de la legislación española en materia de protección de la información? Si[], No[]

Figura I . Valoración del efecto del ordenador sobre la Sociedad

Se aprecia una mayor concienciación de su repercusión en las relaciones interpersonales (pregunta P3) y los potenciales efectos nocivos (pregunta P4). Un cambio mayor se aprecia sobre la conciencia de su responsabilidad profesional, pasando del 79,2% al 88,4% las respuestas afirmativas (P5). Después del curso los alumnos se muestran también más conocedores de su responsabilidad sobre otras facetas de su ejercicio profesional no directamente relacionadas con la protección de la información, como lo demuestra la variación en la respuesta a la pregunta P12, pasando de una mayoría de respuestas negativas a principio de curso (solo 46% de respuestas afirmativas), a mayoría de positivas (62,5%)al final del mismo.

Después del curso hay un ligero aumento en el reconocimiento de que las relaciones familiares están afectadas por el uso del ordenador pasando de un 17,8% a un 23,8% las respuestas afirmativas y un considerable aumento, casi de un 30%, en el reconocimiento del efecto de la TV sobre las relaciones familiares (ver P25 y P26). Respecto a la pregunta de sí puede resultar peligroso divulgar datos personales (P27) se aprecia un considerable aumento, pasando de un 83,72 % a un 97,5 % los alumnos que contestan afirmativamente. A final de curso se 83% se reconoce mas atento a esta problemática P(28), el 86% considera que las reflexiones realizadas durante el curso afectaran a su ejercicio profesional P(29) y el 92,1% consideran haber adquirido una idea general del estado actual de la legislación (P30).

8.- BIBLIOGRAFIA.

J. Rachels. *The elements of Moral Philosophy*. 2nd ed Mc Graw-Hill New York. 1993.

J.M. Shepard, M. G. Goldsby y V. W. Gerde. Teaching Business Ethics Through Literature. Teaching Business Ethics 1:33-51, 1997.

R.D. 994/1999 de 11 de junio. Reglamento de medidas de seguridad. B.O.E. 25-6-99, pags 24241-24245

I. Hernando. Contratos Informáticos. Librería Carmelo. 1995

PROBLEMÁTICA DE LOS CONTENIDOS INTERDISCIPLINARES DE TECNOLOGÍAS DE LA INFORMACIÓN Y DE ADMINISTRACIÓN Y DIRECCIÓN. EL CASO DEL PRESUPUESTO DE PROYECTOS DE SOFTWARE.

José Mª. Torralba

UPV (Universidad Politécnica de Valencia)
EUI y FI (Escuela Universitaria y Facultad de Informática)
FADE (Facultad de Administración y Dirección de Empresas)
e-mail: jtorral@omp.upv.es

Resumen: Se plantea la problemática de la docencia de las materias interdisciplinares, en concreto de Tecnologías de la Información junto con Administración y Dirección. Se aplican los planteamientos que se realizan a la materia interdisciplinar "Presupuesto del proyecto de desarrollo de software a medida y adaptado".

1.- INTRODUCCIÓN.

Las titulaciones técnicas se caracterizan por necesitar una importante componente de formación interdisciplinar, esto es, de materias que se encuentran en la intersección de dos o más disciplinas. Las titulaciones de Informática, como carreras técnicas, participan también de esta característica[35].

Algunas Universidades intentan tímidamente aumentar la formación multidisciplinar, por ejemplo mediante la oferta de titulaciones dobles, de forma que el estudiante cursa simultáneamente dos titulaciones que tienen existencia independiente[36]. La motivación puede derivar de que hay demanda para estas formaciones multidisciplinares[37]. Posiblemente empezaron estas innovaciones en las Universidades privadas, aunque ahora ya están instituidas también en las públicas.

En el ámbito que nos planteamos, de intersección entre las Tecnologías de la Información (en adelante, TI) y la Administración y Dirección, también existe preocupación. Algunos indicadores pueden ser los siguientes:

[35] Una aproximación a la multidisciplinariedad de las titulaciones en Informática, se muestra a través de la relación de Áreas de Ponencias de JENUI'2000. Además de estas áreas puede haber otras.
[36] Un ejemplo bastante extendido es la unión de la Licenciatura en Administración y Dirección de Empresas con la Licenciatura en Derecho; otro ejemplo es la Licenciatura en Economía con la Licenciatura en Derecho. Se programan para una duración de un año más que las carreras independientes.
[37] La mayor demanda podría ser no tanto por saber más materias (multidisciplinariedad o pluridisciplinariedad), sino por saber mejor lo que está entre diferentes disciplinas (interdisciplinariedad).

- La titulación de Ingeniero Técnico en Informática de Gestión podría cubrir los dos grandes ámbitos disciplinares referidos.

- En la Universidad Politécnica de Valencia se ha puesto en marcha, en el curso 1997/98, la Licenciatura en Administración y Dirección de Empresas, que tiene una importante dotación relativa de créditos de TI[38].

- Un informe que solicitó la Generalitat Valenciana a una Consultora Irlandesa, hacia 1990, recomendaba la creación de una carrera mixta de TI y de Administración y Dirección de Empresas.

- R. Camps (1994) refiere una titulación americana en la que las TI y la Administración y Dirección están en una proporción parecida.

Es pues la multidisciplinariedad una realidad, que en algunos casos posiblemente se quiere potenciar, pero tiene su problemática docente, lo que motiva esta comunicación.

a) Objeto.

Se plantea la problemática de la docencia de las materias interdisciplinares, en concreto de TI junto con Administración y Dirección. Los planteamientos que se realizan, se aplican a los contenidos de la materia interdisciplinar "Presupuesto del proyecto de desarrollo de software a medida y adaptado".

2.- PROBLEMÁTICA DE LA DOCENCIA INTERDISCIPLINAR.

Para presentar sintéticamente la problemática referida, haremos mención a los siguientes aspectos:

- Dificultad al elaborar el plan de estudios
- La adscripción a áreas de conocimiento
- Decisión sobre quien asume la docencia
- Cómo se imparten

a) Dificultad al elaborar el plan de estudios.

Los contenidos interdisciplinares pueden pertenecer a distintas áreas de conocimiento, por lo que pueden ser vistos como solapes potenciales de un área con otra, lo que puede inducir, para evitar problemas, a no defenderlos. Los contenidos en cuestión pueden pues no impartirse en ocasiones.
Los contenidos menos interdisciplinares, por el contrario, no tienen el problema anterior; y pueden tener el incentivo de que se conocen mejor, existe más personal formado, hay más documentación, etc.

[38] Véase, por ejemplo, nuestra comunicación a JENUI'99: "Diversas aproximaciones a la formación en Informática de Gestión en la Universidad Politécnica de Valencia".

b) La adscripción a áreas de conocimiento.

Cuantas más áreas relacionadas aparezcan, mejor se cubrirán potencialmente los aspectos de la interdisciplinariedad[39]. Es un primer momento en que conviene dejar la puerta abierta para que después se pueda colaborar entre distintas áreas para dar materias interdisciplinares.

c) Decisión sobre quien asume la docencia.

La docencia tiene que ser asumida por algún área de conocimiento y se plantean diversas alternativas:

- Que la asuma una sola área[40]:
  - La más directamente relacionada
  - La que tiene más capacidad de negociación, etc.

- Que se asuma por más de un área[41] con la finalidad principal de:
  - Coordinar la docencia y cubrir mejor los aspectos interdisciplinares
  - Distribuir los créditos42, etc.
  -

En el caso de que participen varios departamentos, áreas, y unidades docentes, se plantea la cuestión de la coordinación, que no siempre es tarea fácil[43].

Otro aspecto es el de las características del profesorado que se encarga de estos contenidos interdisciplinares: Antigüedad, experiencia, perfil, etc.

En relación con el perfil, y para las materias que nos interesan, caben distintas posibilidades:

- Experto en TI que se introduce en Administración y Dirección
- Experto en Administración y Dirección que se adentra en TI.

Para estas situaciones pueden jugar un importante papel los Profesores Asociados, dedicados a tiempo parcial a la docencia, ya que:

- Pueden estar desempeñando actividades que impliquen la interdisciplinariedad
- Pueden mantener actualizada la formación general de la titulación, etc.

---

[39] En las titulaciones de: A) Ingenierías de Obras Públicas, y B) Ingeniería de Caminos, Canales y Puertos, de la UPV, se ha optado por un número alto de áreas; este es el caso de materias relacionadas con la Administración y Dirección.

[40] Por ejemplo, las materias de Gestión de Proyectos Informáticos han sido asumidas en la EUI de la UPV por el área de Organización de Empresas; sin embargo no es así en otras universidades, como por ejemplo en la EUI y la FI de la Universidad Politécnica de Madrid en que pertenecen al área de Lenguajes y Sistemas Informáticos.

[41] La asignatura de Sistemas Operativos en la EUI de la UPV se ha llegado a impartir entre dos departamentos: 1) Sistemas Informáticos y Computación (con las áreas: a) Lenguajes y Sistemas Informáticos, y b) Ciencias de la Computación e Inteligencia Artificial), y 2) Ingeniería de Sistemas, Computadores y Automática (con las áreas: a) Arquitectura y Tecnología de Computadores, y b) Ingeniería de Sistemas y Automática).

[42] Este es el caso de la asignatura Proyectos Informáticos, de 15 créditos, todos ellos de carácter práctico, basada en la materia troncal Sistemas Informáticos, de la titulación de Ingeniería en Informática de la UPV, que se reparten los créditos entre varios departamentos en función de los proyectos dirigidos en los tres cursos anteriores, pero que no existe coordinación entre los departamentos.

[43] En la Escuela T. S. de Ingenieros de Caminos, Canales y Puertos de la UPV se celebraron reuniones de coordinación entre unidades docentes de distintas áreas y departamentos, en el curso 1996/97, pero puede que estas buenas prácticas no estén muy generalizadas.

d) Cómo se imparten.

Poner en marcha estas materias cuesta mucho, por lo que ya se ha indicado. Por ello, suele ser crítica la disponibilidades de materiales didácticos adaptados al enfoque interdisciplinar.
Todo lo indicado quedaría como muy teórico si no se aplicase a un caso concreto, lo cual se hace en el apartado siguiente.

3. EL CASO DEL PRESUPUESTO DE PROYECTOS SOFTWARE A MEDIDA O ADAPTADO.

Un componente importante de cualquier proyecto de ingeniería es, como se sabe, el Presupuesto, pues es necesario su conocimiento para determinar las viabilidades presupuestaria y financiera (y económico-social en el caso de proyectos públicos).

La realización del presupuesto de un proyecto de desarrollo de software, a medida o adaptado, exige un proceso que puede estructurarse en las actividades que muestra el cuadro nº 1.

ACTIVIDADES para realizar el presupuesto	algunos ámbitos científicos en que se encuadra TRADICIONALMENTE la activi-dad
MEDICIÓN	INGENIERÍA DEL SOFTWARE (IS)
DETERMINACIÓN DE LA PRODUCTIVIDAD	IS ADMINISTRACIÓN Y DIRECCIÓN (AD)
ESTIMACIÓN DEL ESFUER-ZO	IS, AD
DETERMINACIÓN DE PRE-CIOS DE MERCADO	COMERCIALIZACIÓN E INVESTIGA-CIÓN DE MERCADOS (CIM)
CÁLCULO DE COSTES	CONTABILIDAD DE COSTES Y DE GESTIÓN (CCG)
PRESUPUESTO	CCG
PRECIO DE OFERTA AL CLIENTE	AD, CIM

Cuadro nº. 1.- Algunos ámbitos científicos en que se encuadran tradicionalmente las actividades del presupuesto del proyecto de software a medida y adaptado.

Los ámbitos científicos y las áreas de conocimiento que necesariamente tienen que utilizarse son varios, lo que constituye un caso representativo de materia interdisciplinar.

El Presupuesto suele impartirse dentro de una materia más amplia de Gestión de Proyectos Informáticos[44], la cual, como ya se ha indicado, podemos encontrarla tanto en el área de Lenguajes y Sistemas Informáticos como en la de Organización de Empresas, etc.

Este no es un problema privativo de informática, sino que se presenta también en otras tecnologías[45], en cada una de las cuales se le da salidas que pueden ser distintas.

A pesar de ser una materia que forma parte de la generalidad de los manuales de Ingeniería del Software, ha costado que llegara a consolidarse en los planes de estudio, a lo que puede haber contribuido precisamente su condición de interdisciplinariedad tan marcada[46].

a) La aportación del área de Proyectos de Ingeniería.

Como se conoce, hay un área de Proyectos de Ingeniería que está implantada en todas las Ingenierías "tradicionales", pero que no ocurre así en algunas de las nuevas, siendo este el caso de Informática, al menos en varias Universidades.

Se trata de un área con un enfoque interdisciplinar, que coincide con el de los Proyectos Fin de Carrera (PFC) tradicionales, como simulación de trabajos profesionales reales típicos, por lo que desempeña un papel fundamental para los PFC y para la formación de los técnicos orientados al mercado de trabajo más característico.

Este área se ocupa precisamente del Presupuesto de los proyectos de Ingeniería.

Consideramos que falta en Informática afrontar más decididamente estos temas interdisciplinares y una solución existe desde hace mucho tiempo: el área de Proyectos de Ingeniería[47].

BIBLIOGRAFÍA

ANDREU, J. (1996): Plan Integral de estudios de Ingeniería Civil en la Escuela Técnica Superior de Ingenieros de Caminos, Canales y Puertos de la Universidad Politécnica de Valencia. Actas de las II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías. Ed. Universidad Politécnica de Madrid. Madrid.

CAMPS, R. (1994): Ensenyem informática, pero quina?. Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI'94), Barcelona.

[44] Se viene comentando en ediciones anteriores de JENUI, la conveniencia de que hubiera un área de Ponencias de Proyectos que es de contenidos interdisciplinares.

[45] En la titulación de Arquitectura Técnica, el Presupuesto suele impartirse por el área de Construcciones Arquitectónicas; en la Universidad Politécnica de Cataluña y en la UPV se imparte por el área de Organización de Empresas en Arquitectura; en las titulaciones de Ingeniería Técnica de Obras Públicas, por el área de Proyectos de Ingeniería; etc.

[46] En la UPV, la mayoría de los Proyectos Fin de Carrera presentados de 1985 a 1995 no tenían Presupuesto.

[47] Hay también un área de Proyectos Arquitectónicos que cumple análoga función en las Escuelas de Arquitectura.

DE LA PLAZA, S. (199): Normativa de planes de estudio. Actas del VI Congreso Universitario de Innovación Educativa en las Enseñanzas Técnicas. Las Palmas de Gran Canaria.

MICHAVILA, F. (1998): La Reforma necesaria. Conferencia. Actas del VI Congreso Universitario de Innovación Educativa en las Enseñanzas Técnicas. Las Palmas de Gran Canaria.

TORRALBA, J. Mª., OLTRA, J. V., FERRER, R., SÁNCHEZ, A., CABRERA, O. y R. ARENAS (1999): Diversas aproximaciones a la formación en Informática de Gestión en la Universidad Politécnica de Valencia. Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI'99), La Almunia de Dª. Godina, Zaragoza.

UNIVERSIDAD POLITÉCNICA DE VALENCIA (2000): Programa EUROPA. Ed. Universidad Politécnica de Valencia, Valencia.

ÁREA DE INTELIGENCIA ARTIFICIAL

Responsable: Dr. Josep Mª Garrell
(Universitat Ramón Llull)
E-mail: josepmg@salleURL.edu

INTRODUCCIÓN A LAS TÉCNICAS DE APRENDIZAJE AUTOMÁTICO (PARTE I): ESTRUCTURACIÓN DE UN CURSO DOCENTE

Elisabet Golobardes, Ester Bernadó, Xavier Llorà, Josep M. Garrell, Maria Salamó, Joan Camps y David Vernet

Grupo de Investigación en Sistemas Inteligentes, Departamento de Informática, Enginyeria i Arquitectura La Salle, Universitat Ramon Llull, Passeig Bonanova 8, 08022 - Barcelona
e-mail:{elisabet,esterb,xevil,josepmg,mariasal,joanc,dave}@salleURL.edu

RESUMEN: En este artículo se presentan los Seminarios del Grupo de Investigación en Sistemas Inteligentes de *Enginyeria i Arquitectura La Salle* de la *Universitat Ramon Llull*, estructurados como un curso de introducción a las técnicas de aprendizaje automático. Este curso ofrece un marco común a los alumnos que cursan asignaturas de libre elección, o bien realizan sus trabajos y/o proyectos de final de carrera, dentro de nuestro grupo de investigación.

1.- INTRODUCCIÓN

Los Seminarios del Grupo de Investigación en Sistemas Inteligentes de Enginyeria i Arquitectura La Salle (EALS), de la Universitat Ramon Llull (URL) pretenden ser un curso que aglutina diferentes asignaturas de libre elección de la Ingeniería Técnica de Informática de Sistemas y de la Ingeniería en Informática. Al mismo tiempo, pretende establecer un marco común tanto los Trabajos de Final de Carrera como los Proyectos Final de Carrera realizados dentro del grupo. Finalmente, también incluye a los alumnos "colaboradores", en general, con el Grupo de Investigación en Sistemas Inteligentes.

Como se puede observar, el perfil y los conocimientos de los distintos alumnos agrupados bajo el paraguas de los Seminarios del Grupo de Investigación en Sistemas Inteligentes es muy variado. Así pues, para entender como se adecua el temario a cada tipo de estudiante, previamente presentaremos el trabajo que realiza el Grupo de Investigación en Sistemas Inteligentes, y a quién van dirigidos los seminarios dentro del contexto que definen los planes de estudio de EALS.

Una vez descrito el contexto de los Seminarios del grupo, pasaremos a presentar sus objetivos, el planteamiento diseñado, la planificación global de un curso, los criterios de evaluación y algunas conclusiones.

2.- GRUPO DE INVESTIGACIÓN EN SISTEMAS INTELIGENTES

El grupo de investigación en Sistemas Inteligentes (GrupoSI) se enmarca dentro del Grupo de Investigación en Inteligencia Artificial de la Universidad Ramon Llull. Las principales actividades del GrupoSI se basan en las distintas técnicas del paradigma del aprendizaje automático (o aprendizaje artificial). Las líneas básicas de trabajo discurren dentro de las familias del *aprendizaje evolutivo* (algoritmos genéticos, sistemas de clasificación, programación genética y vida artificial), del *aprendizaje analógico* (razonamiento basado en casos), del *aprendizaje conexionista* (redes neuronales), del *aprendizaje inductivo* (principalmente árboles de decisión), y de distintos *híbridos* entre las diferentes técnicas. Nuestro trabajo se ha centrado en resolver problemas de clasificación, de optimización y/o de diagnóstico. Concretamente, el trabajo de investigación del grupo se ha desarrollado, mayoritariamente, dentro del marco que ofrecen distintos proyectos subvencionados en los cuales estamos involucrados:

- Aprendizaje artificial para la predicción y la diagnosis (Funitec)
- Diagnosis automática de cáncer de mama con técnicas de IA (Funitec)
- Generación de modelos acústicos de calles de una ciudad con técnicas de IA (Funitec)
- Generación automática de estrategias de demostración para sistemas paralelos (Funitec)
- Participación en la *Xarxa Temàtica d'Intel· ligència Artificial* de la Generalitat de Catalunya
- Implementación y estudio de herramientas de inteligencia artificial aplicadas a plataformas de enseñanza abierta a distancia y a las redes ATM que las soportan (CICYT/TEL98-0408)
- *Automatic classification of long bone fractures from X-Ray images using the Müller's CCF system* (Maurice E. Müller Fundation)
- Sistema de predicción del riesgo de cáncer de mama basado en técnicas de visión por computador y de aprendizaje artificial automático (FIS 00/0033-02)

3.- ¿A QUIÉN VAN DIRIGIDOS LOS SEMINARIOS?

Enginyeria i Arquitectura La Salle ofrece 7 titulaciones técnicas (una de propia: graduado técnico en multimedia) y 5 de superiores (una de propia: graduado superior en multimedia). Concretamente, se pretende que las titulaciones de la Enginyeria (6+4) tengan un primer curso muy común y que, al mismo tiempo, a partir de asignaturas optativas y de libre elección un estudiante de una titulación pueda adquirir fácilmente un perfil de otra titulación. Una de las maneras de ofrecer esta interdisciplinariedad, es a partir de unas asignaturas de libre elección (denominadas intensificación y especialización de cada sección de la Escuela) que el estudiante puede cursar "colaborando" con los distintos grupos de trabajo de los diferentes departamentos de EALS. Aunque la participación puede ser de alumnos de carreras distintas, los estudiantes que colaboran con nuestro grupo tienen un perfil mayoritariamente informático. Así pues, desde esta perspectiva, en nuestro grupo pueden "colaborar" estudiantes de diferentes titulaciones, principalmente cursando las asignaturas de libre elección, realizando su trabajo o proyecto de final de carrera (ver tablas 1 y 2)[48] o, simplemente, colaborando con nosotros.

[48] El GrupoSI desarrolla su actividad dentro del Departamento de Informática y, más concretamente, sus áreas de interés son afines a las asignaturas de la sección de Tecnología del Software. Por este motivo, las intensificaciones y especializaciones son en Tecnología del Software.

Tabla 1. Asignaturas correspondientes al plan de estudios de Ingeniería Técnica en Informática de Sistemas, EALS, URL

Asignatura	Tipo	Créditos
Intensificación en Tecnología del Software	Libre elección	9
Especialización en Tecnología del Software	Libre elección	9
Trabajo Final de Carrera (TFC)	Obligatoria	8

Tabla 2. Asignaturas correspondientes al plan de estudios de Ingeniería en Informática, EALS, URL

Asignatura	Tipo	Créditos
Intensificación en Tecnología del Software	Libre elección	9
Especialización en Tecnología del Software	Libre elección	9
Proyecto Final de Carrera (PFC)	Obligatoria	8

Con el propósito de agrupar todos estos estudiantes, desde el GrupoSI, elaboramos los *Seminarios* a partir de los cuáles pueden formarse en los principales temas en los que trabajamos y, así, pueden desarrollar sus créditos de libre elección, o bien su TFC o PFC, o bien, su colaboración. Pero estos Seminarios se tendrán que adecuar al perfil de cada estudiante.

4.- CONTENIDO DE LOS SEMINARIOS

Los Seminarios del Grupo SI se centran en las áreas siguientes: algoritmos genéticos (GA), redes neuronales (NN), programación genética (GP), razonamiento basado en casos (CBR) y sistemas clasificadores (CS). Concretamente, les damos una introducción sobre:

- *Computación Evolutiva* introduciendo tres de sus posibles vertientes para la clasificación automática: GA, GP y CS. También se les muestra alguna herramienta propia desarrollada dentro del GrupoSI para trabajar con GA [1,2,3,4,9].

- *Redes Neuronales*. Se pretende realizar una introducción de éstas, y explicar, básicamente, las más clásicas para poder entender el problema de la no linealidad: el Perceptrón, la Adaline, la Madaline y la Backpropagation[5]. Se les muestra alguna herramienta propia del GrupoSI, o bien, alguna herramienta de libre distribución (e.g. TRAJAN 2.0).

- *Razonamiento Basado en Casos*. Ofrecemos una breve introducción a la filosofía del CBR, y detallamos mejor el CBR encarado a resolver problemas de clasificación. Concretamente, les explicamos algunas soluciones que hemos dado para resolver el problema del diagnóstico de tejidos de glándulas mamarias [3,4,7], o bien, para el diagnóstico de mamografías a partir de microcalcificaciones [10,11,12].

En las exposiciones más prácticas procuramos que participen alumnos de otros años "antiguos" siguen colaborando con nuestro grupo.

5.- PLANTEAMIENTO DE LOS SEMINARIOS

Los Seminarios se han diseñado distinguiendo dos tipos de alumnos: (1) los colaboradores (colaboradores o alumnos que cursan las correspondientes asignaturas de libre elección); y (2) los proyectistas (alumnos que realizan su TFC o PFC). Concretamente, presentamos la estructura de los Seminarios para un alumno que trabaja con nosotros por primera vez[49]. Estos Seminarios se estructuran en tres partes que, más o menos, corresponden a los trimestres naturales.

a) Parte I: formación

La primera parte corresponde a una etapa de formación básica en los temas que se han expuesto en la sección 4. Para realizar esta formación, se organizan un conjunto de seminarios, impartidos por los propios miembros del GrupoSI, más la participación de algunos "antiguos" colaboradores. La asistencia a estos seminarios es obligatoria para todos los estudiantes (tanto los de tipo (1) como los de tipo (2)). En estos seminarios se les proporcionará o se les indicará la bibliografía complementaria.

Al finalizar estos seminarios, se les realiza un *test* con un doble propósito. Por un lado, para motivar que los alumnos se vayan formando en los temas impartidos. Y por otro lado, para disponer de una primera "nota". Este test incluye conocimientos básicos y lo suele aprobar cualquier estudiante que haya seguido los seminarios (ver sección 7).

b) Parte II: presentación de trabajos

Después de los seminarios, los colaboradores (1) realizan un trabajo que consiste en la búsqueda bibliográfica sobre un tema concreto, relacionado con alguno de los temas presentados en los Seminarios. Este tema se adecuará al nivel de cada estudiante (primer o segundo ciclo), y lo tendrán que presentar y exponer.

Por lo que respecta a los proyectistas (2), durante este período se concreta su proyecto y se le asigna un ponente (un miembro del GrupoSI afín al proyecto que va a realizar). A partir de aquí, será éste quien llevará personalmente al alumno.

c) Parte III: competición

Una vez presentados los trabajos, se propone a los colaboradores (1) un problema común que tienen que resolver individualmente o por grupos de dos. Cada grupo decide con qué técnica (GA, CBR o NN) se plantea resolverlo e implementarlo. La presentación del trabajo realizado consiste en: a) entregar una memoria donde se detalla y se justifica la resolución del problema realizada; y b) realizar una ***competición***. Ésto se llevará a cabo sobre un mismo entorno desarrollado por el GrupSI (ver Parte II del artículo presente). "Ganará" el grupo que encuentre la mejor solución y obtenga un mejor rendimiento. Los parámetros concretos del problema no los conocerán hasta el mismo día de la competición. Así pues, deberán tener la "habilidad" de ajustar sus algoritmos *in situ* con tal de ser los mejores.

Cabe destacar que, de los miembros del GrupoSI se designa un responsable para cada nuevo curso, y se asignan "tutores/consultores" para cada área (GA, CS, GP, NN y CBR), a los cuáles los alumnos de

[49] Los alumnos que siguen trabajando con el GrupoSI en los años posteriores, realizan un trabajo más personalizado y ad hoc a su perfil.

los Seminarios pueden acudir en los horarios correspondientes. De la misma manera, los alumnos tienen un soporte telemático de los seminarios, donde pueden encontrar información general, documentación, notificaciones, calendario, etc.

6.- PLANIFICACIÓN GLOBAL

Una vez definidos los contenidos de los Seminarios y el planteamiento de los mismos, realizamos una planificación global de éstos. Concretamente, la tabla 3 muestra, como ejemplo, la planificación que se ha realizado para el curso académico 1999/2000, ya que en EALS los planes de estudio consisten en asignaturas anuales. Podemos destacar la particularidad que los cursos de las titulaciones de primer ciclo se evalúan en tres trimestres, mientras que las titulaciones correspondientes a un segundo ciclo se estructuran de forma cuatrimestral. Así pues, se han planificado los Seminarios respetando los períodos de exámenes de los distintos alumnos.

Tabla 3. Planificación global de los Seminarios del Grupo de Investigación en Sistemas Inteligentes para el curso 1999/2000.

Nº	Semana	Sesión (1 hora)	Sesión (1 hora)
1	18-22 octubre	Presentación / GA (I)	GA (II)
2	25-29 octubre	NN (I)	NN (II)
3	1-5 noviembre	NN (III)	GP
4	8-12 noviembre	CBR (I)	CBR (II)
5	15-19 noviembre	Herramienta GA	Herramienta NN
6	22-26 noviembre	CS	Día de trabajo personal
7	29-3 nov./dic.	Clase de dudas	Test
8 9 10	6-10 diciembre 13-17 diciembre 20-24 diciembre	Período de exámenes de primer ciclo	Período de trabajo personal de los de segundo ciclo
	Fiestas de Navidad		
11 12	10-14 enero 17-21 enero	Presentación de los trabajos de segundo ciclo	Período de trabajo personal de los de primer ciclo
13 14 15	24-28 enero 31/4 enero/febrero 7-11 febrero	Período de exámenes de segundo ciclo	Período de trabajo personal de los de primer ciclo
16 17	14-18 febrero 21-25 febrero	Presentación de los trabajos de primer ciclo	
	marzo a junio	Colaboradores (1): implementación de un problema a resolver	Proyectistas (2): perfilan y definen su proyecto
18	Finales de junio	COMPETICIÓN	

7.- CRITERIOS DE EVALUACIÓN

Los criterios de evaluación que aquí presentamos, corresponden a los que se aplican a los alumnos que cursan una asignatura de libre elección dentro de los Seminarios (Intensificación y/o Especialización en Tecnología del Software). Concretamente, la nota final pretende ponderar los aspectos siguientes:

1. *Parte I: formación.* La asistencia a los seminarios, la participación en los seminarios, y la nota obtenida en el test. La nota del test es decisiva a efectos de decidir si el estudiante continua realizando normalmente los Seminarios. Consideramos que los estudiantes que lo suspenden no tienen ningún interés por el grupo. A este tipo de estudiantes se les proponen otro tipo de trabajos directamente para Setiembre.

2. *Parte II: presentación de los trabajos.* Evaluación del "trabajo-teórico" realizado, y evaluación de la presentación del mismo.

3. *Parte III: competición.* Evaluación de la memoria del "trabajo-práctico" realizado, y evaluación de la propia competición.

8.- CONCLUSIONES

Anteriormente al planteamiento de los Seminarios del GrupoSI (los cuáles se introdujeron por primera vez en el curso académico 1997/1998), los alumnos colaboraban directamente con algún tema concreto y con un profesor/a dado. Las desventajas que esto conllevaba eran que: 1) el alumno no adquiría una visión global de los distintos temas que se trataban dentro del GrupoSI y, al mismo tiempo, no tenía la sensación de estar integrado dentro de un grupo de personas con los mismos intereses; y 2) por parte del profesor/a, éste dedicaba mucho tiempo de su trabajo para un único alumno –o para un grupo muy reducido de personas-, casi equivalente al número de horas que se destinan a una asignatura de libre elección con los mismos créditos. Así pues, los Seminarios –aunque necesitan un trabajo continuo por nuestra parte- creemos que son positivos en distintos aspectos:

1. Los alumnos salen mejor preparados para una futura integración en el GrupoSI, ya que se les dan nociones básicas de diferentes temas en los cuáles estamos trabajando. Aunque su trabajo lo realicen sobre un tema concreto, el planteamiento de los seminarios les permite profundizar en el tema que más les interese.

2. Los alumnos sienten que pertenecen a un grupo de trabajo y esto, en general, los motiva más.

3. Los alumnos –según han comentado ellos mismos- encuentran una motivación complementaria. Una de las claves ha sido precisamente la *competición* que organizamos que, al fin y al cabo, sirve para realizar el mismo trabajo con cierta motivación para encontrar la mejor solución.

4. A los distintos miembros del GrupoSI nos permite organizar mejor nuestro tiempo y participar en los Seminarios con los temas que tenemos más afines.

Globalmente, aunque los Seminarios van evolucionando cada curso académico, creemos que éstos son positivos y constructivos para los alumnos que quieren definir su perfil en Sistemas Inteligentes como forma complementaria a las asignaturas de las titulaciones afines a nuestro grupo [6,8].

BIBLIOGRAFÍA

[1] E. Bernadó, A. Mekaouche, J.M. Garrell, *A Study of a Genetic Classifier System Based on the Pittsburgh Approach on a Medical Domain*, IEA/AIE-99, 1999, Springer Verlag, 175-184

[2] G. Calzada, J.M. Garrell, R. Barti, *ModelGP: A GP-based platform for predictive model construction*, Proceedings of SOCO'99, 1999

[3] J.M. Garrell, E. Golobardes, E. Bernadó and X. Llorà, *Automatic Classification of Mammary Biopsy Images with Machine Learning Techniques*, Proceedings EIS'98, Volume 3: Artificial Intelligence, ICSS Academic Press, 1998, 411-418

[4] J.M. Garrell, E. Golobardes, E. Bernadó and X. Llorà, *Automatic diagnosis with genetic algorithms and case-based reasoning*, Artificial Intelligence in Engineering, Elsevier Science Ltd. Vol. 13, No. 4, October 1999, 367-372, ISSN: 0954-1810

[5] E. Golobardes, E. Pous, M. Román. *Introducció a les xarxes neuronals, presentació de les Backpropagation*, INPUT 10, Febrero 1996, 17-26

[6] E. Golobardes, *¿Dónde están los límites de una introducción a la Inteligencia Artificial?* JENUI'97, Junio 1997, 293-300

[7] E. Golobardes, J.M. Garrell, *Avaluació d'un sistema classificador basat en casos per a la diagnosi de biòpsies de teixit de glàndules mamàries*. Jornades d'Intel· ligència Artificial: Noves Tendències (JIA'97), ACIA No. 12, Tardor 1997, 41-49

[8] E. Golobardes, J.M. Garrell, *¿Qué relación hay entre la asignatura de IA, la investigación y la aplicación de sus tecnologías?*, JENUI'98, Julio 1998, 347-354

[9] X. Llorà, J.M. Garrell, *GENIFER: A Nearest Neighbour based Classifier System using GA*, Proceedings of GECCO'99, 1999, CA: Morgan Kaufmann, pp. 797

[10] X. Llorà, E. Golobardes, M. Salamó, J. Martí, *Diagnosis of microcalcifications using Case-Based Reasoning and Genetic Algorithms*, Proceedings EIS'2000, To appear.

[11] J. Martí, J. Español, E. Golobardes, J. Freixenet, R. Garcia, M. Salamó, *Classification of microcalcifications in digital mammograms using Case-Based Reasoning*, Proceedings of IWDM-2000, To appear.

[12] M. Salamó, E. Golobardes, J.M. Garrell, J. Martí. *Clasificación de microcalcificaciones usando razonamiento basado en casos*, Proceedings CAEPIA'99/TTIA'99, 1999, ISBN: 931170-0-5

INTRODUCCIÓN A LAS TÉCNICAS DE APRENDIZAJE AUTOMÁTICO (PARTE II): ENTORNO DE COMPETICIÓN

Xavier Llorà, Ester Bernadó, Elisabet Golobardes, Josep M. Garrell, Maria Salamó, Joan Camps y David Vernet

Grupo de Investigación en Sistemas Inteligentes, Departamento de Informática, Enginyeria i Arquitectura La Salle, Universitat Ramon Llull, Passeig Bonanova 8, 08022 - Barcelona
e-mail:{xevil,esterb,elisabet,josepmg,mariasal,joanc,dave}@salleURL.edu

Resumen: El presente artículo describe la parte práctica de la asignatura de formación en Aprendizaje Automático, en el entorno de nuestro grupo de investigación. Se estructura como una competición entre distintos grupos de trabajo, lo cual sienta las bases para motivar al alumno y establecer un marco común de comparación de distintas metodologías de Aprendizaje Automático.

1.- INTRODUCCIÓN

La asignatura de introducción a las técnicas de aprendizaje automático es un punto de unión entre investigación y docencia. Permite dar a conocer a los alumnos la investigación que desarrolla el grupo de Sistemas Inteligentes, mediante una metodología docente flexible y muy participativa.

La asignatura consiste en una parte formativa, estructurada en seminarios, y una parte eminentemente práctica cuya finalidad es estimular la curiosidad e interés del alumno. La práctica se basa en dos actividades complementarias. La primera es el estudio bibliográfico y profundización en un tema concreto (relacionado con los temas impartidos en los seminarios). En segundo lugar, los alumnos, divididos en grupos de trabajo, desarrollan un pequeño sistema inteligente aplicado sobre un problema concreto.

Este artículo describe el diseño del problema a resolver por los alumnos, el cual se ha definido sobre dos bases: aplicar las metodologías impartidas en los seminarios y motivar la participación del alumno. En la sección 2 se detallan los aspectos más relevantes de este diseño. A continuación, se presenta el problema que se ha definido así como la plataforma desarrollada. Finalmente, se muestran los resultados y conclusiones de la experiencia.

2.- DISEÑO DE LAS PRÁCTICAS

El diseño de la parte práctica de la introducción a los Sistemas Inteligentes pretende cubrir tres objetivos principales: (1) aplicación práctica de las metodologías tratadas en los seminarios, (2)

incentivar la motivación del alumno y (3) divulgar los resultados a través de una plataforma Web. A continuación se detallan cada uno de estos objectivos.

2.1.- Aplicación práctica de las metodologías tratadas

La principal función de las parte práctica es que los alumnos tomen contacto con la teoría expuesta en los seminarios. Debido a la variedad de técnicas y metodologías tratadas, hay ciertas consideraciones a tener en cuenta.

En primer lugar, la práctica debe permitir la aplicación de todas las metodologías. Recordemos que las metodologías tratadas en los seminarios son muy heterogéneas. Éstas son: redes neunorales (NN), algoritmos genéticos (GA), programación genética (GP), sistemas clasificadores (CS) y razonamiento basado en casos (CBR). Por tanto, la primera labor del alumno es analizar la aplicabilidad de dichas técnicas en el problema definido. A partir de ahí, debe escoger la metodología más apropiada según su criterio y adaptarla convenientemente.

Es interesante, a la vez, que los distintos grupos de trabajo escojan metodologías distintas para la resolución del problema. De esta forma, será posible establecer una comparativa muy didáctica entre ellas.

Otro punto muy importante a tener en cuenta es la complejidad del problema. Por una parte, el problema o entorno que se defina debe tener cierta complejidad, para justificar la aplicación de técnicas de aprendizaje automático. No obstante, debe definirse de forma acotada, acorde al perfil de los alumnos que cursan la asignatura.

2.2.- Incentivar la motivación del alumno

La aplicación práctica de las técnicas vistas en clase es, por si misma, una motivación para el alumno, puesto que ve en profundidad una técnica concreta, aumenta su comprensión de la misma y a la vez verifica su funcionamiento. Pero, además, creemos importante incentivar su participación en las prácticas si definimos un problema que les resulte atractivo, divertido o simplemente, nuevo o distinto a los que normalmente ha realizado en otras asignaturas. En segundo lugar, podemos aumentar el interés del alumno si definimos un entorno de **competición** entre los distintos grupos. De esta forma, si los diferentes sistemas desarrollados pueden entrar en competición directa, cada grupo se esforzará en hacer de su sistema *el mejor.*

2.3.- Plataforma Web

Un objetivo colateral de esta parte práctica es la difusión del entorno de competición que se defina, así como la difusión de los resultados que se obtengan. Por lo que al entorno se refiere, su difusión en una página Web puede ser útil desde distintos puntos de vista: (1) se puede usar como entorno de test para metodologías de aprendizaje automático y (2) dar a conocer la asignatura para los futuros alumnos de la asignatura.

3.- DESCRIPCIÓN DEL PROBLEMA

El problema diseñado se basa en el aprendizaje artificial de un agente (hormiga) el cual tiene que moverse en un mapa bidimensional con el objetivo de buscar comida.

Como primera prueba piloto hemos acotado el problema de la siguiente forma. Las posiciones del mapa pueden estar en tres estados distintos: vacías, con comida o ocupadas con una hormiga. Cada hormiga tiene 8 sensores, correspondientes a las casillas colindantes. La hormiga se mueve en ciclos discretos y tiene que decidir qué movimiento realizar, en función de la percepción de los sensores. Este movimiento está limitado también a una de las ocho casillas que la rodean. El objetivo de la hormiga es comer el máximo posible durante la vida de la misma, la cual se define por un número máximo de movimientos.

Todas las hormigas participantes compiten en un mismo mapa a la vez. En principio, cada grupo compite con una única hormiga, aunque también cabe la posibilidad que cada grupo participe en la competición con más de una hormiga. De este modo, podrían definirse comportamientos de competición entre las hormigas de distintos grupos, así como de colaboración entre hormigas de un mismo participante, pudiendo o no existir comunicación explícita entre ellas. Este último comportamiento se deja al libre criterio de cada grupo.

El marco de este problema sugiere muchas variantes posibles (algunas de ellas se pueden encontrar en la bibliografía). Por ejemplo, el entorno Woods usado por Wilson [6] se caracteriza por mapas con obstáculos (además de comida) y el objetivo de la hormiga es alcanzar la comida lo más rápido posible. En este caso, no hay competición ni colaboración entre hormigas. Tan sólo hay una hormiga por mapa. Otros entornos, como el descrito por Potter [5], dos robots compiten por la comida que deben alcanzar para sobrevivir y reemplazar así la pérdida de energía debida al movimiento.

En nuestro caso, el entorno que hemos definido es suficientemente amplio para que los alumnos puedan desarrollar sistemas relativamente simples, así como sistemas mucho más complejos que incorporen por ejemplo, comportamientos cooperantes, memoria, etcétera. Además, la implementación del entorno contempla la posible modificación o ampliación del mismo, introduciendo así algunas de las variantes anteriormente comentadas.

3.1.- El trabajo de los alumnos

El trabajo realizado por los alumnos se ve reflejado en la implementación de su hormiga, que les permitirá tomar parte en la correspondiente competición. Debido a la finalidad divulgativa del trabajo a través de una plataforma Web, se han fijado ciertas restricciones.

Concretamente nos referimos a que la hormiga se ha modelado como una *interfaz* (API) la cual modela el comportamiento de la hormiga anteriormente descrito. Básicamente, los estudiantes tienen que implementar la siguiente *interfaz* Java [1,2]:

```
public interface Ant extends Remote {
      void initialize () throws RemoteException;
      void newMap () throws RemoteException;
      void born () throws RemoteException;
      int  move ( char [] baEnviroment ) throws RemoteException;
      void cellContent ( char cCell ) throws RemoteException;
      void die () throws RemoteException;
      void endMap () throws RemoteException;
      void endSimulation () throws RemoteException;
}
```

Dicha *interfaz* modela completamente el comportamiento de la hormiga en el entorno, permitiendo una integración directa del trabajo de los alumnos en la plataforma.

3.2.- Plataforma

Una vez los alumnos han diseñado e implementado su hormiga, es el momento de tomar parte en la competición. Dicha competición se realiza sobre una plataforma específicamente diseñada para dicho objetivo. El funcionamiento básico de la misma se puede resumir en las siguientes fases:

- Configuración del entorno en el cual las hormigas vivirán (*carga de mapas*)
- Incorporación de las hormigas realizadas por los alumnos
- Selección del modo de competición
- Simulación (competición propiamente dicha)

Configuración del entorno

Consiste en la carga de los mapas que se utilizarán para realizar las simulaciones. Por cada mapa se indica el número de simulaciones, el número máximo de pasos de las hormigas y el peso asociado a la complejidad del mapa.

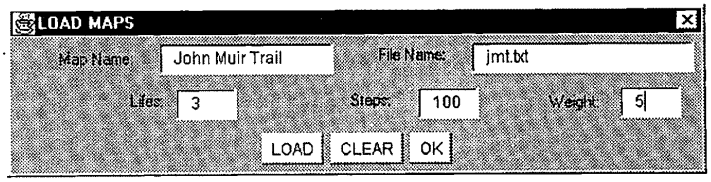


Figura 1 Carga interactiva de mapas

En la figura 1 se carga el mapa John Muir Trail [4], donde se definen tres vidas, cien pasos y un peso asociado de valor 5.

Carga de las hormigas

Cuando el entorno ya se ha configurado, se procede a la carga de las hormigas. Del conjunto de hormigas disponibles, se pueden seleccionar cuáles de ellas participan en la competición, así como el número de las mismas.

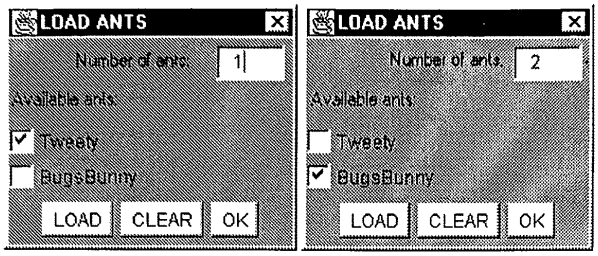


Figura 2 Carga de hormigas

En el ejemplo anterior se cargan tres hormigas: un *Tweety* y dos *BugsBunny*. Como se puede apreciar, la plataforma ofrece la máxima flexibilidad en la configuración del cuadro de participantes, permitiendo así distintos escenarios de competición.

Selección del modo de competición

Una vez seleccionadas las hormigas y el número de participantes de cada una de ellas, se pasa a la elección de la modalidad de competición. Hay dos modalidades. La primera consiste en que cada hormiga participe aisladamente, es decir en un mapa propio. De esta forma se puede apreciar el comportamiento individual de cada hormiga. La segunda modalidad ubica todas las hormigas en un único entorno –mapa–. Dicha variante busca la competencia entre hormigas contrarias –en el ejemplo anterior, entre *Tweety* y *BugsBunny*– y la cooperación entre hormigas homónimas –por ejemplo entre las dos *BugsBunny*–.

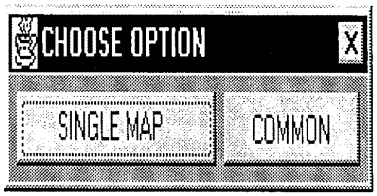


Figura 3 Selección del modo de

Simulación

Llegados a este punto la competición ya está lista para ser realizada. En la figura siguiente se presenta el aspecto de la competición.

La figura 4 ilustra el mapa selecionado con las tres hormigas participantes, así como los gráficos de rendimiento alimenticio de cada hormiga. En la parte central de la figura se muestran los controles de la simulación.

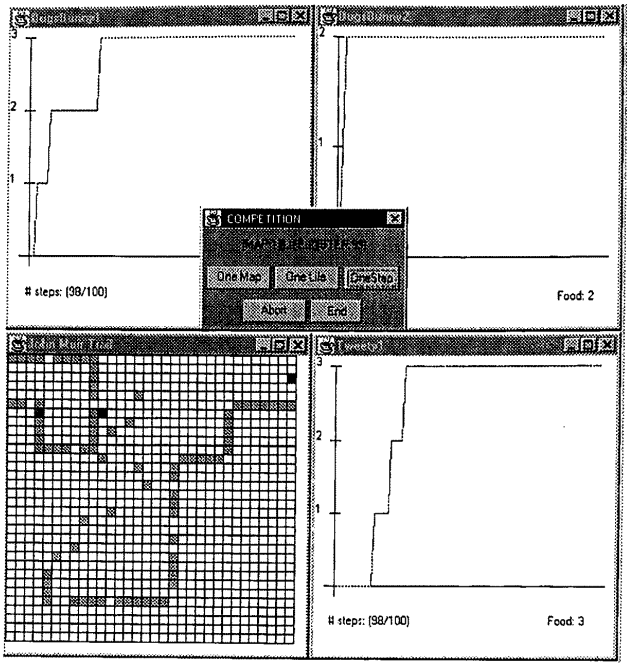


Figura 4 Aspecto de la competición

4.- RESULTADOS DE LA EXPERIENCIA

Este esquema de competición se implantó por primera vez en el curso 1998-1999. Éste es, por tanto, su segundo curso. Los grupos que participaron en la competición del curso 1998-1999 fueron cinco, los cuales decidieron usar las siguientes metodologías:

- Programación Genética (dos grupos)
- Sistemas Clasificadores, basados en SCS de Goldberg (dos grupos)
- Razonamiento Basado en Casos (un grupo)

La experiencia resultó muy positiva. Algunas de las conclusiones de los alumnos fueron:

- se llevaron una grata sorpresa al ver que los métodos estudiados proporcionaban un comportamiento "inteligente" a las hormigas.
- profundizaron en una metodología concreta, descubriendo las peculiaridades inherentes a la misma.
- se dieron cuenta que el trabajo desarrollado no era más que un punto de partida, desde el cual quedan muchas líneas abiertas por investigar.

5.- CONCLUSIONES Y LÍNEAS DE FUTURO

En este artículo se ha presentado el diseño de la parte práctica de una asignatura cuyo objetivo es introducir al alumno en el aprendizaje automático. La implementación de un sistema inteligente por parte de los alumnos permite una mayor consolidación de los conocimientos transmitidos en los seminarios. Así mismo, su estructuración en un esquema de competición estimula el interés del alumno hacia la inteligencia artificial y en particular, hacia las actividades de nuestro grupo de investigación en sistemas inteligentes.

Esta experiencia docente fue realizada por primera vez el curso 1998-1999 y durante este segundo curso estamos consolidando la plataforma de competición. Por tanto, esperamos en un futuro próximo poder divulgar la plataforma en una página Web, abriendo así la competición a toda persona interesada.

6.- AGRADECIMIENTOS

Nos gustaría agradecer a Enginyeria i Arquitectura La Salle el soporte que ofrece a la actividad del grupo de Sistemas Inteligentes. Así mismo, muchas gracias a Mireia Nadal y Vicenç Chamorro por su colaboración en el desarrollo y consolidación de la plataforma Web.

7.- BIBLIOGRAFÍA

[1] Ken Arnold and James Gosling, *The Java Programming Language*, Addison Wesley, 1998.
[2] Gary Cornell and Cay S. Horstmann, *Core Java Vol I Fundamentals*, Prentice-Hall, 1999.
[3] David E. Goldberg, *Genetic Algorithms in search, optimazation and machine learning*, Addison-Wesley Publishing Company, Inc., 1989.
[4] Steven Levy, *Artificial life: a report from the frontier where computers meet biology*, Vintage Books, 1992.
[5] Mitchell A. Potter, Kenneth A, DeJong and John J. Grefenstette, *A Coevolutionary approach to Learning Sequential Decision Rules*, Proceedings of the sixth International Conference (ICGA95), L.J. Eshelman (eds.), San Francisco, CA, Morgan Kaufmann.
[6] Stewart W. Wilson, *Generalization in the XCS Classifier System*, Genetic Programming 1998: Proceedings of the third annual conference, J. Koza et al. (eds.), San Francisco, CA, Morgan Kaufmann.

DESARROLLO DE UNA HERRAMIENTA DE AYUDA A LA ENSEÑANZA DE HEURÍSTICAS

J. Brito Santana, Francisco J. Martínez, José Moreno Pérez, J. Marcos Moreno Vega

Centro Superior de Informática
Universidad de La Laguna
38271 La Laguna, Santa Cruz de Tenerife
e-mail: (jbrito, fmartinz, jamoreno, jmmoreno) @ull.es

RESUMEN: En el presente trabajo se presenta una aplicación desarrollada en entornos gráficos interactivos que permite simular, paso a paso, el desarrollo de heurísticas. Esta herramienta facilita la docencia al profesor de la asignatura optativa Heurísticas que se imparte en la Ingeniería Técnica en Informática de Gestión de la Universidad de La Laguna y permite al alumno comprender y asimilar conceptos, y mejorar el conocimiento práctico de estas técnicas.

1.INTRODUCCIÓN.

Desde las primeras experiencias docentes con la asignatura *Heurística*, se destacaron algunas dificultades de aprendizaje en relación con ciertos conceptos difíciles de comprender y adquirir por medio de métodos tradicionales, no interactivos ni multimedios. Estas dificultades no son específicas de la asignatura sino en general son dificultades comunes a la enseñanza de las materias del área de informática. Entre las mayores dificultades específicas encontradas se hallan las siguientes:

a) la imposibilidad de comprender correctamente el funcionamiento de algunas técnicas heurísticas, ya que éstas se basan en sucesos aleatorios o dependen de los valores que toman ciertos parámetros. Son comunes afirmaciones del tipo *no puedo seguir una traza de este procedimiento, ¿porque cada vez que lo ejecuto obtengo resultados diferentes?,* o cuestiones como *¿cuál es la influencia que este parámetro tiene en el comportamiento de la heurística?*

b) la tendencia de muchas implementaciones a funcionar como cajas negras, sin posibilidad de conocer, en modo paso a paso, el funcionamiento de las técnicas consideradas.

c) el escaso tiempo que se puede dedicar a un alumno en clase teórica o práctica. El grado de asimilación de conceptos varía de un alumno a otro, pero es muy difícil realizar una enseñanza personalizada con el número moderado de alumnos que cursan la asignatura.

Por ello, nos planteamos como objetivo diseñar e implementar una herramienta de apoyo a la enseñanza de técnicas heurísticas que facilite la asimilación de conceptos difíciles en un contexto práctico y en un entorno gráfico.

El resultado es una herramienta gráfica que facilita la exposición de las ideas y procedimientos en clases teóricas y prácticas, una herramienta interactiva que permite diálogos entre la aplicación y el alumno o profesor, capaz de representar diferentes modelos y opciones a petición de los usuarios, visualizar representaciones gráficas de las decisiones tomadas, seguir la traza de los procedimientos implementados y reproducir experimentos continuamente.

La herramienta desarrollada bajo entornos WINDOWS, a modo de tutorial o laboratorio virtual, es un programa eminentemente didáctico, que permite simular el comportamiento de diferentes heurísticas de tal manera que podemos desarrollar y controlar los comportamientos aplicados a problemas concretos, en este caso el TSP.

2.- CONTENIDOS TEÓRICOS Y PRÁCTICOS DE LA ASIGNATURA.

En la Universidad de La Laguna, las técnicas heurísticas se imparten en algunas asignaturas genéricas de Inteligencia Artificial y, fundamentalmente, dentro de la asignatura optativa denominada Heurísticas del plan de estudios de la Ingeniería Técnica en Informática de Gestión del Centro Superior de Informática.

Se trata de una asignatura optativa que tiene asignada una carga docente de seis créditos: tres teóricos y tres prácticos. Los descriptores de la asignatura son: Recristalización Simulada, Búsqueda Tabú, Algoritmos Genéticos, Métodos Multiarranque, Relajación Lagrangiana y Evaluación de Heurísticas. El programa que se imparte está estructurado de la siguiente manera:

Capítulo 1: *Introducción*
- Tema 1.1: Problemas: formulación, ejemplos y representación de soluciones
- Tema 1.2: Heurísticas

Capítulo 2: *Métodos de búsqueda por entornos*
- Tema 2.1: Búsquedas por entornos
- Tema 2.2: GRASP
- Tema 2.3: Métodos multiarranque
- Tema 2.4: Recristalización simulada
- Tema 2.5:Búsqueda por entornos variables

Capítulo 3: *Uso de memoria.* Olvidando y recordando cosas
- Tema 3.1: Búsqueda tabú

Capítulo 4: *La Naturaleza en nuestra ayuda*
- Tema 4.1: Algoritmos Genéticos
- Tema 4.2: Colonias de hormigas

Capítulo 5: *Evaluación de Heurísticas*
- Tema 5.1: Métodos estadísticos
- Tema 5.2: Cotas inferiores. Relajación Lagrangiana.

Se trata de introducir al alumno en las nociones de heurísticas y el papel que juegan en Inteligencia Artificial y en Investigación Operativa y se consideran los principales campos de aplicación de las técnicas. Se estudian los procedimientos heurísticos más importantes, que van desde los basados en búsqueda por entornos hasta los procedimientos es que imitan el comportamiento contenido en la naturaleza. Los últimos temas está dedicado a la evaluación de heurísticas.

3.- DESCRIPCIÓN DE LA HERRAMIENTA.

La aplicación que se desarrollado la hemos denominado VTSPS (Visual Travelling Salesman Problem Solver). Se trata de una aplicación software implementada usando Borland C++ Builder y que se ejecuta bajo Windows. Entre otras características, VTSPS muestra, durante la ejecución de una heurística particular, qué está sucediendo, qué decisiones se han tomado y en qué premisas se han basado. Todo esto está cuidadosamente integrado en un entorno que ofrece la suficiente flexibilidad como para cambiar los datos del problema en cualquier momento y detectar, así, los puntos débiles de las heurísticas en estudio.

Las técnicas heurísticas incluidas en VTSPS son las enumeradas en la sección anterior, es decir, las que se estudian en la asignatura Heurísticas de la Universidad de La Laguna.

El problema considerado como base para las diferentes implementaciones es el Problema del Viajante de Comercio. La elección se realizó por varios motivos: por ser un problema fácil de definir y difícil de resolver, por estar ampliamente documentado, por ser el problema que más se emplea para evaluar el rendimiento de las heurísticas, por la gran cantidad de representaciones distintas que se han propuesto para sus soluciones y por pertenecer a una amplia clase de problemas importantes en los que se debe determinar la permutación óptima de elementos que optimice una cierta función de coste.

Al ejecutar VTSPS, lo primero que observa el usuario es que la pantalla queda dividida en tres espacios claramente diferenciados. En el primero de ellos se encuentran *menús desplegables*, el segundo consta de una *barra de botones* y el tercero, que actúa como *área de trabajo*, se reserva como espacio para mostrar resultados o recibir datos por teclado.

En los menús desplegables de la página principal se encuentran las diversas opciones del programa agrupadas en diferentes categorías que son las siguientes. *Fichero:* incluye operaciones clásicas sobre ficheros como cargar datos de un problema desde fichero, almacenar los datos de un problema o importar y exportar a varios formatos; *Solución inicial*: se enumeran diferentes posibilidades que existen para construir soluciones de partida desde las que realizar la búsqueda; *Operaciones:* recoge los movimientos que pueden aplicarse a la solución actual; *Metaheurística*: permite activar las diferentes estrategias heurísticas: búsquedas aleatorias, búsquedas locales, multistart, recocido simulado, algoritmos genéticos, búsqueda tabú, búsqueda por entornos variables y colonias de hormigas; *Notación*: ofrece las diversas posibilidades asociadas al cambio de representación de la solución; *Cruces:* permite efectuar, paso a paso, varios de los operadores de cruce para el Problema del Viajante de Comercio dado que el cruce es uno de los operadores de implementación más compleja en un Algoritmo Genético; *Ver*: recoge diferentes opciones que tratan con la visualización mediante una representación gráfica de las soluciones, del problema o de las técnicas heurísticas; *Opciones*: muestra opciones genéricas como selección de directorios de trabajo o modo paso a paso en la ejecución.

Un menú especialmente importante es el denominado: *metaheurística.* En el se incluyen las diversas técnicas heurísticas que, por medio de nuevos menús desplegables, pueden ser apropiadamente parametrizadas. Uno de estos menús, el correspondiente al procedimiento heurístico basado en colonias de hormigas se, encuentra en la figura 1.

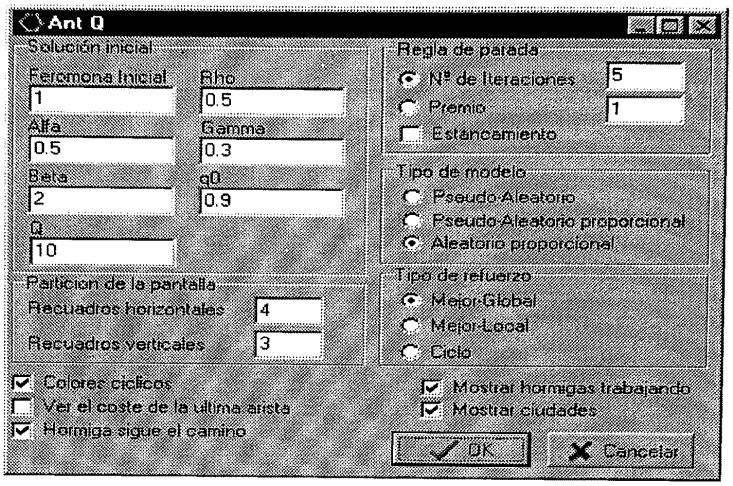


Figura 1

La barra de botones y las funciones de cada botón se recogen en las figura 2 y 3. En ella se incluyen las operaciones que con mayor frecuencia se realizan.

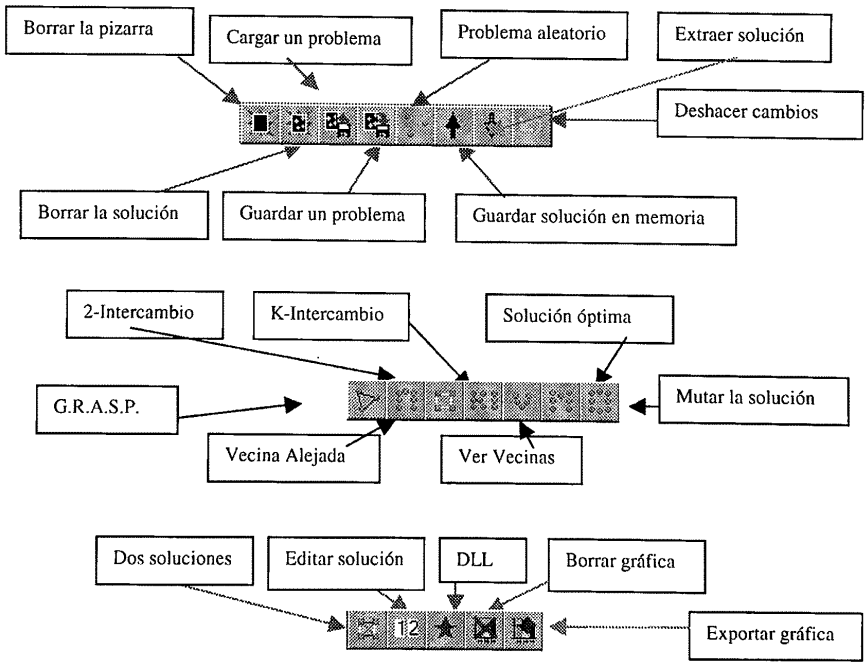


Figura 3

El área de trabajo posee varias solapas: *grafo*, *ciudades*, *distancias*, *histórico*, *datos* y *comparativas*. Cuando se activa la solapa grafo, la pantalla funciona como una pizarra en la que es posible escribir un problema del TSP. Posee para ello opciones de escritura, borrado, desplazamiento, zoom, rotación y bloqueo. El resto de las solapas contienen la siguiente información: *ciudades*, muestra una lista con las coordenadas de los nodos del problema; *distancias*, muestra la matriz de distancia entre nodos del problema actual; *histórico*, refleja nuestras operaciones y el resultado de las mismas; *datos*, guarda los resultados de mejora de todas las heurísticas ejecutadas hasta el momento; *comparativas*, muestra en una única gráfica los resultados almacenados en la solapa datos. En la figura 4, se muestra la apariencia del área de trabajo durante la ejecución, paso a paso, del procedimiento heurístico basado en colonias de hormigas. Asociada a la última arista del recorrido se encuentra el valor en que se ha basado tal elección.

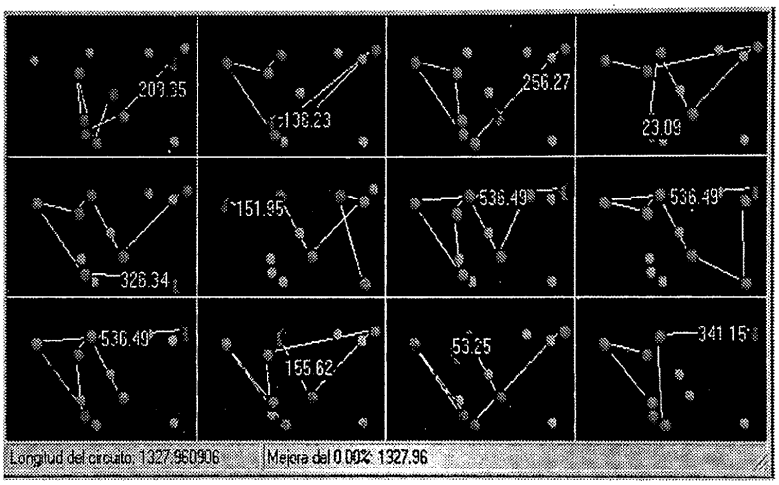


Figura 4

4.- METODOLOGÍA.

La metodología que se utiliza para que el alumno alcance las capacidades propuestas en la asignatura, combina las exposiciones teóricas del docente, con utilización de medios gráficos, en las que el alumno va conociendo los distintos procedimientos heurísticos, con ejemplos de implementación de algoritmos concretos y un conjunto de aplicaciones prácticas.

Con la herramienta que hemos desarrollado el profesor puede ir explicado cada una de las ideas y visualizando con su ayuda, la traza de los algoritmos implementados.

Los alumnos participan aportando sus ideas para mejora los procedimientos propuesto y la búsqueda de soluciones .

En las prácticas semanales se abordan por parte de los alumnos la implementación práctica de los procedimientos descritos en las clases teórica. Además, para la comprensión integrada de las diferentes heurísticas y su aplicación a problemas concretos de la industria y la empresa, los alumnos en grupo trabajan sobre ellos recopilando información, analizándola, proponiendo procedimientos heurísticos de solución, obteniendo soluciones aplicando diversos

procedimientos y estrategias, analizando los resultados obtenidos y exponen el trabajo realizado. Hay que destacar que con esta herramienta el alumno puede experimentar por sí mismo y el profesor tutorizar y resolver problemas puntuales.

A los alumnos se les facilita la herramienta al mismo tiempo que se utiliza para las exposiciones en clases teóricas por parte del profesor, explicándoles las diferentes posibilidades para su utilización. El alumno dispone de una herramienta para su uso particular, con lo que puede ir estudiando y siguiendo lo que se explica en las clases como refuerzo.

Por tanto metodológicamente VTSPS se puede usar de dos maneras: como ayuda para impartir las clases teóricas y prácticas (con él se puede, por ejemplo, construir problemas con características especiales que sirvan para analizar el comportamiento de una heurística) y cómo herramienta de prácticas autoformativas por parte del alumno. Durante el curso 1999-2000, se le ha dado a VTSPS estos dos usos.

5.- CONCLUSIONES.

A partir de la utilización de la aplicación como herramienta docente y de ayuda en las prácticas de los alumnos, se ha constatado que ayuda gráficamente a presentar los problemas y los métodos, y a comparar las soluciones. Valoramos como muy positivo la mejora el aprendizaje de conceptos abstractos ya que podemos conceptualizarlos de manera más natural a base de simular su comportamiento. Los alumnos son capaces de entender y asimilar mejor el funcionamiento de los procedimientos heurísticos. También hemos observado que los alumnos han incrementado la retención de conocimientos, por repetición, por el papel activo del alumno y por la experimentación o descubrimiento activo. El alumno en las prácticas puede experimentar por sí mismo, avanzar más rápido y el profesor dedicar más tiempo a tutorizar y resolver problemas puntuales.

Ha supuesto un refuerzo para la motivación de los alumnos, con enseñanza más personalizada directa, más tiempo para experimentar y resolver dudas presenciales.

REFERENCIAS

[1] **Colin R. Reeves.** *Modern Heuristic Techniques for Combinatorial Problems.* Blackwell Scientific Publications, 1993

[2] **F. J. Martínez.** *VTSPS.* Proyecto de Licenciatura de Ingeniero en Informática. 1999. Universidad de La Laguna.

[3] **Z. Michalewicz.** *Genetic Algorithms + Data Structures = Evolution Programs.* Ed.. Springer 1994

[4] **N. Mladenovic and P. Hansen.** *Variable Neighborhood Search*: Principles and Applications. Comuters Ops. Res. Vol.24. No. 11 1997

ÁREA DE INFORMÁTICA INDUSTRIAL

Responsable: Dr. Iñaki Iñigo Ochoa de Chinchetru
(Universidad de Zaragoza)
E-mail: iiochoa@posta.unizar.es

LA TECNOLOGIA DE COMPUTADORES EN LAS INGENIERIAS INFORMATICAS

Josep Fernández Ruzafa, Antoni Grau

Dept. de Ingeniería de Sistemas, Automática e Informática Industrial, ESAII
Facultad de Informática de Barcelona FIB
Universidad Politécnica de Catalunya UPC
e-mail: {ruzafa, agrau}@esaii.upc.es

RESUMEN: En el presente artículo se presenta la docencia que imparte el Departamento de ESAII en la Facultad de Informática de Barcelona en el campo de la Tecnología de Computadores. Se plantean las deficiencias que existen y se proponen soluciones para que todos los alumnos de la FIB tengan unos mínimos conocimientos en Tecnología de los Computadores a partir de la inmediata revisión de los Planes de Estudios.

1. INTRODUCCIÓN.

Desde el año 1960, en que se inicio el uso comercial de los computadores, estamos inmersos en una verdadera revolución digital, que especialmente en los últimos 10 años, esta transformado nuestra sociedad y cambiando nuestra forma de vivir. Un claro indicador de este fenómeno es que muchos de los sistemas que antes eran analógicos, ahora son digitales, por ejemplo:

- El almacenamiento y reproducción de sonido (los CD).
- La telefonía digital (tanto las centrales digitales de teléfonos, como la telefonía móvil GSM)
- El almacenamiento y reproducción de imágenes -cámaras fotográficas digitales- y de secuencias de imágenes (vídeos) -cámaras de vídeo DVD)
- Sistemas automáticos de control y gestión de diferentes elementos de un coche (carburación, sensores, ..)

En todos estos sistemas, su "cerebro" es un circuito basado en dispositivo de lógica digital (circuitos integrados de tipo "dispositivos de lógica programable" un microcontrolador o un sistema basado en un microprocesador y aquellas empresas que no sean capaces de modernizar sus productos están condenadas al fracaso.

En este contexto socioeconómico, es obligación de las Universidades formar técnicos que permitan a nuestro sector industrial seguir la evolución tecnológica. No hemos de ser un país que tan sólo "consume" tecnología, sino ser un país que también "genera" tecnología.

De aquí la necesidad de formar ingenieros capaces de diseñar e implementar sistemas digitales (*hardware*) a medida para dar solución a una aplicación o problema concreto, los llamados sistemas empotrados (*embeded systems*). La tendencia actual, dadas las características de las demandas, es la utilización de sistemas conectables en red (Bus USB, *Can*, *Ethernet*, ...) para

incrementar la modularidad del sistema, facilitar su acceso desde un computador remoto y facilitar la expansión del sistema. De esta forma, cada vez más la complejidad y las prestaciones del sistema están pasado del *hardware* al *software*. De aquí el interés que sean ingenieros en informática, con una amplia base en la tecnología de la programación, los encargados de desarrollar estos sistemas en vez de ingenieros en electrónica, que a pesar de tener una amplia base en electrónica digital, muy a menudo presentan deficiencias en el desarrollo de sistemas basados en microprocesadores y en la tecnología de la programación.

Para que un ingeniero informático pueda desarrollar estas competencias, es necesario ampliar la formación en tecnología de la programación, ingeniería del *software* y arquitectura de computadores con una formación en tecnología de computadores: diseño, implementación de sistemas digitales (en el sentido más amplio: electrónica digital, PLD, FPGA, microcontroladores, sistemas basados en microprocesadores, computadores industriales), y su configuración y/o programación.

1.1. Objetivos de una formación en tecnología de computadores

El objetivo general de una formación de ETIS es dar al alumno una competencia en la construcción, implantación, utilización y mantenimiento de un sistema informático. Dentro del área de TC, el alumno ha de ser capaz de trasladar las mismas destrezas y aptitudes en el ámbito de los sistemas digitales. Este objetivo genérico lo podemos concretar en los siguientes objetivos básicos:

- Comprender los conceptos de la electrónica en general y de la electrónica digital y la microelectrónica en particular.
- Reconocer el proceso de fabricación de circuitos integrados.
- Manejar los instrumentos básicos de un laboratorio de electrónica.
- Usar las herramientas de CAD para el diseño de circuitos electrónicos.
- Diseñar sistemas basados en lógica digital y decidir que tipo de dispositivos utilizar para su implementación.
- Analizar el comportamiento y funcionamiento de un sistema digital.
- Comprender y reconocer los conceptos de periféricos y de interfaz.
- Decidir e integrar que elementos deben formar parte de un sistema que da solución a una aplicación concreta.
- Confeccionar una documentación técnica de un sistema.
- Estar habituado a trabajar en equipo y tener capacidad de comunicación con los clientes y los miembros del equipo.

Una formación de ciclo largo tendrá unos objetivos más amplios y más variados. Dentro de la EI existen varios perfiles o competencias profesionales. Entre ellas podemos identificar claramente la tecnología de computadoras o más ampliamente la informática industrial. El objetivo general de una formación en TC dentro de la EI es dar al alumno una competencia en el diseño y evaluación de sistemas digitales, así como en su construcción, implantación, utilización y mantenimiento. Como se puede ver, únicamente las dos primeras destrezas y conocimientos separan la formación del ETIS de la del EI. Este objetivo genérico lo podemos concretar en los objetivos básicos de la ETIS, más lo siguientes objetivos específicos de la EI:

- Identificar los problemas inherentes de la utilización de un sistema digital en un entorno industrial.

- Desarrollar una especificación del sistema a partir de la información indicada por el cliente y la propia búsqueda de información.
- Diseñar e implementar, tanto el hardware como el software, sistemas basados en microcontroladores o microprocesadores y decidir de que elementos debe estar formado.
- Usar las herramientas de CAD/CAE para el diseño de sistemas digitales: lenguajes de descripción de hardware, entornos de desarrollo para PLD, FPGA, microcontroladores y microprocesadores.
- Juzgar que funciones del sistema han de ser realizadas en hardware o en software.
- Evaluar las diferentes alternativas para la implementación de un sistema, y decidir cual es la más adecuada según los criterios definidos.
- Comprender el funcionamiento y la tecnología asociada a los periféricos multimedia.
- Evaluar y contrastar el comportamiento y funcionamiento de un sistema digital.
- Organizar y dirigir un equipo de trabajo.

Estos objetivos serán conseguido mediante el trabajo realizado a lo largo de las diferentes asignaturas que forman el perfil TC. Por sus características algunos de estos objetivos serán trabajados más en clases de teoría y problemas, y otros en clases de laboratorio. En los dos próximos capítulos se repasaran que contenidos consideran los autores, que se han de incluir en teórica y en práctica dentro de la formación en TC.

2. LOS CONOCIMIENTOS DENTRO DE LA FORMACIÓN EN TECNOLOGIA DE COMPUTADORES.

Los conocimientos necesarios para alcanzar los objetivos que se han fijado en la ETIS y en la EI están recogidos en la tabla 1.

Comunes a las ETIS y EI	Específicos a la ETIS	Específicos a la EI
Electrónica básica	Test y mantenimiento de sistemas digitales	Lenguajes de Descripción de hardware
Electrónica digital		Diseño de sistemas basados en microcontroladores y microprocesadores
Microelectrónica básica		Sistemas multimedia
Diseño e implementación de circuitos lógicos.		Computadores, periféricos y buses industriales
Análisis de sistemas digitales		
Periféricos y interfaces estándar		

Tabla 1. *Conocimientos en las ETIS y EI*

3. DOCENCIA EN LA FACULTAD DE INFORMÁTICA DE BARCELONA (FIB)

En la FIB, la docencia relativa a la Tecnología de Computadores siempre ha correspondido al departamento de ESAII (Ingeniería de Sistemas, Automática e Informática Industrial), el cual

estuvo presente (bajo el nombre de TC, Tecnología de Computadores) en la fundación de la FIB.

Ya en el inicio de los estudios de la carrera de informática, se vio de forma imprescindible la impartición de materias con contenidos de tecnología, aspecto que da una visión más real a los alumnos sobre cómo se han diseñado y fabricado muchas de las máquinas y dispositivos electrónicos que se usan de forma habitual y cotidiana.
Después de varios cambios en los Planes de Estudios, la configuración que existe hoy en día referente a asignaturas de Tecnología de Computadores en la FIB es el que se muestra en la figura 1.

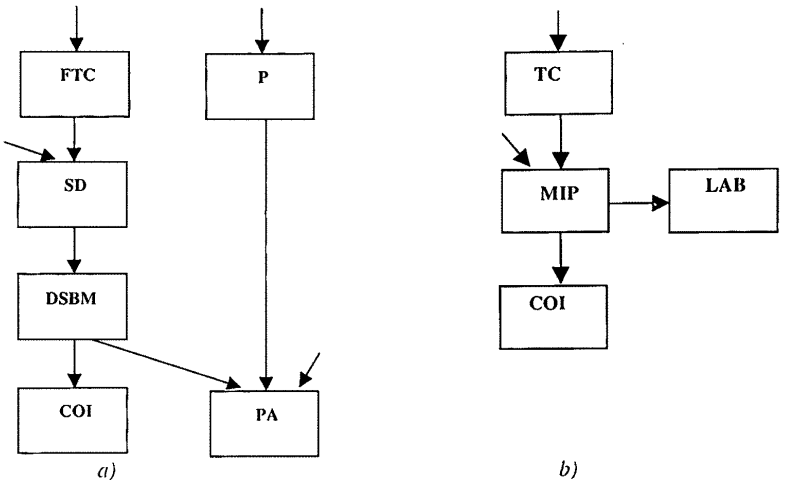


Figura 1. *Asignaturas de Tecnología de Computadores en la FIB: a) actualidad; b) propuesta del Departamento de ESAII.*

El bloque de asignaturas del departamento de ESAII, correspondientes a Tecnología de Computadores, puede ser cursado a partir del 3er cuatrimestre con el único requisito de haber aprobado la fase de Selección (y con ella Física) y Estructura de Computadores I (EC-I, solo para Periféricos P, PA significa Periféricos Avanzados).

A continuación, detallamos cada asignatura para comprender mejor cuales son los objetivos que se persiguen y su metodología.

Asignatura FTC. Fundamentos Tecnológicos de los Computadores. Asignatura de 6 créditos divididos en 3 cr. de teoría, 1.5 cr. de problemas y 1.5 cr. de laboratorio. El único requisito es Física. Los objetivos docentes son básicamente la adquisición de un nivel básico de conocimientos en el campo de la ingeniería de sistemas informáticos por lo que respecta a *hardware*. A tal efecto se estudian las tecnologías de fabricación de los principales dispositivos de estado sólido, se analizan familias lógicas de uso común para alcanzar una capacidad de análisis de los elementos y sistemas electrónicos de incidencia en el campo de la informática. El temario a grandes rasgos es:

1) evolución tecnológica de los ordenadores;
2) estado sólido: semiconductores, unión PN, diodos;
3) transistores bipolares: efecto transistor, polarización, amplificación, conmutación;

4) transistores de efecto campo: transistores JFET, MOS y circuitos;
5) circuitos integrados digitales: integración, tecnología de fabricación, principales familias, alta escala de integración, situación actual y perspectivas.

Las clases teóricas se complementan con prácticas de laboratorio con el fin de completar los objetivos de carácter experimental: adiestramiento en la utilización del instrumental de medida habitual asociado a los equipos informáticos así como el conocimiento y familiarización con la tecnología electrónica.

Cada cuatrimestre hay unos 90 alumnos de los cuales 60 son alumnos que cursan la asignatura de forma obligatoria (técnicos de sistemas).

Asignatura SD. Sistemas Digitales. Asignatura de 7.5 créditos. El requisito básico es FTC. Los objetivos docentes son básicamente la adquisición de una base sólida de utilización de funciones lógicas elementales y complejas y el diseño de circuitos combinacionales y secuenciales. Se introducen las herramientas de descripción y simulación para ayudas al diseño y verificación de sistemas digitales con la máxima eficiencia y fiabilidad.

El temario a grandes rasgos es:

1) sistemas digitales: caract. generales,cableado, programable, aplicaciones;
2) circuitos SSI y MSI: familias actuales, interficies y componentes;
3) implementación de sistemas lógicos: con SSI, MSI, ROM, PLD;
4) circuitos secuenciales básicos: biestables, análisis y diseño de sistemas síncronos y asíncronos;
5) circuitos impulsionales: derivadores, retardadores, secuenciadores;
6) diseño e implementación de sistemas digitales: selección de tecnología y escala, herramientas CAD de ayuda al diseño. Normas y estándares.

Las clases teóricas se complementan con prácticas de laboratorio donde se utilizaran herramientas informáticas para la implementación de *hardware*. En las prácticas se diseña e implementan sistemas digitales con lógica combinacional y un sistema secuencial.

Cada cuatrimestre hay unos 30 alumnos cursando la asignatura de forma optativa.

Asignatura DSBM. Diseño de Sistemas basado en Microprocesadores. Asignatura de 9 créditos. El requisito básico es SD. Los objetivos docentes son la presentación de los elementos que forman un sistema microprocesador, su conexionado al bus y la utilización como solución a aplicaciones. También se da una visión descriptiva de los sistema microprocesadores avanzados y sistemas multiprocesadores.

El temario a grandes rasgos es:

1) sistemas digitales programables: microprocesador y microcomputador;
2) memoria, E/S: puertos programables, interrupciones, interficies;
3) transferencia de datos: serie, paralelo, estándares, DMA, relojes;
4) microcomputadores integrados: arquitecturas, E/S, aplicaciones;
5) procesadores especializados y sistemas multimicroprocesador.

Las clases teóricas se complementan con prácticas de laboratorio en las cuales se implementará una aplicación que usa un microcomputador.

Esta asignatura se imparte anualmente con unos 30 alumnos cursando la asignatura de forma optativa.

Asignatura COI. Computadores Industriales. Asignatura de 9 créditos. Los objetivos docentes es el estudio de las características que se requieren en un ordenador para ser implantado en un entorno industrial. Se aprenden a diseñar tarjetas con interficies para compatibilizar buses y adaptarse a las necesidades industriales.

El temario a grandes rasgos es:

1) requerimientos de la industria: tiempo real, autómatas, concepto de bus;
2) entorno industrial: ruido, acoplamiento, inmunidad, aislamiento, ejemplos de diseño;
3) interficies: adquisición de datos,control de potencia, E/S, monitorización
4) configuración del ordenador industrial: especificaciones de un diseño, sistemas operativos en tiempo real, selección del bus, memoria.

Esta asignatura tiene un alto contenido de sesiones de laboratorio para implementar un sistema microprocesador industrial completo. Esta asignatura se imparte anualmente con unos 20 alumnos cursando la asignatura de forma optativa.

Ninguna de las asignaturas de Tecnología de Computadores, las cuales configuran el perfil de Informática Industrial, son obligatorias en la titulación de Ingeniería en Informática, hecho más que sorprendente. Solamente hay dos asignaturas (FTC y P) que son obligatorias en la titulación de Ingeniería Técnica en Informática de Sistemas, no siéndolo entre otras ni Sistemas Digitales ni DSBM, aspecto que vuelve a sorprender.

4. PROPUESTA DE MEJORA EN LOS CONOCIMIENTOS A IMPARTIR

Después de la presentación del estado actual y sus deficiencias, creemos que los alumnos han de tener un mayor grado de conocimiento de Tecnología de Computadores mediante una reforma del Plan de Estudios. En la figura 1.b) se propone un nuevo esquema fundiendo las asignaturas actuales y actualizando los temarios. La asignatura TC (Tecnología de Computadores) es la fusión de FTC y SD, mientras que MIP (Microprocesadores y Periféricos) es la mezcla de DSBM y P. Se crea una asignatura específica de laboratorio [JEN98].

La FIB reconoce un perfil (competencia profesional) en Informática Industrial [GUI99, pág. 21] pero no se expide ningún documento acreditativo ni certificado. Desde el departamento de ESAII venimos pidiendo a la FIB y a su decano como máximo representante, que se reconozca que la Tecnología de Computadores es una área de merecido aprendizaje por parte de cualquier ingeniero superior informático. Este reconocimiento lo exigimos de 2 formas:

- paso de optatividad a obligatoriedad en la ingeniería superior de las asignaturas de nueva creación TC (Tecnología de Computadores) y MIP (microprocesadores y Periféricos).

- reconocimiento explícito (con acreditación) del perfil de Ingeniero en Informática Industrial. Este perfil está siendo reclamado también desde amplios sectores de la industria tecnológica en Catalunya.

En estos momentos se está llevando a cabo una reforma de los Planes de Estudio en la FIB y creemos que es el mejor momento (y el marco indicado) para "arreglar este error histórico" y crear el perfil en II con asignaturas obligatorias para los titulados superiores.

REFERENCIAS

[GUI99] Guía Docente de la Facultad de Informática de Barcelona, Universidad Politécnica de Catalunya, curso 99-00.

[JEN98] Casals, A., Martínez, A.B. y Grau A., "La Formación en Informática Industrial", JENUI'98, Andorra, Julio, 1998, p. 230-235.

LA ASIGNATURA ADQUISICIÓN Y ACONDICIONAMIENTO DE LA SEÑAL EN LA UNIVERSIDAD DE ALICANTE.

Francisco A. Pujol[1], F. Javier Ferrández[1], Andrés Fuster[1], Juan Manuel García Chamizo[1]

[1]*Departamento de Tecnología Informática y Computación*
Universidad de Alicante
Apartado de Correos 99 (03080) Alicante
e-mail: [fpujol, fjferrran, fuster, juanma]@dtic.ua.es

RESUMEN: En el presente artículo se presenta una propuesta para la asignatura optativa de la titulación de Ingeniería Informática *Adquisición y Acondicionamiento de la Señal*. Se describen los objetivos, contenidos y evaluación de esta asignatura y, dado que se trata de una asignatura de marcado carácter práctico, se incide en la propuesta de trabajos que deben ser llevados a cabo por los alumnos.

1.- INTRODUCCIÓN.

En el desarrollo de una aplicación industrial una de las principales características que se deben considerar es el hecho de que se trata de una aplicación distribuida, entendiendo como tal aquella en la que los datos y los métodos que actúan sobre ellos se encuentran dispersos en un área geográfica determinada [2],[5]. En la mayoría de los casos de aplicaciones industriales distribuidas, la información es generada por sensores, las acciones necesarias son ejecutadas por actuadores, y el procesamiento de la información es llevado a cabo por algoritmos ejecutados mediante microcontroladores o por computadores personales [3],[4].

Un objetivo primordial de la tarea docente en las asignaturas relacionadas con la Informática Industrial es asegurar que los titulados estén preparados para afrontar retos competitivos dentro del sector industrial [1]. Para ello, los alumnos deben tener el conocimiento de la situación tecnológica actual que les permita integrarse sin dificultad en el mundo industrial, y además deben tener la capacidad para adaptarse a los nuevos cambios que se producen en el sector de la Informática Industrial más afín al ingeniero informático.

En este artículo describiremos la propuesta que hacemos para la asignatura *Adquisición y Acondicionamiento de la Señal* (AAS), impartida por el Departamento de Tecnología Informática y Computación de la Universidad de Alicante. En la sección 2 presentamos los objetivos de la asignatura, pasando a continuación a abordar sus contenidos y la evaluación de la misma en la sección 3. Dada la importancia que el trabajo práctico posee a la hora de la evaluación, en la sección 4 describimos en profundidad la propuesta de prácticas de laboratorio que hemos efectuado. Finalmente, en la sección 5 extraemos una serie de conclusiones al trabajo presentado.

2.- OBJETIVOS.

De acuerdo con los Planes de Estudios de Informática de la Universidad de Alicante [6], esta asignatura se oferta como optativa tanto para la titulación de Ingeniero Informático, como para las titulaciones técnicas de Ingeniero Técnico en Informática de Gestión y en Informática de Sistemas. Su carga docente es de 4,5 créditos (3 de teoría y 1,5 de tipo práctico) y su duración es cuatrimestral.

Considerando en primer lugar los objetivos que se plantean en esta asignatura, debemos señalar que lo que se pretende principalmente es que los alumnos que la cursan adquieran una base tecnológica sobre la adquisición de la señal que contiene la información del estado de un sistema a controlar y su tratamiento. Para llevar a cabo estos objetivos, se analizan los problemas derivados de la captación y transmisión de la señal y se estudian los circuitos más idóneos de acondicionamiento y protección que se han de utilizar. Finalmente, se examinan los aspectos tecnológicos que se deben considerar en la fase de diseño.

A continuación, analizamos con mayor profundidad estos objetivos:

1. Dar a conocer los sistemas de adquisición de datos y su acondicionamiento para ser utilizados en los sistemas informáticos.

2. Enseñar las tecnologías de percepción de señales y de actuación, así como las arquitecturas de los sistemas de acondicionamiento.

3. Desarrollar las habilidades y aptitudes necesarias para el diseño y realización de sistemas de adquisición y acondicionamiento de los datos.

4. Fomentar el hábito de realización de proyectos completos para que el objetivo último de ver realizado un ingenio completo proporcione confianza al autor.

5. Relacionar los contenidos de la asignatura con los de materias afines (Ingeniería del Mantenimiento, Sistemas Industriales, Diseño de Circuitos, etc.) para afianzar la madurez disciplinaria del estudiante de Ingeniería Informática.

La consecución de estos objetivos se vertebra utilizando unos contenidos y una evaluación adecuados, como comentamos en la siguiente sección.

3.- CONTENIDOS Y EVALUACIÓN.

El programa de esta asignatura recoge la problemática relacionada con la medida de parámetros físicos en general, su adquisición, almacenamiento, pre-proceso y presentación. En líneas generales, estos contenidos se agrupan en los cinco bloques temáticos que ahora se presentan.

a) Generalidades.

Se da una presentación general a la materia abordada en el curso, incluyendo conceptos tales como tipos de señales, clasificación de sensores y actuadores, y una introducción a la especificación y diseño de un sistema de adquisición y acondicionamiento de la señal. Este bloque se organiza en las dos primeras horas de clase, y permite al alumno obtener una visión global del contenido de la asignatura.

b) Percepción y actuación.

Consideramos en este caso los efectos físicos para la realización de sensores, así como las distintas clases de sensores y transductores existentes en el mercado, centrándonos en aplicaciones de los mismos en entornos industriales. En este bloque pretendemos describir también los principios de percepción y actuación que van a guiar un sistema de adquisición y acondicionamiento de la señal. Estos contenidos se desarrollan durante ocho horas.

c) Acondicionamiento de señales.

Es el tema de mayor densidad en cuanto a materia abordada dentro de la asignatura, por lo que se le dedican unas 13 horas lectivas. En este bloque se incide en la necesidad de realizar un procesamiento de la señal adquirida por los sensores y transductores a la hora de que el sistema actúe de manera adecuada cuando se recibe una determinada información de entrada.

Se revisan conceptos de electrónica, incluyendo, entre otros, una descripción del amplificador operacional y sus aplicaciones. Se tratan también cuestiones tales como amplificación de las señales, filtros analógicos, tratamiento analógico de la señal, y finalmente se consideran de manera teórica y práctica la conversión analógica/digital y digital/analógica.

d) Diseño de sistemas de adquisición y acondicionamiento de señales.

En las cuatro horas dedicadas a este bloque se profundiza en cuestiones relacionadas con los problemas de implementación real de un sistema, como son el ruido, espúreos, sistemas de compensación de errores, aislamiento, etc., así como la planificación del diseño de un sistema AAS, sirviendo de base para el trabajo práctico que los alumnos deben llevar a cabo en las prácticas.

e) Sistemas y dispositivos específicos.

En este último bloque temático se presentan sistemas y dispositivos reales: conversores A/D y D/A disponibles en el mercado, sensores y actuadores concretos, ejemplos de sistemas completos (domóticos, aplicados en la industria,...), etc. Para cada uno de éstos, se comentan sus especificaciones técnicas, márgenes de utilización, aplicaciones en las que se utilizan, etc., con lo que se pretende que el alumno tome contacto con los distintos componentes que pueden ser utilizados en una aplicación real. Se dedican tres horas a desarrollar estos contenidos.

f) Evaluación.

Dado el carácter eminentemente práctico de la asignatura, la evaluación de *Adquisición y Acondicionamiento de la Señal* consiste en la valoración de un trabajo práctico realizado a lo largo del curso. El trabajo mínimo que se debe realizar consiste en la especificación y la simulación software de la práctica elegida, mientras que los trabajos que se consideren de calidad suficiente son propuestos para su realización física. En la siguiente sección presentamos algunos de los trabajos prácticos propuestos.

4.- PRÁCTICAS DE LABORATORIO.

Las prácticas consisten en la realización de un proyecto a elegir de entre los que son propuestos al comienzo del curso por parte del profesor. Cada proyecto será realizado por un equipo de tres especialistas: el profesor, que ejercerá como director, y dos estudiantes con funciones de ingenieros informáticos.

El proyecto consta de tres fases, con plazos de realización distribuidos a lo largo del curso. Estas fases son:

- Especificación y diseño
- Simulación
- Realización física

Las dos primeras fases son de carácter obligatorio. La tercera fase puede ser realizada opcionalmente, a propuesta del profesor, en función de la evaluación de la simulación. Pasamos a describir a continuación algunos de los proyectos propuestos para ser implementados por los alumnos.

a) Estación de diagnóstico para computadores personales.

Se trata de desarrollar un sistema para la medición de parámetros (potenciales eléctricos, temperaturas, etc.) en distintos elementos de los computadores personales a fin de determinar el estado de los mismos y poder localizar posibles averías.

La estación se realizará a su vez sobre arquitectura tipo PC y consistirá en un módulo de adquisición de datos (los parámetros del computador objeto de diagnosis), un módulo de acondicionamiento para poder ser suministrados al computador de la estación y el software de diagnóstico. La tecnología a utilizar podrá elegirse entre circuitos integrados e impresos; o bien, entornos de desarrollo (preferiblemente de dominio público) de los ya existentes en el mercado.

b) Sistema perceptual para máquinas-herramienta.

Se trata de desarrollar un sistema para obtener los parámetros de contexto y estado de máquinas herramientas. Se pretende que el sistema sea flexible para que pueda tener características de estandarización; por lo que se plantea, como decisión preliminar de diseño, que el sistema tenga capacidad para adquisición de las magnitudes comunes en cualquier máquina-herramienta (temperatura, posición, distancia, velocidad, etc.) y, así mismo, proporcionar entradas para poder obtener parámetros particulares de algunas máquinas y que son diferentes de unas máquinas a otras (intensidad de luz, peso, etc.). El sistema será, por tanto, parametrizable.

c) Sistema domótico.

Se considera una vivienda unifamiliar aislada con las dependencias al uso: dormitorios, salón, cocina, cuartos de baño, garaje, piscina y jardín.

Se pretende desarrollar el sistema de percepción y actuación para operación, regulación y monitorización del equipamiento de la casa. Es evidente que una propuesta de esta envergadura

puede constituir un proyecto enormemente ambicioso, cuya realización exija un tiempo dilatado. Para resolver este problema, debe tomarse como consideración de diseño previa que la realización del proyecto en todas sus fases se ajuste al plan de tiempo establecido en la realización de las prácticas de la asignatura.

d) Sistema de posicionamiento local.

El uso de robots móviles en ambientes industriales plantea exigencias de localización que difícilmente pueden ser ayudados mediante sistemas como el GPS debido a la existencia de jaulas de Faraday que a menudo constituyen los elementos constructivos metálicos. Se encuentra justificado por esta y otras razones el empleo de sistemas locales de posicionamiento. Se propone, pues, desarrollar un sistema de este tipo para localización en recintos con forma de poliedro recto.

5.- CONCLUSIONES.

Resulta indudable que el uso de la informática se ha extendido vertiginosamente a todos los sectores de nuestra sociedad, pero, sin embargo, son pocos los centros que ofrecen una formación específica en cuanto a la aplicación de la Informática al entorno industrial. Es evidente que los conocimientos y técnicas que ha de dominar un especialista en Informática Industrial no son los mismos que el de los informáticos con especialidades en gestión o sistemas.

En este artículo hemos presentado nuestra propuesta para la asignatura *Adquisición y Acondicionamiento de la Señal*, de tal modo que los alumnos relacionen los conceptos adquiridos con problemas reales aplicados al sector industrial. Consideramos que es imprescindible presentar la asignatura con unos objetivos claros y unos contenidos adecuados a las necesidades de los alumnos, tal y como se ha realizado. Mediante el método de evaluación propuesto los alumnos tienen una primera toma de contacto con el diseño y especificación de un proyecto real, con el consiguiente beneficio tanto a nivel docente como personal. Además, la estructuración de las prácticas en forma de un proyecto en cuyo equipo se incluye al profesor, fomenta una mayor relación profesor-alumno y permite realizar una evaluación continua de la progresión de los alumnos, aspectos que consideramos muy importantes y los cuales redundan en que la motivación de los alumnos y los resultados obtenidos sean excelentes.

6.- REFERENCIAS.

[1] Casals A. et al. *La formación en Informática Industrial.* Actas de las IV Jornadas sobre la Enseñanza Universitaria en Informática (JENUI'98), pp. 230-235. Andorra, Julio 1998.

[2] Considine, D. M. *Process Instruments and Control Handbook.* Ed. McGraw-Hill. 1993.

[3] Pallás Areny, Ramón. *Adquisición y Distribución de Señales.* Ed. Marcombo-Boixareu. Barcelona, 1993.

[4] Pallás Areny, Ramón. *Sensores y Acondicionadores de Señal.* Ed. Marcombo-Boixareu. Barcelona, 1998.

[5] Rico López, Rafael; de Frutos Redondo, José Antonio. *Sistemas de Adquisición y Tratamiento de Datos*. Servicio de Publicaciones. Universidad de Alcalá de Henares, 1996.

[6] Plan de Estudios conducente al título de Ingeniero en Informática de la Escuela Politécnica Superior de la Universidad de Alicante. Resolución de 18 de septiembre de 1992 de la Universidad de Alicante. B.O.E. nº 37, de 12 de febrero de 1993, pág. 4467-4478.

DISEÑO DE LAS PÁGINAS WEB DE LAS ASIGNATURAS DE INFORMÁTICA INDUSTRIAL CON CAPACIDAD PARA REALIZAR ADQUISICIÓN Y ENVÍO DE DATOS A BAJO NIVEL UTILIZANDO LENGUAJE JAVA

H. Hassan, A. Perles, J. Albaladejo, J.M. Martínez, C. Domínguez

Dept. Informática de Sistemas y Computadores (DISCA)
Escuela Universitaria de Ingeniería Técnica Industrial de Valencia
Universidad Politécnica de Valencia
e-mail: {husein,aperles,jalba,jmmr,carlosd}@disca.upv.es

RESUMEN: En la actualidad, el auge de la informática y sobre todo de las comunicaciones a través de Internet nos brinda, en general, la posibilidad de mejorar los servicios de información, y más concretamente en el ámbito docente. Sobre esta temática trata el presente trabajo. El objetivo fundamental es ofrecer un servicio de información y aplicación de la asignatura de Informática Industrial. Este servicio consta tanto de Información general de la asignatura, como de la posibilidad de realización *"on line"* de las prácticas de laboratorio, simulaciones animadas de los trabajos propuestos por la asignatura.

1.- INTRODUCCIÓN.

El objetivo fundamental es ofrecer un servicio de información y aplicación de la asignatura de Informática Industrial. Este servicio consta de:

- Información general de la asignatura, en la que se puede consultar los temarios de que consta, así como se tiene la posibilidad de visualizar todas las transparencias que serán utilizadas por los profesores en el desarrollo de las clases, también se tiene acceso a información sobre los profesores que impartirán la asignatura, (Despacho donde se encuentran, horarios de tutorías y posibilidad de enviarles preguntas mediante correo electrónico), fechas de evaluación, contenidos de las prácticas, etc.,).

- Realización *"on line"* de las prácticas de laboratorio, mediante la utilización de aplicaciones realizadas en lenguaje Java, el alumno puede desde una máquina remota acceder a una tarjeta de adquisición de datos Pcl-812g conectada a la máquina servidora o directamente a sus puertos, con la posibilidad de recibir y enviar señales externas. Esta es la parte más compleja de todo el proyecto, pues supone interactuar con el medio exterior de un sistema de control desde cualquier parte del mundo.

- Simulaciones animadas de los trabajos propuestos por la asignatura. *Applets* de Java que muestran a los alumnos de forma visual el funcionamiento y los requerimientos de los ejercicios de programación que deben realizarse al final del curso.

2.- HERRAMIENTAS DE DESARROLLO.

El lenguaje utilizado para crear las páginas de información que podemos ver cuando navegamos por *World Wide Web* es el lenguaje *HTML (HyperText Markup Languaje)*, un lenguaje de definición de páginas derivado de *SGML (Standard Generic Markup Languaje)*. El lenguaje *SGML* se desarrolló para facilitar la utilización de documentos de distintas plataformas. Con *SGML* se separaba el documento del sistema operativo en el cual se había creado.

2.1. - JAVA.

Java es una tecnología para la construcción de aplicaciones o programas que son ejecutables por múltiples ordenadores a través de la red. Java promete extender el papel de Internet desde el terreno de las comunicaciones hacia una red en la cual puedan ejecutarse las aplicaciones completas. Simplifica también la construcción de programas que se muevan a través de la red y desempeñen funciones en ordenadores remotos en nombre del usuario.

Antes de Java, Internet se utilizaba principalmente para intercambio de información. Las páginas *Web* han permitido a los usuarios acceder a servicios en Internet sin tener que aprender secretos comandos y ha hecho sencillo para los negocios la creación de imágenes corporativas en línea, proporcionando información sobre productos e incluso la venta de mercancía a través de los *PCs*. La tecnología Java llevará estos adelantos todavía más lejos haciendo posible suministrar aplicaciones completamente interactivas vía *Web*. El éxito de Java se debe a que ofrece las siguientes posibilidades:

- Escribir programas fiables y robustos.
- Construir una aplicación para prácticamente cualquier plataforma, y ejecutar dicha aplicación en cualquier otra plataforma soportada sin tener que repetir la compilación del código.
- Distribuir sus aplicaciones a lo largo de la red de una forma segura.

Los programas en Java pueden incrustarse en documentos *Web*, convirtiendo páginas estáticas en aplicaciones que se ejecutan en el ordenador del usuario. Con esto, la documentación en línea no será como un libro impreso sino que podrá incluir simulaciones, modelos de trabajo, e incluso herramientas especializadas.

2.2. – CONDICIONES DE REALIZACIÓN.

Para realizar la parte más complicada se nos plantearon diversos problemas, en primer lugar la elección del lenguaje de programación a utilizar.
Existen diversas formas de programar para lograr que una página *Web* haga algo más que simplemente mostrar información:

- *CGI.(Common Gateway Interface)* Estos programas no van más allá de trasladar una petición realizada en un formulario y devolvernos la información recopilada en el servidor, pero no son verdaderas aplicaciones desarrolladas para ejecutarse en Internet.

- *ActiveX* de Microsoft. No es un nuevo lenguaje de programación, sino que es un conjunto de tecnologías que pueden emplearse en el diseño de controles que posteriormente se utilizan para mejorar el aspecto de las páginas *Web*. *ActiveX*

proporciona un mecanismo para extender cualquier lenguaje de programación permitiéndonos el desarrollo de controles y aplicaciones para Internet.

- *C++*. Es el lenguaje de programación orientado a objetos más utilizado y más popular actualmente, debido a su amplia gama de posibilidades que nos permiten fácilmente acceder a bajo nivel (direcciones de memoria, acceso a puertos, etc.). Esta característica lo hace útil para la realización de nuestro proyecto.

- *Java*. Aunque originalmente Java no se desarrolló como una herramienta exclusiva para Internet, ha sido este el terreno donde se ha aplicado en mayor medida. Java se pensó como un lenguaje de programación orientado a objetos seguro y cuyo código pudiera ejecutarse en diferentes plataformas. Podemos considerarlo como una evolución de C++. Su principal virtud es la seguridad, permite la ejecución de un entorno protegido donde los virus no pueden introducirse, ya que no genera archivos ejecutables sino que genera bytecodes (clases de Java) que se ejecutan en cualquier plataforma que esté equipada con el interprete adecuado. Sin embargo presenta el inconveniente de no incluir en sus prestaciones el acceso a bajo nivel.

2.3. - SOLUCIÓN Y ESQUEMA FINAL.

La arquitectura empleada sobre la que se desarrolla la aplicación es la mostrada en la figura 4-1, donde podemos observar las entidades que intervienen en la aplicación y sus vínculos o relaciones. Se dispone de un ordenador que es la máquina servidora, en el cual se tiene instalada la tarjeta de adquisición de datos. Esta tarjeta recibirá señales de los aparatos generadores de funciones o fuentes de alimentación que se requieran en la realización de las prácticas. A través de Internet se conectarán a la máquina servidora los usuarios que quieran acceder a estos servicios.

En la máquina servidora se integrarán el servidor *Web*, el servidor Java y el navegador de Internet. Las máquinas clientes deberán disponer de navegador.

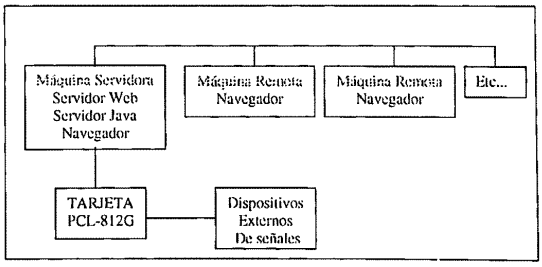


Figura 1. Arquitectura hardware

Definitivamente se ha optado por el empleo de los llamados Métodos Nativos integrados en Java. Se trata de realizar los programas que acceden a bajo nivel en lenguaje C++ para crear una librería dinámica (*archivo.dll*). Esta librería contiene el método nativo (*archivo.c*) y un archivo de cabecera (*archivo.h*), este último sirve de interface entre Java y C++ para invocar el método desde la aplicación (*servlet*) de Java. Este *servlet* es un *applet* de Java que se ejecuta

desde el lado del servidor, de este modo evitamos la restricción de seguridad de los visores Java para invocar código nativo.

Para la gestión de los *servlets* se necesita un servidor de Java (*Java Web Server*). Es necesario integrar la librería dinámica y el *servlet* en un *archivo.jar* necesario para la ejecución en el servidor.

El esquema de la solución software se puede ver en la figura 4-2.

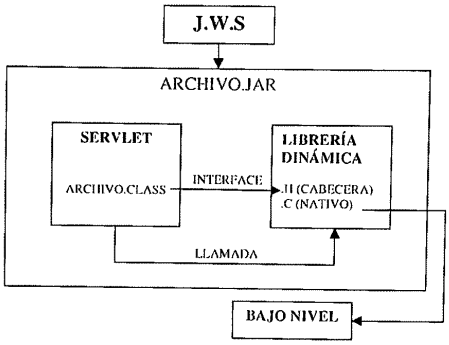


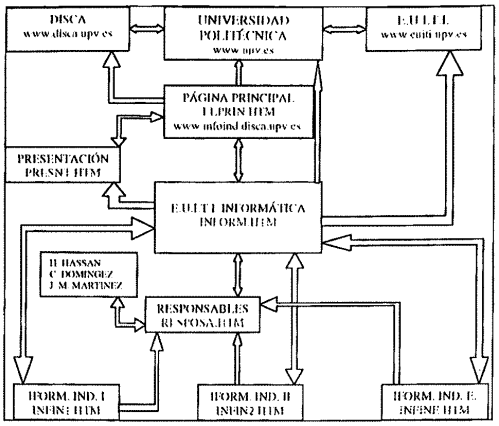
Figura 2. Arquitectura software

3. ESTRUCTURA DEL SERVIDOR

El objetivo de las páginas *Web* del Departamento de Informática de Sistemas y Computadores (DISCA) de la E.U.I.T.I.V. es el de ayudar a los alumnos en el estudio de las asignaturas de informática industrial. Para cumplir con este objetivo hemos incluido en estas páginas una amplia serie de apartados que a continuación pasamos a enumerar:

- La presentación, con información general de las asignaturas del departamento.
- Vínculos con otras páginas relacionadas o de posible interés para el alumno.
- Información sobre los profesores responsables de las asignaturas.
- Información sobre cada asignatura concreta

A continuación se muestra un esquema general de todas las páginas con sus correspondientes vínculos.



La primera página que verá el alumno cuando conecte con el servidor es la página principal, a partir de esta se puede acceder a una página de presentación, a la página de informática de la E.U.I.T.I.V., a la página principal del DISCA y a la página principal de la U.P.V. A partir de la página de informática de la E.U.I.T.I.V. el alumno podrá acceder a toda la información referente a las asignaturas anteriormente descrita. En el esquema, las distintas páginas están representadas por cajas de texto, los vínculos entre ellas se representan con flechas, siendo estas de doble sentido cuando es posible retornar de una página a la anterior a través de un hipervínculo. Los rótulos "*.HTM" indican el nombre del archivo de cada página, y los rótulos "www..." indican la dirección en el *Web*. La página de responsables ofrece información de los profesores que imparten las asignaturas, en ella podemos conocer la ubicación de dichos profesores dentro de la universidad y sus horarios de tutorías, también tenemos la posibilidad de realizar una consulta mediante correo electrónico.

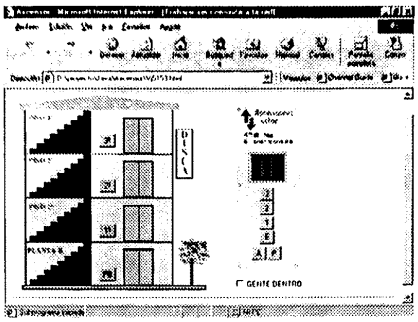
4. SIMULADORES DE LOS TRABAJOS DE LAS ASIGNATURAS

4.1 ASCENSOR

Este *Applet* de Java, que simula el funcionamiento de un ascensor, tiene las siguientes características específicas:

- Utilización de multihilo con el fin de atender a los eventos al mismo tiempo que ejecuta las acciones.
- Información gráfica y sonora en pantalla.
- Interactivo por medio de botones actuables con el puntero del ratón.
- Funcionamiento inteligente para atender las llamadas por orden de nivel.

Cuando se carga el *Applet* en un navegador se puede ver la siguiente pantalla que simula el funcionamiento de un ascensor:



A la izquierda de la pantalla se ve el edificio con los cuatro niveles (tres pisos y planta baja), cada uno con su correspondiente botón de llamada. A la derecha se ve la consola interior del ascensor con el visualizador que indica la posición, los cuatro botones de nivel y dos botones más, uno de alarma y uno de paro. Estos botones interiores sólo funcionan si el cuadro de diálogo GENTE DENTRO está activado.

Los botones se activan pulsándolos con el puntero del ratón, a continuación el programa realizará las acciones pertinentes. Se pueden activar varios botones a la vez, mientras se ejecutan las acciones, sus llamadas se irán atendiendo según el ciclo inteligente. El botón de alarma '*A*', mostrará en el visualizador una letra ***A*** y emitirá un sonido de alarma. El botón de paro '*P*', mostrará en el visualizador de información una letra ***P*** y suspende todas las acciones, excepto la de alarma, hasta que vuelve a ser pulsado y por consiguiente desactivado.

4.2 MÁQUINA EXPENDEDORA

Este *Applet* de Java, que simula el funcionamiento de una máquina expendedora de bebidas, tiene las siguientes características específicas:

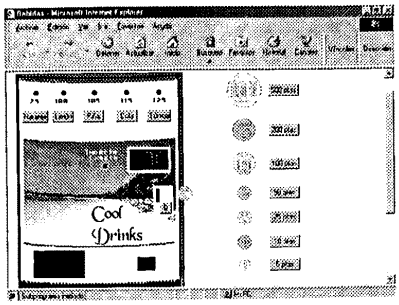
- Utilización de multihilo con el fin de atender a los eventos al mismo tiempo que ejecuta las acciones.

- Información gráfica en pantalla, visualización de la cantidad de dinero introducido, diodo led indicador de cambio agotado, diodo led indicador de bebida agotada, visualización de monedas introduciéndose y de devolución y visualización de bebida expendida.

- Interactivo por medio de botones actuables con el puntero del ratón.

- Funcionamiento inteligente para controlar las existencias, el cambio y el dinero introducido.

En la siguiente imagen se puede ver un ejemplo del funcionamiento del programa simulando el funcionamiento de una máquina expendedora de bebidas:



A la izquierda de la pantalla se muestra la máquina expendedora, en la cual se distinguen los botones de los diferentes tipos de bebida disponibles. Arriba de estos botones, está el precio de cada bebida, también hay un diodo led indicador de bebida agotada. Más abajo está el visualizador mostrando la cantidad de dinero introducido y a su izquierda otro led indicador de importe exacto. Debajo del visualizador se encuentra la ranura para la inserción de las monedas con un botón de devolución a su lado. En la parte inferior de la máquina se encuentran las salidas de latas y de cambio. En la parte derecha de la pantalla, se muestran los siete tipos de monedas admisibles por la máquina. Al costado de cada una hay un botón actuable para introducirlas. En la imagen anterior se puede ver una moneda de 25 pesetas introduciéndose y el visualizador indica que hay 100 pesetas en el interior.

El usuario podrá insertar monedas de cualquier tipo, hasta que el programa considere que ya hay la suficiente cantidad para comprar la bebida más cara (por ejemplo, si se echa una moneda de 500 pesetas, la máquina ya no admitirá más). A continuación se pulsará el botón de la bebida deseada y ésta será expendida por la máquina si la cantidad de dinero que se ha insertado es suficiente. El visualizador mostrará en todo momento la cantidad de dinero, descontando el precio de la bebida en el momento de servirla. El usuario puede pulsar el botón de devolución cuando lo desee para recuperar el dinero que marca el visualizador.

Cuando el diodo led asociado a una bebida esté encendido, la máquina no servirá esta bebida aunque se pulse su botón. Tampoco admitirá cierto tipo de monedas si está encendido el led de importe exacto, puesto que no dispone de cambio para las devoluciones.

4.3 PRÁCTICAS *ONLINE:*

Una de las prácticas que se realizan habitualmente en el laboratorio de la asignatura de Informática Industrial. Consiste en la adquisición de datos analógicos mediante la tarjeta de adquisición PCL-812G conectada a un ordenador. Con esta aplicación se pretende que esta práctica se pueda realizar desde cualquier máquina remota conectada a la red de Internet.

La aplicación realiza las siguientes funciones:

- Solicita, mediante un formulario, la introducción de la ganancia y del número de conversiones (adquisiciones) que se quieran obtener.

- Comprueba si alguno de estos datos solicitados es incorrecto, advirtiendo de ello al usuario.

- Realiza las adquisiciones y sus posteriores conversiones, almacenándolas en la memoria.

- Muestra los resultados en una tabla con su valor digital y su equivalente c voltios.

- Por último, representa en una gráfica los resultados obtenidos. Esta gráfica es interactuable en todo momento por el usuario, pudiendo éste variar su resolución, para apreciar mejor la representación de las señales.

4.3.1 CONTROL DE UN VEHÍCULO GUIADO POR ULTRASONIDOS

Este vínculo hace referencia a un proyecto realizado por dos alumnos de la E.U.I.T.I.V., se trata de un pequeño automóvil que dispone de unos sensores ultrasónicos que le permiten esquivar cualquier obstáculo que se interponga entre el punto de partida y el punto de destino. Para que este coche inicie el movimiento necesita que le sean enviados dos parámetros que corresponden a las coordenadas X e Y del punto de destino. Estos parámetros son enviados a través de un cable conectado al puerto serie de un ordenador, que en nuestro caso será la máquina servidora. Como en el vínculo anterior, el objetivo es poder enviar esos datos desde cualquier ordenador conectado a internet.

Así pues esta aplicación deberá realizar las siguientes funciones:

- Solicita, mediante un formulario, la introducción de la coordenada X y la coordenada Y pertenecientes al punto de destino del coche.

- Comprueba si alguno de estos datos solicitados es incorrecto, advirtiendo de ello al usuario.

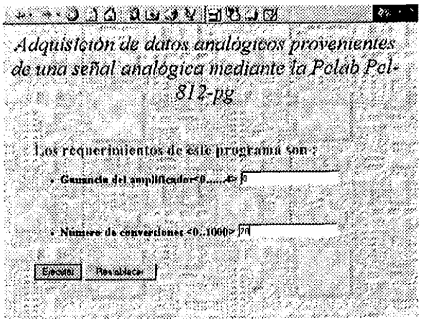
- Escribe los datos en el puerto serie de la máquina servidora para que estos sean adquiridos por el coche.

- Por último, dibuja en la pantalla del usuario, una simulación animada que muestra la trayectoria que realizará el vehículo desde el punto inicial hasta el punto definido por las coordenadas.

Explicación del funcionamiento.

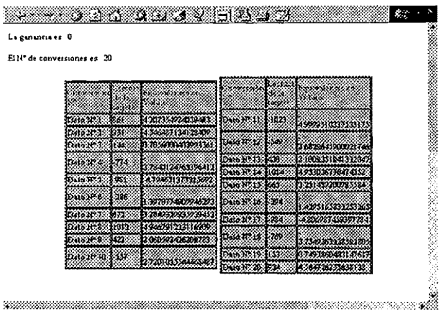
Vistas las especificaciones de función de la interface de prácticas *on-line* y después de exponer cómo se realizaba dicha interface, sólo queda mostrar cómo hay que ejecutar esta aplicación, observando y analizando su funcionamiento.

- Adquisición de datos: Este es el primer ejemplo de la aplicación. Lo primero que se ve cuando se accede a este enlace es el siguiente formulario:

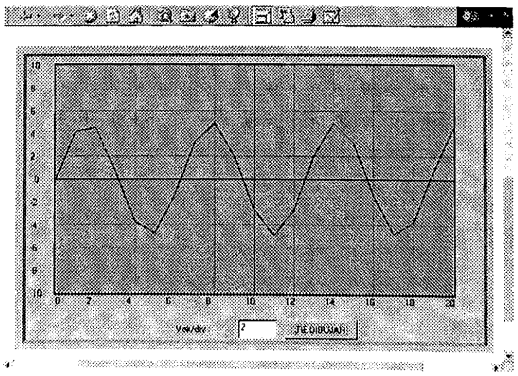


Como se puede observar en la imagen, el usuario debe introducir la ganancia del amplificador de la *Pclab*, entre los valores 0 y 4, también elegirá el número de conversiones que se realizarán entre 0 y1000. Si el valor introducido no está entre estos márgenes, se muestra un mensaje de error.

Una vez introducidos los valores correctamente, se pulsa el botón de ejecutar para que el formulario envíe los datos al *Servlet* de adquisición el cual, tras realizar su función mostrará los resultados en una tabla como la que se puede ver en la siguiente imagen:



Además de la tabla de resultados, en la parte inferior de la página generada por el *Servlet*, se mostrará una gráfica interactiva en la cual el usuario podrá cambiar la resolución de la amplitud de la señal mostrada:



5. CONCLUSIONES

En la presente ponencia se ha presentado un servidor de las asignaturas de informática industrial de la EUITIV. No se trata simplemente de un servidor de páginas de texto, sino que además incorpora un conjunto de simuladores que sirven para que los alumnos conozcan de forma detallada cuáles son las especificaciones de los trabajos finales de la asignatura. Por otra parte, se dispone de la posibilidad de realizar prácticas *on-line*, lo que reduciría considerablemente el coste de los laboratorios puesto que el número de prototipos industriales podría ser más pequeño.

APORTACIONES DE LA INGENIERÍA DE LA PROGRAMACIÓN AL CURRÍCULUM DEL INGENIERO TÉCNICO INDUSTRIAL.

José Ramón Balsas Almagro[1], José Joaquín Aguilera García[1].

[1]*Departamento de Informática. Universidad de Jaén.*
Escuela Politécnica Superior.
Av. Madrid, 35. 23071 Jaén
e-mail: {jrbalsas, jjaguile}@ujaen.es

RESUMEN: La creciente necesidad de conocimientos informáticos en las titulaciones de Ingeniería Técnica Industrial, plantea la actualización de los diseños curriculares de las asignaturas que abordan estos temas para adaptarlas a las nuevas necesidades del mercado. En el presente trabajo se expone la experiencia desarrollada en la docencia de la asignatura Ingeniería de la Programación para Ingenieros Técnicos Industriales.

1.- INTRODUCCIÓN.

En la actualidad, es un hecho que los conocimientos informáticos están cada vez más arraigados en los diversos ámbitos de la sociedad. Su aplicación en el entorno empresarial ha condicionado su inclusión en los diferentes planes de estudio de las titulaciones universitarias. Así, es frecuente encontrarse con asignaturas de fundamentos de Informática de carácter troncal, optativo o de libre configuración en la mayoría de las carreras universitarias.

Las áreas que más conocimientos Informáticos demandan actualmente son aquellas relacionadas con las distintas Ingenierías Técnicas. Aplicaciones como el control y monitorización de dispositivos, la simulación de procesos industriales, el desarrollo de aplicaciones de cálculo o la programación de dispositivos electrónicos, hacen necesario dotar a los planes de estudio de estas titulaciones con conocimientos más profundos en técnicas de desarrollo de software.

En el presente trabajo se exponen las impresiones obtenidas durante cuatro años impartiendo la asignatura de Ingeniería de la Programación a los alumnos de Ingeniería Técnica Industrial, especialidad de Electrónica de la universidad de Jaén, así como las conclusiones a las que se ha llegado sobre los conocimientos en Informática que deben poseer los alumnos de esta titulación a lo largo de sus estudios. Para ello, el trabajo está organizado en los siguientes apartados: en el punto 2 se expone la influencia de la informática en los contenidos del actual plan de estudios de la titulación. En el punto 3 se presenta la asignatura Ingeniería de la Programación intentando justificar los objetivos que persigue, el temario utilizado, la organización de las prácticas y los criterios de evaluación utilizados. En el último punto se presentan algunas conclusiones sobre los resultados obtenidos en la docencia de la asignatura.

2.- LA INFORMÁTICA EN EL PLAN DE ESTUDIOS DE INGENIERÍA TÉCNICA INDUSTRIAL, ESPECIALIDAD EN ELECTRÓNICA.

El papel de los contenidos relacionados con la Informática en el anterior plan de estudios del 72, era meramente residual y estaban integrados como parte de asignaturas de matemáticas. Con el nuevo plan de estudios de 1996 se concede a la Informática un papel más destacado en la formación del alumno, apareciendo en su formación una mayor oferta de asignaturas.

A continuación se destaca la oferta de asignaturas de contenidos Informáticos existentes en el nuevo plan de estudios:

a) Asignaturas Troncales y Obligatorias
- Fundamentos de Informática
- Informática Industrial
- Electrónica Digital

b) Asignaturas Optativas
- Informática Gráfica para Aplicaciones Industriales
- Ingeniería de la Programación
- Microprocesadores
- Redes de Computadores

Existen otro conjunto de asignaturas que utilizan la Informática como herramienta o aplicación:

a) Asignaturas Troncales:
- Automatización Industrial
- Diseño Asistido por Ordenador
- Proyecto Fin de Carrera

b) Asignaturas Optativas
- Controladores Lógicos
- Fabricación Asistida por Ordenador

Además, se utilizan multitud de aplicaciones informáticas de cálculo, diseño y simulación en las asignaturas de: Matemáticas I, Matemáticas II, Sistemas Digitales Avanzados, Sistemas de Adquisición de Datos, Simulación y Diseño Electrónico, Regulación Automática II e Ingeniería del Mecanizado.

En definitiva, podemos concluir que la especialidad de electrónica es la que presenta mayor número de créditos relacionados con la disciplina Informática, siendo las áreas de arquitectura de computadores, programación y estructuras de datos, las más explotadas o desarrolladas en el conjunto del plan de estudios.

3.- LA ASIGNATURA INGENIERÍA DE LA PROGRAMACIÓN.

a) Presentación

Ingeniería de la Programación es una asignatura optativa de carácter cuatrimestral con 6 créditos de carga docente. Tres de los créditos son utilizados para la enseñanza de contenidos teóricos y tres para la

realización de prácticas. A pesar de su carácter optativo, está recomendada para alumnos de segundo curso y se imparte en el primer cuatrimestre. Su ubicación en el calendario académico intenta responder a las necesidades en habilidades de desarrollo de software en las que se basarán otras asignaturas coetáneas y posteriores.

Aunque la asignatura está recomendada para la especialidad de Electrónica, también se oferta a las especialidades de Electricidad y Mecánica, áreas que suelen aportar durante el curso un 15% de los alumnos de la asignatura. Podemos observar en este hecho el interés que en muchos alumnos despierta la especialización de los conocimientos en programación que poseen de primer curso.

Un aspecto importante a la hora de tener en cuenta para el desarrollo de la asignatura es determinar cuáles son los conocimientos que posee el alumno cuando elige esta asignatura. En el caso de Ingeniería de la Programación, los alumnos han cursado en primer curso la asignatura Fundamentos de Informática en la que se les han presentado y han afianzado conceptos fundamentales de la disciplina informática. En particular, traen experiencia en el diseño de pequeños algoritmos para la resolución de problemas relacionados con su área de conocimiento y, a la vez, conocen algunas nociones sobre la utilización de tipos de datos básicos, como pueden ser vectores, matrices y registros.

b) Objetivos

En el plan de estudios de la titulación aparecen cuales son los descriptores que deben guiar los contenidos de la asignatura. Estos son: *Programación Avanzada e Ingeniería del Software.* Es decir, por un lado necesitaremos intensificar y especializar los conocimientos en programación del alumno para permitirle ampliar su visión en diferentes técnicas de programación existentes, enfocando en todo momento el aprendizaje hacia aspectos que le sean de utilidad en su futura trayectoria profesional. Por otro lado, puesto que los desarrollos software que deberá abordar sobrepasarán el nivel de pequeñas aplicaciones, se ve necesario que conozca métodos de Ingeniería del Software que le ayuden en el desarrollo de aplicaciones de complejidad superior.

Sirva como ejemplo el Proyecto Fin de Carrera, donde tenemos un reflejo evidente de la importancia de las técnicas de programación y desarrollo de aplicaciones adquiridas a lo largo de la titulación. Esta asignatura, de carácter obligatorio y que consta de 6 créditos prácticos, debe ser cursada cuando se tengan aprobadas todas las asignaturas de la titulación.

En su gran mayoría, los proyectos fin de carrera consisten o se apoyan en la realización de aplicaciones software. Algunos ejemplos pueden ser: aplicaciones en entorno CAD, aplicaciones para la monitorización de dispositivos de adquisición de datos, programación de simuladores de dispositivos electrónicos (osciladores, entrenadores digitales, etc.) o de instalaciones eléctricas, etc.

Ante esta perspectiva, parece clara la necesidad de que el alumno tenga una buena base en programación que le facilite el desarrollo de aplicaciones de cierta envergadura. Evidentemente, estas técnicas deben ir sustentadas por algunas nociones básicas en Ingeniería del Software que le permitan tener una visión de conjunto del software que deberá desarrollar y mantener, quizá como parte integrante de algún equipo de desarrollo. Además deberá empezar a utilizar herramientas de programación que le puedan ser útiles a lo largo de su actividad profesional.

En consecuencia, los objetivos genéricos que se han planteado en la asignatura son los siguientes:

- Comprender y aplicar los principios de la programación estructurada.
- Aprender a realizar documentaciones sencillas de las aplicaciones desarrolladas.

- Conocer, comprender y analizar distintos Tipos de Datos Abstractos, así como su aplicación en problemas concretos.
- Comprender y utilizar desde una aplicación los diversos recursos de un ordenador.
- Realizar el análisis y diseño de aplicaciones empleando técnicas de Ingeniería del Software.
- Conocer los fundamentos de la programación orientada a objetos.
- Conocer y utilizar algún entorno actual de programación visual.

c) Temario

Para el cumplir de los objetivos planteados se plantea el siguiente temario de teoría:

- Introducción a la Ingeniería del Software
- Metodología de la programación
- Representación de la Información
- Análisis y diseño de aplicaciones
- Gestión de recursos del ordenador
- Introducción a la programación orientada a objetos
- Programación en entornos gráficos

d) Organización de las prácticas

En el desarrollo del programa práctico se debe contar con la limitación inicial en programación con la que parten los alumnos. Sus conocimientos se reducen a la utilización de una notación algorítmica para el desarrollo de pequeños problemas de cálculo. En cuanto a estructuras de datos, su experiencia se reduce a trabajo con vectores y matrices. Además, sus conocimientos en la utilización de un lenguaje de programación concreto son muy limitados, reduciéndose a algún lenguaje de tipo script visto durante 2 o 3 sesiones de prácticas en primer curso.

La necesidad inherente de los alumnos de la titulación de realizar programas para comunicarse con diversos dispositivos electrónicos planteó la posibilidad de utilizar C como lenguaje de programación de alto nivel para el desarrollo de las prácticas. En este sentido, se utilizan de 3 a 4 sesiones de prácticas para familiarizar al alumno con la sintaxis del lenguaje. En estas sesiones introductorias se utiliza un entorno sencillo de programación bajo msdos, RHIDE, de dominio público y con pocas necesidades hardware, para compilar aplicaciones utilizando DJGPP fácilmente accesible por los alumnos a través de la red y respetando la política de licencias de software de la Universidad. Una vez que el alumno ha asimilado la sintaxis del lenguaje se introduce la herramienta Borland C-Builder para el desarrollo de aplicaciones en entorno gráfico.

Es interesante observar la facilidad que muestran los alumnos para adaptarse al nuevo lenguaje, en principio por la predisposición que tienen para su aprendizaje. Por otro lado, la buena base en desarrollo algorítmico que poseen les facilita la labor de implementar algoritmos sencillos vistos durante el primer curso en muy poco tiempo.

Una vez conocido el lenguaje de programación se comienzan a desarrollar los distintos módulos de prácticas, los cuales deben ser realizados por parejas. De esta manera, se van familiarizando con el trabajo en equipo y con las dificultades que se pueden plantear en el desarrollo de software.

Los distintos módulos que se van a desarrollar son:

- Desarrollo de programas que hagan uso de las estructuras de datos y Tipos de Datos Abstractos vistos en teoría.

- Análisis de pequeñas aplicaciones siguiendo la metodología de análisis estructurado y utilizando alguna herramienta CASE para el desarrollo de los diagramas de flujo de datos, diccionario de datos, etc.

- Introducción a la programación visual orientada a eventos utilizando el entorno de desarrollo Borland C-Builder y posterior desarrollo completo: análisis e implementación de alguna aplicación donde pongan en práctica los conocimientos adquiridos durante el curso.

Para la entrega de las prácticas existen unas fechas prefijadas en las cuales la puntuación de las mismas se realizará sobre 12 puntos, incentivando de esta manera el trabajo del alumno y favoreciendo que no se le acumulen para el "último día". Aparte, las prácticas siempre se pueden entregar hasta el día del examen.

e) Método de evaluación

Durante los primeros años en la implantación de la asignatura se optó por un método de evaluación que ponderase los contenidos teóricos y prácticos desarrollados a lo largo del curso. De esta forma, la evaluación del alumno era llevada a cabo mediante la realización de una prueba de carácter teórico que iba a suponer el 60% de la nota definitiva. Dicha prueba estaba compuesta por preguntas de tipo test donde el alumno demostraba su grado de comprensión en los conocimientos adquiridos. Además, debía responder a algunas preguntas abiertas donde ponía en práctica su capacidad de aplicación de las diversas técnicas aprendidas para la resolución de problemas. Por otro lado, se evaluaban las prácticas desarrolladas por cada grupo a lo largo de todo el curso, las cuales suponían un 40% de la nota final.

Tras dos años de aplicación se pudo observar, por los resultados obtenidos y por conversaciones con los propios alumnos, que los objetivos iniciales no se cumplían en su totalidad. Muchos de los alumnos orientaban más su formación a la adquisición de contenidos teóricos que no sabían poner en práctica en el desarrollo de software en otras asignaturas de la titulación. Esta cuestión llevó a potenciar la dimensión práctica para incentivar la motivación y el aprendizaje autónomo del alumno hacia los contenidos de la asignatura.

Actualmente, la calificación final del alumno se obtiene a partir de la evaluación de los tres factores que se pretenden inculcar: la adquisición de nuevos conocimientos y técnicas de programación, la aplicación práctica de las mismas en problemas reales y su capacidad para profundizar en los conocimientos adquiridos mediante la realización de trabajos guiados. Con esta nueva perspectiva, la calificación final se obtiene de la siguiente manera:

- El 40% mediante una prueba escrita de contenido teórico/práctico.

- El 50% mediante la realización de las prácticas por grupos de dos personas a lo largo del curso.

- El 10% realizando un pequeño trabajo guiado que deberá exponer en clase al final del curso.

4.- CONCLUSIONES

El planteamiento presentado en la asignatura a lo largo de los 4 años durante los que se ha impartido ha conseguido interesantes logros. Por un lado, la inclusión de C como lenguaje de programación ha posibilitado a otras asignaturas de la titulación disponer de mayor tiempo para desarrollar sus temarios teóricos y prácticos. Además, la base en programación desarrollada y los conocimientos adquiridos en el análisis de aplicaciones, han facilitado considerablemente la inclusión de prácticas más interesantes en otras asignaturas, en las que el alumno puede seguir aplicando lo aprendido en esta asignatura. Especial mención merece la asignatura Proyecto de Fin de Carrera, donde ha aumentado el número de alumnos interesados en el desarrollo de sistemas software para tareas de control, monitorización, análisis, etc. poniendo en práctica las técnicas aprendidas en la asignatura y afianzadas durante el resto de sus estudio. Para tal fin, el Departamento Informática ha presentado un pequeño manual con los requerimientos, en cuanto a documentación se refiere, que debe tener la parte informática de los proyectos realizados por los Ingenieros Técnicos Industriales

5.- REFERENCIAS

Casals A. et al. "La formación en Informática Industrial". Jenui'98. 1998.

Duran Aranda, E. et al. *"La universidad y las Enseñanzas Técnico-Profesionales. Una visión general"*. VII Congreso Universitario de Innovación Educativa en las Enseñanzas Técnicas. 1999. Huelva.

Domínguez C. *"Innovación educativa en Informática Industrial"*. II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías. 1996.

Perles, A. "Enseñanza de la Informática Industrial mediante proyectos reales simplificados". Jenui'99. 1999. La Almunia de Dª Godina. Zaragoza.

LA IMPORTANCIA DEL USO DE PROCESOS FÍSICOS REALES EN LA ENSEÑANZA UNIVERSITARIA DE LA INGENIERÍA.

Silvia Terrasa [1], Patricia Balbastre [1], Alfons Crespo [1].

[1] *D.I.S.C.A., U.P.V. (Departamento de Informática de Sistemas y Computadores, Universidad Politécnica de Valencia)*
e-mail: sterrasa@disca.upv.es
patricia@disca.upv.es
alfons@disca.upv.es

RESUMEN: Este artículo revisa nuestra experiencia en el uso de procesos reales en la enseñanza universitaria. En concreto describe los resultados obtenidos en un grupo de asignaturas basadas en los Sistemas de Tiempo Real, las cuales se imparten en varias escuelas de la Universidad Politécnica de Valencia. El hecho de impartir estas asignaturas en diversas titulaciones permite, en cierta forma, generalizar los beneficios que aportan los procesos físicos reales en el ámbito universitario con los resultados de una asignatura.

1.- INTRODUCCIÓN.

En los últimos años, gracias a la proliferación de asignaturas optativas en los nuevos planes de estudio y a la especialización de las carreras universitarias, se ha ofrecido la oportunidad para desarrollar nuevas iniciativas que han dado lugar a una mejora considerable de la enseñanza universitaria. Una de estas nuevas iniciativas es la que se presenta en el presente artículo.

La iniciativa parte de la incorporación de asignaturas basadas en Sistemas de Tiempo Real a los nuevos planes de estudio. Este tipo de asignaturas, basadas fundamentalmente en el diseño e implementación de aplicaciones de sistemas de control, requieren un enfoque inminentemente práctico para su mayor comprensión. La primera idea que se tuvo fue basar las sesiones prácticas en simuladores, ya que éstos ofrecen un entorno que más fácil de utilizar y de comprender por los alumnos. Sin embargo, finalmente se optó por introducir los procesos físicos reales, aunque para ello se aumentara la dificultad de las sesiones prácticas.

El resultado de esta iniciativa fue sorprendente ya que a pesar del aumento en la dificultad en las sesiones prácticas, los alumnos estaban mucho más motivados y asimilaban los conceptos teóricos con mayor rapidez. El presente curso académico se ha llegado un paso más lejos y se ha basado la evaluación de la asignatura (al menos en uno de los centros en los que se imparte) en un trabajo final basado precisamente en uno de los procesos reales de que se dispone. La experiencia ha sido muy satisfactoria y por ello se presenta en este artículo.

Este articulo se organiza de la siguiente manera: en la Sección 2 se describen los objetivos concretos de las asignaturas de Sistemas de Tiempo Real. La sección 3 se dedicará a describir los procesos reales de que se dispone, así como las aplicaciones concretas que se pretende conseguir de cada uno de ellos. Finalmente en la Sección 4 se comentan las conclusiones a las que se ha llegado después de la experiencia de este curso académico.

2.- SISTEMAS DE TIEMPO REAL

Actualmente la asignatura de Sistemas de Tiempo Real *(STR)* se imparte en cuatro titulaciones distintas: Ingeniero Industrial, Ingeniero en Automática i Electrónica Industrial, Ingeniero en Informática, Ingeniero Técnico en Informática (en las dos especialidades). En todas ellas se imparte como asignatura optativa de último curso excepto en el caso de Ingeniero en Automática i Electrónica que es una asignatura obligatoria.

La asignatura de *STR*, en todas las titulaciones en las que se imparte, se enmarca dentro de lo que se denomina Ingeniería de Control i los objetivos más relevantes de ésta son saber diseñar e implementar sistemas de control. Las asignaturas de control aportan el conocimiento y la habilidad para comprender y analizar los procesos industriales y los métodos y técnicas para diseñar e implementar el sistema de control. Los ingenieros deben especializarse en muchos aspectos como:

- Teoría de Control.
- Modelización.
- Simulación.
- Técnicas avanzadas de control.
- Programación de propósito general.
- Planificación.

A parte de los objetivos generales de las asignaturas que se engloban en la llamada Ingeniería de Control, la asignatura de *STR* tiene una serie de objetivos concretos:

- Entender las restricciones de tiempo y los elementos que intervienen en el desarrollo de aplicaciones controladas por computadora.
- Especificar el comportamiento del sistema a través de redes de Petri.
- Diseñar e implementar aplicaciones de control por computadora.
- Conocer los lenguajes de alto nivel y sus abstracciones para implementar la concurrencia, comunicación y control del tiempo (Ada'95).
- Conocer la influencia del sistema operativo y los algoritmos de planificación.
- Conocer y asimilar las técnicas de análisis de planificabilidad.

Una de las cosas que hay que tener en cuenta a la hora de poner en práctica estos objetivos, es la titulación donde se va a impartir la asignatura, es decir, aunque globalmente los objetivos son

los mismos, según la titulación se hará hincapié en una aspecto o en otro. Por ejemplo para ingenieros industriales se centrará la atención mas en teoría de control que en aspectos de programación y sin embargo para ingenieros informáticos pasará al revés. A pesar de estas diferencias, en todos los casos se han obtenido mejores resultados una vez introducidos los procesos físicos reales a las sesiones prácticas.

3.- DESCRIPCIÓN DE LOS PROCESOS UTILIZADOS.

Actualmente se dispone de tres procesos físicos distintos:

- Control de llenado de dos depósitos de agua
- Control de guiado de un robot móvil tipo Khepera
- Control de gujado de un robot móvil tipo Lego

A continuación se pasará a describir cada uno de los procesos físicos.

a) Depósitos de Agua

El montaje realizado en el laboratorio consta de un depósito de agua con una salida regulada por una válvula y una entrada de agua. El caudal de salida desemboca en una cuba de la cual se extrae el agua de entrada al depósito mediante una bomba. Para medir el nivel de agua se dispone de un sensor de presión colocado en la base del depósito. Una tarjeta de adquisición de datos conectada al PC proporciona las señales para accionar la bomba y leer elvoltaje de la lectura del sensor. La Figura 1 muestra una foto del montaje.

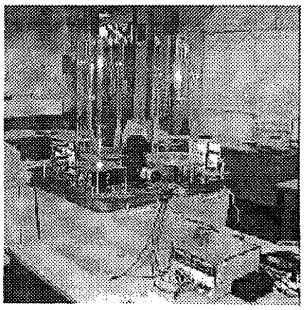


Figura 1 Depósito de agua

Trabajo de los Alumnos

Los alumnos deben diseñar una aplicación que controle el nivel del agua dentro del depósito. La regulación deberá realizarse, en primer lugar con un regulador proporcional y finalmente con un regulador PID (proporcional, integrador y derivativo). Las fases del desarrollo de la aplicación serán: primero se hará trabajar a los alumnos con el proceso simulado probando cada tipo de regulador sobre el modelo y finalmente trabajarán con el proceso real. Las dificultades del trabajo surgen al pasar del modelo simulado al real, ya que deben enfrentarse a errores en la lectura del sensor y otras imprecisiones del sistema.

b) Robot Khepera

El proceso a controlar por los alumnos en el laboratorio es un pequeño robot (Khepera [4]). Mide 27 mm de radio y 2 cm de espesor.

Está provisto de dos ruedas, controladas por dos motores de corriente continua. Además, estas ruedas tienen cada una un encoder incremental que proporciona información odométrica. El sistema de sensores lo forman ocho sensores de infrarrojos con unalcance de 60 mm, situados a lo largo del perímetro del robot. Una imagen del robot Khepera se muestra en la Figura 2.

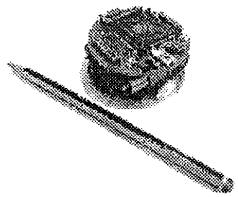


Figura 2 Robot Khepera

Trabajo de los Alumnos

El objetivo es diseñar una aplicación que controle al robot para que, dentro de un entorno con diversos obstáculos, alcance ciertas posiciones dentro de este entorno. Un esquema del proceso puede apreciarse en la Figura 3.

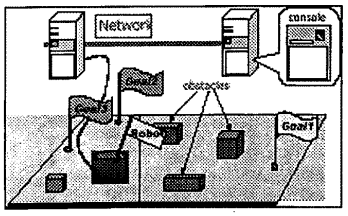


Figura 3 Esquema de la aplicación

Los alumnos deben realizar una aplicación concurrente que, mediante el uso de tareas controle al robot. Además, deben diseñar un algoritmo de evitación de obstáculos y consecución de objetivo, combinando estos dos para obtener una trayectoria válida del robot. En las primeras fases del trabajo se llevarán a cabo las pruebas con un simulador, a fin de que se prueben distintos algoritmos de navegación y el diseño del sistema de tiempo real. Finalmente, cuando el diseño sobre el simulador funcione satisfactoriamente, se probará sobre el robot real, a fin de que los alumnos se den cuenta de los problemas que surgen cuando se trabaja con un proceso físico real.

c) Robot Lego.

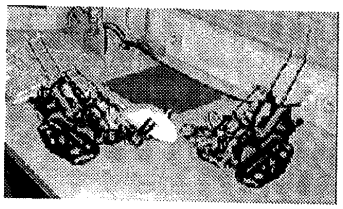


Figura 4.- Robots LEGO

El robot Lego Cybermaster [7] pertenece a la gama de productos de Lego Technic. Consta de una unidad móvil y una torre de comunicación, tal y como puede apreciarse en la Figura 4. La torre se conecta el PC mediante una conexión serie. La comunicación entre la torre y el vehículo se establece por radio. La alimentación se realiza únicamente por baterías. Consta de tres motores, uno por cada rueda y un tercer motor para accionar la pinza delantera. Para conocer la posición del robot, cada rueda dispone de un tacómetro que proporciona los pasos del motor. Dispone, también de un sensor de contacto en la pinza para detectar si se ha chocado con algún objeto.

La comunicación entre el PC y el robot se realiza mediante un control ActiveX suministrado por el fabricante, el cual proporciona las funciones necesarias para comandar el robot.

Trabajo de los Alumnos

El trabajo consiste en diseñar un programa en Ada95, el cual sea capaz de ejecutar una secuencia de ejecuciones automáticas, mejorada con unas aproximaciones manuales, sobre un Robot tipo Lego.

El robot debe de partir desde una posición de inicio, y acercarse a uno de los tres contenedores situados delante de él a una determinada distancia, para depositarlo en una posición de destino. Si no lo encuentra, el programa debe de pasar a control manual, hasta que se detecte el sensor, momento en el cual se pasara a control automático, alzará el contenedor y lo dejará en el punto de partida. El proceso debe repetirse con los otros dos contenedores.

En la Figura 5 se muestra un esquema del proceso.

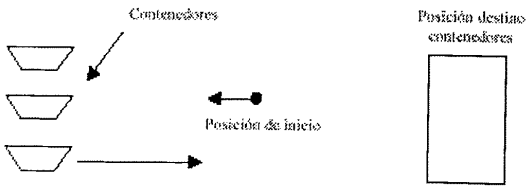


Figura 5 Descripción de la aplicación.

Se pretende que los alumnos realicen la aplicación teniendo en cuenta aspectos de tiempo real pero también atendiendo a los problemas mecánicos que pueden surgir durante el desarrollo de las prácticas. Sobre todo, en cuestiones de odometría y sensorización del robot, donde un simulador no les hubiera proporcionado la misma experiencia.

5.-CONCLUSIONES Y ESPECTATIVAS DE FUTURO.

Este artículo ha examinado la mejora que ha supuesto el introducir procesos físicos reales en las sesiones prácticas de las asignaturas de tiempo real impartidas en varias escuelas de la Universidad Politécnica de Valencia.

El motivo de la introducción de prototipos reales responde a una idea demandada por los propios alumnos, los cuales quieren enfrentarse lo más pronto posible a procesos reales, lo más similares a los trabajos que desempeñarán en el futuro en su vida profesional. Sin abandonar las prácticas con simuladores, el uso de prototipos enseña al alumno las dificultades y diferencias que existen con la simulación de procesos de control.

Aunque ha supuesto un esfuerzo extra, tanto por parte del profesorado como por parte del alumnado, los resultados han sido altamente satisfactorios, viéndose un incremento en la motivación de los estudiantes. Las consecuencias directas de este aumento de la motivación han sido el incremento en trabajos voluntarios de asignatura para profundizar más en el control de los procesos vistos en las sesiones prácticas.

En el futuro se pretende ampliar el número y tipo de procesos y aumentar la complejidad de los trabajos prácticos. También se quiere utilizar los montajes para la docencia en otras asignaturas, como sistemas distribuidos de control, robótica móvil, etc.

6.- BIBLIOGRAFÍA

[1] A. Crespo, J. Vila, F. Blanes, I. Ripoll. Real-Time Education in a Control Engineering Curriculum. *Real-Time Systems Education*. IEEE Computer Society Press. 1998

[2] W.A. Halang., J. Zalewski. A Model for Real-Time Systems Curriculum. *Real-Time Systems Education*. IEEE Computer Society Press. 1996, pp: 39-48.

[3] A. Burns and A. Wellings, Real-Time Systems and their Programming Languages(2nd Edn), Addison-Wesley, 1996.

[4] Balbastre P., Terrasa S., Vila J., Crespo A. Experiences using Ada in a Real-Time and Distributed laboratory Ada Letters Volume 19 Number 3 September 1999 ACM Press pp. 145-157

[5] Khepera User Manual. http://www.k-team.com/

[6] Braitenberg, Valentino, Vehicles: Experiments in Synthetic Psychology, MIT Press, 1987.

[7] Cybermaster Lego Technics http://www.lego.com/cybermaster

ÁREA DE INGENIERÍA DEL SOFTWARE

Responsable: Edmundo Tovar
(Universidad Politécnica de Madrid)
E-mail: etovar@fi.upm.es

ANÁLISIS DE EXPERIENCIAS EN LA IMPARTICIÓN DE INGENIERÍA SOFTWARE I

Calvo-Manzano, J.A., Carrillo, J., Cuevas, G., San-Feliu, T., Tovar, E.

Departamento de Lenguajes y Sistemas Informáticos e Ingeniería de Software
Facultad de Informática
Universidad Politécnica de Madrid
e-mail: etovar@fi.upm.es

RESUMEN: En este artículo se analizan las experiencias obtenidas tras la impartición docente de la materia de Ingeniería Software I. Esta asignatura tiene un fuerte carácter práctico, por lo que el análisis es efectuado uno a uno por los resultados obtenidos por cada una de las práctica. Y, finalmente, se realiza una valoración global, tras lo cual se determinan las acciones propuestas que mejoren su docencia en el próximo curso.

1.- INTRODUCCIÓN.

Durante el presente curso 1999-2000 se comenzó ha impartir por primera vez la asignatura de Ingeniería del Software I del plan de estudios de 1996 en la Facultad de Informática de Madrid de la Universidad Politécnica de Madrid. Ya desde el primer momento nos planteamos desarrollar la asignatura basada en prácticas. De esta forma el alumno dentro de un grupo de trabajo vive la experiencia de los conceptos expuestos en clase.

No se disponía del tiempo necesario para dar exhaustivamente todos los conceptos expuestos en el temario oficial pero hemos intentado cubrir al máximo. Dado el énfasis en el proceso de aprendizaje basado en la sucesión de exposición teórica y aplicación práctica.

Como resultado de la impartición de la Asignatura de Ingeniería del Software, se han logrado alcanzar unos logros y se han detectado una serie de defectos a partir de los cuales se han extraído una serie de conclusiones de mejora para los años posteriores

2.- CONTEXTO DE LA ASIGNATURA.

La asignatura de Ingeniería del Software I es una asignatura troncal en el plan de estudios de 1996 de la Facultad de Informática de la Universidad Politécnica de Madrid que se imparte en el segundo ciclo. La ficha técnica de la asignatura se encuentra descrita en la tabla 1.

Nombre	Ingeniería Software I
Emplazamiento en el Plan de Estudios	4º curso, en el segundo ciclo del Plan 96
Carga lectiva	9 créditos (6 teoricos y 3 prácticos)
Departamento encargado	Lenguajes y Sistemas Informáticos e Ingeniería del Software
Perfil de alumnos	Todos los alumnos han realizado los cursos anteriores en la Facultad de Informática
Nº alumnos	120
Temario	1. Factores Humanos 2. Ingeniería de Sistemas 3. Introducción Ingeniería del Software 4. Modelos de Proceso 5. Evaluación 6. Gestión de Proyectos 7. Gestión de la Configuración 8. Aseguramiento de la Calidad

Tabla 1: Ficha técnica de Ingeniería Software I.

3.- MÉTODO DE ENSEÑANZA.

La metodología docente utilizada no se basa en el método magistral puro (paradigma conductista). El método magistral impone al alumno lo que debe de aprender sin apenas ocuparse en las estrategias o medios para aprenderlo. Sería correcto aplicarlo si la estructura lógica de los contenidos del plan de estudios(ciencia y técnica) en cuanto a producto se correspondiera con las estructuras del proceso implicado en la asimilación por parte del alumno. No obstante las clases presenciales tienen las siguientes virtudes: en las clases se destacan los principios básicos y constituyen un modo de introducción de la nueva materia. Permiten la posibilidad de interactuar con el alumno en el ámbito de la discusión de los problemas. Sin embargo, los defectos más patentes de las clases magistrales son entre otros la tendencia a fomentar comportamientos de carácter pasivo por parte de los alumnos. Este carácter pasivo reduce la efectividad respecto a los métodos que comprometen plenamente al alumno. Por ello, la metodología aplicada este año se basa en un método mixto, que complementa la labor desarrollada en clase con los trabajos prácticos que engloban la totalidad de los contenidos de la asignatura, y que ya fue descrita en la anterior edición de las jornadas jenui. Estos trabajos se desarrollan fuera del horario de la clase.

4.- TRABAJOS PROPUESTOS A LOS ALUMNOS Y SUS RESULTADOS.

Al tener la asignatura que se presenta un contenido práctico importante (3 créditos) la nota de los alumnos se obtiene en un 50% a través de un examen teórico y en el otro 50% a través de la práctica final desarrollada. Siendo necesario para calcular la nota tener aprobado tanto el examen teórico como la práctica a desarrollar y haber preparado tres presentaciones de tres artículos, relevantes según los contenidos de la asignatura, escogidos entre las revistas del sector. Si finalmente los alumnos se deciden a contar al resto de compañeros alguna de las presentaciones realizadas, la nota se incrementa para tener en cuenta el esfuerzo que supone la presentación en público. Los principales materiales didácticos son: los casos y la bibliografía de referencia.

La actividad práctica del alumno se desarrolla a lo largo de todo el curso a través de la serie de prácticas previstas. Se desarrolla un trabajo en grupo sobre un supuesto, del cual se entregan las sucesivas memorias y se realiza una presentación de las soluciones aportadas.

En todo momento se realiza una evaluación progresiva y continua a través de las sucesivas prácticas y de las tutorías con los alumnos.

La labor de control la realiza el propio profesor de una forma continua. La participación del alumno es constante a través de las tutorías.

Los alumnos han realizado a lo largo del curso académico siete trabajos:

1. Factores Humanos

Todos los equipos han determinado su perfil de personalidad y el papel que jugará cada componente del grupo. No hay nada más que comentar que su realización ha permitido tomar conciencia de las características de cada equipo, de manera que se facilite la integración de sus componentes y que conformen un equipo de trabajo para llevar a cabo la actividad del curso

2. Evaluación de Proceso

Esta práctica se ha evaluado utilizando una plantilla que contiene los diferentes contenidos que se piden para los procesos descritos de dos proyectos propuestos: el cumplimiento de cada una de las prácticas base de cada proceso, el cumplimiento de las prácticas genéricas, la situación de cada proceso y de cada categoría, el nivel de madurez de cada proceso y las oportunidades de mejora.

Resultado: Han tenido que repetir 6 grupos de 24, el (25 %). Las notas han sido 9 grupos entre 6 y 6,9; entre 7 y 8,9 hay 4 grupos; entre 9 y 10 hay 5 grupos.

Valoración: Los que han tenido que realizar una segunda entrega de la práctica han alcanzado la máxima nota, por lo que se consideran que los objetivos de aprendizaje han sido cumplidos. La razón fundamental puede residir en que todos los grupos han pasado por el despacho del profesor para explicar la práctica. Ahí es donde realmente se han enterado para qué servía la practica, pues el profesor empleaba una media de 15 minutos explicándoles cómo se utiliza en las empresas, etc. Y ellos también me preguntaban dudas que les surgían.

Sin embargo, es necesario buscar un método para poder evaluar, pues en algunos grupos, no muchos, se notaba que había gente que había trabajado menos, o también que se habían repartido las partes de la práctica y luego a la hora de preguntarles no sabían muy bien el por qué era así.

3. Definición de proyecto

1) Nivel de integración	1.1- ¿Se ha definido el propósito y el contexto del proyecto? 1.2-¿Se ha han realizado supuestos? 1.3.-¿Si se han realizado supuesto que información se ha requerido ampliar? 1.4 ¿Se ha definido el producto? 1.5. ¿Qué áreas de trabajo participan en el trabajo? 1.6. ¿Se han definido los riesgos?
2) Nivel estratégico	2.1 ¿Se han definido los hitos intermedios? 2.2 ¿Se han enlazado con los objetivos finales.? 2.3 ¿Se han definido las responsabilidades de las áreas de trabajo? 2.4 ¿Se han descompuesto los hitos en paquetes de trabajo?
3) Nivel táctico	3.1 ¿Se ha definido el Plan de actividades en forma de red? 3.2 ¿Existen nodos aislados? 3.3 ¿Es la red conexa y con un unico nodo final? 3.4 ¿Es coherente con la definición de hitos intermedios?

Tabla 2: Ficha técnica de Ingeniería Software I.

La evaluación se siguió basándose en una lista de comprobación de los diferentes apartados exigidos en la memoria. Se valoraron cada uno de los puntos que se pedían. En la tabla 2 se presenta resumen los aspectos considerados.

Posteriormente se valoraron otros aspectos como la claridad en la presentación.

Resultado: La nota media obtenida en los trabajos tras una primera entrega es de 6,97. Sólo tuvo que repetir un grupo de los 24 que la realizaron (un 4,16 %).

Valoración: La tasa tan baja de repeticiones se explica por la forma de corrección. Era necesario, previo a la entrega del documento final, entregar una versión en borrador. Sobre esta versión se eliminaban la mayoría de defectos.

4. Estimación Tiempo y Coste

La evaluación se realizó basándose en una lista de comprobación de los diferentes apartados exigidos en la memoria. Se valoraron cada uno de lospuntos en este caso los contenidos de la practica.

Resultado: La nota media tras la primera entrega es de 7,56: Ningún grupo tuvo que repetir.

Valoración: Los alumnos obtuvieron una estimación del coste y tiempo basándose en los algoritmos explicados en clase no presenta mayores dificultades por ello se justifica el hecho de no-existencia de grupos que tuvieran que repetir. La diferente valoración se debe a que se valoraron otros aspectos como es la justificación de los diferentes apartados, informes adicionales presentados y finalmente la claridad en la presentación

5. Gestión Configuración

La corrección se realiza a través de una checklist con los diferentes contenidos a entregar. Posteriormente la nota obtenida se refinaba con aspectos complementarios.

Resultado: La media global es 6, y 12 grupos repiten de 24 (50%).

Valoración: Han aparecido distintos problemas en esta práctica: repartición de partes entre miembros del equipo y falta posterior de integración, copia de lo contado en las clases magistrales. Además, no todos los miembros dominan el tema, por lo que es necesario buscar un método de examinar individualmente a todos los miembros. Este tema tiene un componente eminentemente práctico pero necesita de teoría. Y el problema es que a esas alturas de curso los alumnos se habían repartido las prácticas, y en este en particular se centraban casi en copiar lo contado en teoría.

Olvidaban hacer el plan y se concentraban en describir la configuración (Un aspecto importante pero complementario al plan).

Éste es un tema práctico que debe ser explicado al principio para que los alumnos lo apliquen durante todo el curso. Ello les permitiría un adiestramiento completo en esta técnica.

6. Aseguramiento de la Calidad

En esta práctica se ha pretendido elaborar un Plan de Calidad basado en el contexto del proyecto para que han tenido que realizar previamente una planificación. La evaluación se ha realizado a través de una checklist en el que aparecían aquellos aspectos más importantes para el profesor: descripción correcta de una "policy" de calidad, descripción de un procedimiento,...

Resultado: Todos los grupos superaron esta práctica en la primera entrega. La nota media fue de 7,2.

Valoración: Intencionadamente se decidió no hacer repetir a ningún grupo la realización de la práctica, puesto que ésta se llevó a cabo en las dos últimas semanas del periodo lectivo. Ello significaba que no había tiempo material para sucesivas versiones.

7. Factores Humanos (II)

El curso se ha culminado con un trabajo de autoevaluación realizado como una práctica más realizada como equipo a lo largo del curso. Es en esta evaluación donde se expone y discute las metas marcadas al inicio del curso y las metas que se han logrado.

Esta ultima práctica tiene los siguientes fines:

- Averiguar y reflexionar sobre lo aprendido en el aspecto de trabajo en equipo.
- Revisar lo enseñado y corregir errores.
- Buscar la mejora de los métodos utilizados.

En esta evaluación se ha desarrollado un cuestionario basado en el método Simplex para definir la manera de resolver los problemas por parte de los alumnos. Los resultados del mismo han mostrado que los alumnos de Ingeniería del Software de una manera mayoritaria resuelven los problemas aplicando la experiencia por medio de una implicación personal directa. Estos se muestran en la figura 1.

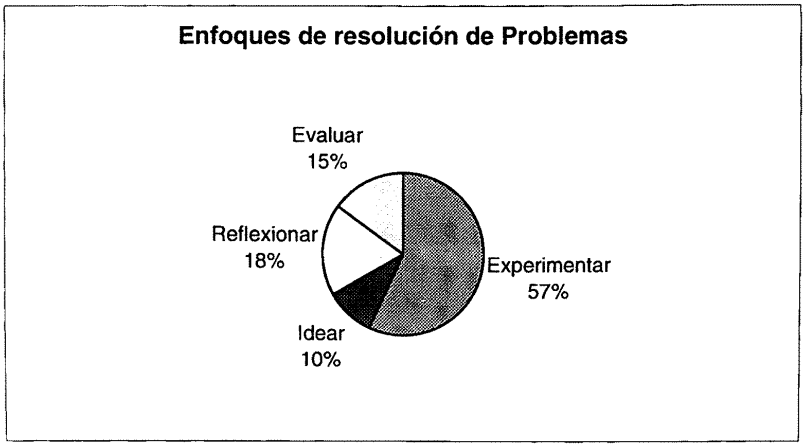


Figura 1: Datos globales obtenidos para la resolución de problemas.

En concreto los valores previstos correspondientes a la reflexión (distante abstracto teórico) y los de evaluar (establecer juicios basados en ideas) e idear (generación de ideas sin valoración) son adecuados.

5.- VALORACIÓN GLOBAL.

Logros obtenidos.

El objetivo principal establecido al principio del curso académico que estribaba en hacer comprender a los alumnos el papel de la dirección y control de proyectos, así como los mecanismos de mejora se ha logrado. El principal obstáculo era la carencia de métodos y de conocimientos de metodologías por parte de los alumnos. Esta docencia será impartida en la asignatura Ingeniería del Software (II) ubicada en el curso posterior.

La meta se ha conseguido articulando los sucesivos trabajos como hitos de un proyecto Esto ha permitido evitar el obstáculo provocado por no disponer de los alumnos con conocimientos en metodologías.

Otros logros que han reconocido los alumnos han sido: Aprender a trabajar en equipo, aprender con vistas a la salida profesional, adquirir conocimientos básicos en I.S, realizar prácticas con calidad, planificación del trabajo disponible y dedicado, distribuir esfuerzos entre los miembros, no tener problemas personales, poner en práctica lo visto en clase, aprender de los errores y cumplir con la planificación.

Defectos detectados.

- Orden de los temas.

El orden de los temas ha influido negativamente en la realización de los trabajos por la distribución de los aspectos de gestión de configuración. Esta apartado de carácter práctico se dispuso en el segundo cuatrimestre. Dado el carácter global de la gestión de configuración llevando el control de todo el producto impidió utilizarse en toda su potencialidad.

- Desacoplamiento entre los trabajos.

Ha surgido una dificultad de acoplamiento e interconexión entre todas y cada uno de las partes de los trabajos prácticos. La disposición de los temas presentes en el plan de estudios ha provocado una dispersión de los esfuerzos en la realización de los trabajos.

Propuesta de mejora.

En definitiva cuando a los alumnos se les ha tutorizado revisando el borrador que ha servido para encauzarles, ha mejorado la valoración. Hay que cuidar la correcta relación y programación en el tiempo de las prácticas, y hay que evitar la tendencia a escribir memorias, centrando a los alumnos en una adecuada presentación. Pues las prácticas son un elemento de comunicación e intercambio de información no de papel.

Además, se propone un nuevo orden de los temas:

1. -Introducción a la Ingeniería del Software.
2. -Factores humanos.
3. -Ingeniería de Sistemas.
4. -Gestión de la Configuración.
5. -Gestión de Proyectos.
6. -Aseguramiento de la Calidad
7. -Modelos de Proceso.
8. -Evaluación

5.- CONCLUSIONES.

Si bien es pronto para obtener resultados más elaboramos sí contamos con la encuesta realizada a los alumnos acerca de los objetivos alcanzados y que ya ha sido presentada en la sección anterior.

Entendemos que la reorganización de los contenidos abundara en la mejora de la docencia. Aunque reconocemos que el perfil de los alumnos debe reforzarse en los aspectos de reflexión y evaluación.

UTILIZACIÓN DE SWEBOK Y PMBOK PARA LA ORGANIZACIÓN DE CONTENIDOS EN LA ESC. SUP. DE INFORMÁTICA DE LA UCLM

Francisco Ruiz[1], Mario Piattini[1], Macario Polo[1]

[1]Grupo ALARCOS
Escuela Superior de Informática. Universidad de Castilla-La Mancha
Ronda de Calatrava, 5. 13071 Ciudad Real
Tel: 926-295300. Fax: 926-295354
e-mail: {fruiz, mpiattin, mpolo}@inf-cr.uclm.es

Resumen: Diversos organismos y asociaciones internacionales han elaborado en los últimos años propuestas de "cuerpos de conocimiento" en materias relacionadas con los estudios de Informática. A continuación presentamos una propuesta de aplicación de SWEBOK (para ingeniería del software) y PMBOK (para gestión de proyectos) en los contenidos de los planes de estudios impartidos en la Escuela Superior de Informática (ESI) de la Universidad de Castilla-La Mancha (UCLM).

1.- INTRODUCCIÓN.

Desde octubre de 1998, en la ESI de Ciudad Real, se imparten las tres titulaciones oficiales de Informática [BOE, 1998].

- Ingeniería en Informática, ISI (1° y 2° ciclo, 5 años).
- Ingeniería Técnica en Informática de Sistemas, ITIS (1° ciclo).
- Ingeniería Técnica en Informática de Gestión, ITIG (1° ciclo).

El año académico 1999/2000 se han impartido los cursos 1°, 2°, 4° y 5° de los planes reformados, mientras que 3° reformado se impartirá a partir de octubre de 2000.

En este documento se presenta un análisis de la situación de los contenidos en Ingeniería del Software (IS) y en Gestión de Proyectos (GP) [1] en los tres planes de estudios citados. Para ello se utilizan como base los dos marcos de referencia que consideramos más significativos [2] al definir contenidos de ambas materias:

[1] Se ha realizado un análisis conjunto de ambas materias por la relación estrecha entre ellas y por el interés para una mejor organización docente, especialmente en cuanto a la intensificación de segundo ciclo en Sistemas de Información.

[2] Esta opinión no excluye la existencia de otras fuentes útiles e interesantes para los docentes de ingeniería y/o gestión de proyectos software como, por ejemplo, la propuesta de curriculum académico en IS detallada en [Bagert et al, 1999], el estándar IEEE para planes de gestión de proyectos software [IEEE, 1998], o la propuesta de MÉTRICA 3 [METRICA, 2000].

- **SWEBOK**: Guide to the Software Engineering Body of Knowledge [SWEBOK, 2000].
- **PMBOK**: A Guide to the Project Management Body of Knowledge [PMBOK, 1996].

En los demás apartados se presentan las asignaturas que tienen contenidos relacionados con IS o GP (apartado 4) y se propone el reparto de los contenidos indicados en SWEBOK y PMBOK entre todas ellas (apartados 5 y 6). Por último, en el apartado 7 se exponen algunas conclusiones.

2.- EL SWEBOK.

SWEBOK es un proyecto conjunto de IEEE-CS y ACM que está siendo desarrollado por el *Software Engineering Coordinating Committee*. El proyecto consta de tres fases estando actualmente a punto de concluir la segunda, de la cual ha sido presentada la versión 0.7 en abril de 2000. En [Bourque et al, 1999] puede encontrarse la presentación del proyecto publicada en IEEE Software. Actualmente SWEBOK está siendo debatido también en el Subcomité JTC1/SC7 de ingeniería del software de ISO.

Los objetivos principales de SWEBOK son cinco:

1. Promover una visión consistente del mundo de la IS.
2. Clarificar el papel –y delimitar las fronteras- de la IS con respecto a otras disciplinas asociadas: ciencia de la computación, gestión de proyectos, ingeniería de computadores, y matemáticas.
3. Caracterizar los contenidos de la disciplina.
4. Proveer acceso a los contenidos del cuerpo de conocimientos.
5. Proveer las bases para desarrollar planes de estudios o materiales para certificaciones individuales.

Áreas de Conocimiento	*Disciplinas Relacionadas*
Requisitos Software	Ciencias Cognitivas y Factores Humanos
Diseño de Software	Ingeniería de Computadores
Construcción de Software	Ciencia de la Computación
Prueba del Software	Gestión y Ciencia de la Gestión
Mantenimiento del Software	Matemáticas
Gestión de la Configuración Software	Gestión de Proyectos
Gestión de la IS	Ingeniería de Sistemas
Proceso de IS	
Herramientas y Métodos en IS	
Calidad del Software	

Tabla 1. Áreas de conocimiento de la Ingeniería del Software y disciplinas asociadas.

En SWEBOK se considera que la Ingeniería del Software está formada por 10 áreas de conocimiento, y que tiene 7 disciplinas relacionadas (tabla 1).

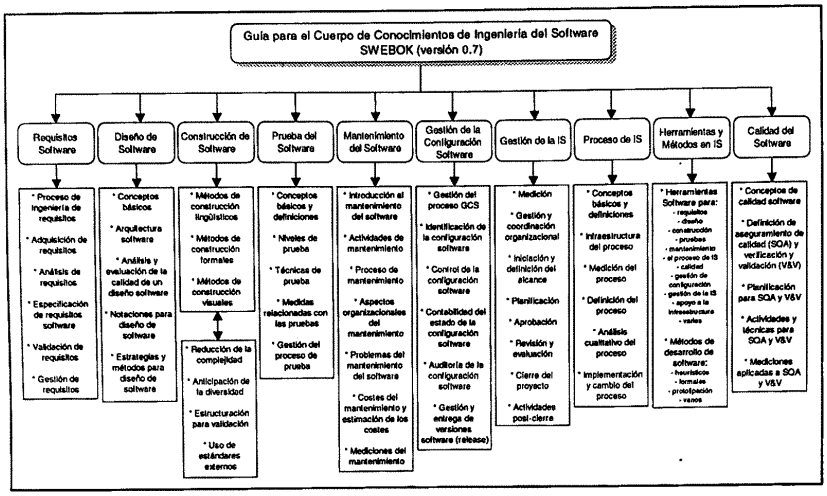


Figura 1. Descomposición temática de SWEBOK (versión Stoneman

Para cada área de conocimiento se propone una descomposición en temas, que en algunos casos se subdividen en otros temas (figura 1). Para cada tema, en SWEBOK se incluyen varios contenidos:

- descripción del tema,
- lista de materiales de referencia (libros o artículos),
- categoría cognitiva según la taxonomía de [Bloom et al, 1956] , y
- referencias a disciplinas relacionadas (fuera de la IS).

GESTIÓN DE PROYECTOS	Grupos de Procesos				
Áreas de Conocimiento	*Iniciación*	*Planificación*	*Ejecución*	*Control*	*Finalización*
G. de la Integración		- Desarrollo del plan del proyecto	- Ejecución del plan del proyecto	- Control global de cambios	
G. del Alcance	- Iniciación	- Planificación del alcance - Definición del alcance	- Verificación del alcance	- Control de cambios en el alcance	
G. de los Tiempos		- Definición de actividades - Secuenciación de actividades - Estimación de la duración de las actividades - Desarrollo del calendario		- Control del calendario	
G. de los Costes		- Planificación de recursos - Estimación de costes - Presupuesto de Costes		- Control de costes	
G. de la Calidad		- Planificación de la calidad	- Aseguramiento de la calidad	- Control de calidad	
G. de Recursos Humanos		- Planificación organizacional - Adquisición del personal	- Desarrollo del equipo		
G. de las Comunicaciones		- Planificación de las comunicaciones	- Distribución de la información	- Informes de ejecución	- Cierre administrativo
G. de Riesgos		- Identificación de riesgos - Estimación de riesgos - Desarrollar respuestas a a los riesgos		- Control de las respuestas a los riesgos	
G. de Adquisiciones		- Planificación de adquisiciones - Planificación de ofertas	- Tramitación de ofertas - Selección de proveedores - Administración de los contratos		- Cierre de contratos

Figura 2 La Gestión de Proyectos según PMBOK (versión 1996).

3.- EL PMBOK.

PMBOK es la propuesta de cuerpo de conocimientos para gestión de proyectos (en general) más reconocida a nivel internacional. Ha sido elaborada por el *Project Management Institute* (PMI). Actualmente está disponible la segunda versión, publicada en 1996 [PMBOK, 1996]. Ha tenido una importante acogida en la comunidad informática al ser reconocido en 1998 como estándar IEEE 1490, y ser utilizado por ISO como base para su futuro estándar 16326 de gestión de proyectos software.

En PMBOK, el cuerpo de conocimientos de la GP se estructura en base a dos dimensiones: 9 áreas de conocimiento y 5 grupos de procesos. Cada uno de los 37 procesos diferentes identificados pertenece a un área de conocimiento y a un grupo (figura 2).

4.- ASIGNATURAS RELACIONADAS.

Las asignaturas de los planes de estudios que tienen relación con la ingeniería del software y/o la gestión de proyectos son [3]:

Curso	Asignatura	Titulaciones	Créditos	Relación
3°	Ingeniería del Software (ISo)	ITIS, ITIG	12	muy alta
3°	Bases de Datos (BDa)	ISI, ITIS, ITIG	9	media
3° (optativa)	Herramientas y Entornos de Programación (HEP)	ISI, ITIS, ITIG	6	alta
4°	Ingeniería del Software I (ISo-1)	ISI	9	muy alta
4°	Planificación y Gestión de Sistemas de Información (PGSI)	ISI	9	muy alta
5°	Ingeniería del Software II (ISo-2)	ISI	9	muy alta

Tabla 2. Asignaturas relacionadas con la ingeniería del software y la gestión de proyectos.

Los contenidos de IS se incluyen fundamentalmente en las asignaturas del mismo nombre (salvo las herramientas software, incluidas en HEP; el diseño y construcción de datos, incluido en BDa; y la gestión de la IS, incluida en PGSI). Los contenidos de GP se presentan en ISo (ITIS e ITIG) y se detallan en PGSI (ISI), salvo la gestión de la calidad incluida en ISo-2.

5.- REPARTO DE LOS CONTENIDOS DE SWEBOK.

Para repartir los contenidos de ingeniería del software hemos considerado los dos niveles de enseñanza propuestos por el SEI (*Software Engineering Institute, Carnegie Mellon University*) en [Bagert et al, 1999]:

> "*in the Small*", es decir, aplicación de los principios de IS al desarrollo de un producto software por un individuo;
> "*in the Large*", es decir, aplicación de los principios de IS al desarrollo de un producto software por un equipo de personas.

Creemos que las características y objetivos de las ingenierías técnicas se ajustan mejor al nivel *Small*, mientras que los de la ingeniería informática de dos ciclos se corresponden más con el nivel *Large* [4]. Con esta filosofía, los contenidos de SWEBOK se distribuyen de la siguiente manera:

- Una introducción a todas las áreas de conocimiento de SWEBOK en ISo (3° de ITIS e ITIG) - profundizando más en el diseño -, y
- Un reparto de los contenidos detallados de SWEBOK entre ISo-1 e ISo-2 (4° y 5° de ISI), salvo gestión de la IS (desarrollado en PGSI).

[3] No se incluye la asignatura troncal de 5° "Sistemas Informáticos" porque, en realidad, consiste en la realización del proyecto fin de carrera. Aunque sólo analizamos los contenidos referidos en SWEBOK y PMBOK, las asignaturas citadas también incluyen algunos temas adicionales, por ejemplo, "Aspectos Éticos y Profesionales" en PGSI.

[4] En ITIS e ITIG sólo se hace una breve introducción a los contenidos de GP - que en ISI se desarrollan en profundidad - y los contenidos de IS son mucho más básicos que en ISI.

Asignatura	Contenidos de SWEBOK (1er Ciclo)	
	Áreas de Conocimiento	Temas
ISo (3°)	Proceso de IS	Conceptos básicos y definiciones
	Requisitos Soft.	Proceso de Ingeniería de Requisitos
	Diseño de Soft.	Conceptos básicos Notaciones para diseño de soft (estructurado y OO.) Estrategias y métodos para diseño de soft. (estructurado y OO.)
	Construcción de Soft.	Métodos de construcción (introducción)
	Prueba del Soft.	Conceptos básicos y definiciones
	Mantenimiento del Soft.	Introducción al mantenimiento ...
	Gestión de la Configuración Soft.	Gestión del proceso GCS (introducción)
	Gestión de la IS	Medición (introducción)
	Calidad del Soft.	Conceptos de calidad soft.
BDa (3°)	Diseño de Soft.	Notaciones para diseño de soft (datos) Estrategias y métodos para diseño de soft. (datos)
	Construcción de Soft.	Métodos de construcción (datos)
HEP (3°)	Herram. y Mét. en IS	Herramientas Software para ...

Tabla 3. Reparto de los contenidos de SWEBOK (1er Ciclo).

Asignatura	Contenidos de SWEBOK (2do Ciclo)	
	Áreas de Conocimiento	Temas
ISo-1 (4°)	Requisitos Soft.	Completo (los 6 temas)
	Diseño de Soft.	Conceptos básicos Notaciones para diseño de soft (estructurado) Estrategias y métodos para diseño de soft. (estr.)
	Construcción de Soft.	Métodos lingüísticos Métodos visuales
	Prueba del Soft.	Completo (los 5 temas)
	Mantenimiento del Soft.	Completo (los 7 temas)
	Gestión de la Configuración Soft.	Completo (los 6 temas)
	Herramientas y Métodos en IS	Completo (los 2 temas)
ISo-2 (5°)	Diseño de Soft.	Arquitectura software Análisis y evaluación de la calidad de un diseño Notaciones para diseño de soft (OO) Estrategias y métodos para diseño de soft. (OO)
	Construcción de Soft.	Métodos formales
	Proceso de IS	Completo (los 6 temas)
	Calidad del soft.	Completo (los 5 temas)

Tabla 4. Reparto de los contenidos de SWEBOK (2do Ciclo).

6.- REPARTO DE LOS CONTENIDOS DE PMBOK.

Siguiendo la diferenciación en los niveles *Small* y *Large*, los contenidos de PMBOK para los ingenieros técnicos deberán ser más básicos que para los ingenieros de 2º ciclo. Por esta razón, el reparto de contenidos propuesto se centra en la asignatura PGSI de 4º, que incluye todo el PMBOK salvo la Gestión de la Calidad, ya contemplada en SWEBOK (dentro de ISo-2):

Asignatura	Contenidos de PMBOK
ISo (3º)	Gestión de Proyectos / Gestión de la IS: el modelo PMBOK.
PGSI (4º)	Gestión de Proyectos / Gestión de la IS: el modelo PMBOK. Gestión de la Integración y el Alcance Gestión de los Tiempos Gestión de los Costes G. de Riesgos G. de Recursos Humanos G. de las Comunicaciones G. de Adquisiciones

Tabla 5. Reparto de los contenidos de PMBOK.

7.- CONCLUSIONES.

Los docentes en ingeniería del software y gestión de proyectos informáticos tienen a su disposición unas herramientas muy interesantes: los cuerpos de conocimiento conocidos como SWEBOK y PMBOK. Estos documentos, complementados con otros - referidos en la bibliografía - y, por supuesto, con las restricciones e indicaciones que establece la legislación vigente y cada plan de estudios particular, son una ayuda muy útil para:

- definir mejor los contenidos de las asignaturas correspondientes, y
- orientar la reforma de los planes de estudios de Informática.

8.- BIBLIOGRAFÍA.

[Bager et al, 1999] Bagert, D.J., Hilburn, T.B., Hislop, G., Lutz, M., McCracken, M. y Mengel, S. *Guidelines for Software Engineering Education Version 1.0.* Technical Report, Carnegie Mellon University – Software Engineering Institute CMU/SEI-99-TR032. Disponible en http://www.sei.cmu.edu/collaborating/ed/workgroup-ed.html.

[Bloom et al, 1956] Bloom, B., Englehart, M., Furst, E., Hill, W. y Krathwohl, D. *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain.* Disponible en http://chiron.valdosta.edu/whuitt/col/cogsys/bloom.html.

[BOE, 1998] Boletin Oficial del Estado. *Planes de Estudios de la Escuela Superior de Informática de la Universidad de Castilla-La Mancha*: ITIG (BOE de 19-enero-1999, pg. 2497), ISI (BOE de 19-enero-1999, pg. 2504), ITIS (BOE de 20-enero-1999, pg. 2808).

[Bourque et al, 1999] Bourque, P., Dupuis, R., y Abran, A. The Guide to the Software Engineering Body of Knowledge. *IEEE Software*, vol. 16, no. 6, 1999.

[IEEE, 1998] IEEE std 1058 – 1998: *Standard for Software Project Management Plans.* IEEE Computer Society Press, USA.

[METRICA, 2000] Ministerio de Administraciones Públicas - Consejo Superior de Informática. *MÉTRICA 3: Metodología de planificación y desarrollo de Sistemas de Información.* Disponible en http://www.map.es/csi/pg5m42.htm.

[PMBOK, 1996] Project Management Institute. *A Guide to the Project Management Body of Knowledge.* PMI Publishing Division, USA 1996. Disponible en http://www.pmi.org/publictn/pmboktoc.htm.

[SWEBOK, 2000] Software Engineering Coordinating Committee (IEEE-CS y ACM). *Guide to the Software Engineering Body of Knowledge* (Stone Man versión 0.7). Disponible en http://www.swebok.org.

DOCENCIA EN GESTIÓN Y ESTIMACIÓN DE PROYECTOS SOFTWARE: UN ENFOQUE CONSTRUCTIVISTA

Isabel Ramos Román[1], Joaquín J. Domínguez Torrecilla[2]

Dpto. de Lenguajes y Sistemas Informáticos. Universidad de Sevilla
e-mail[1]: isabel.ramos@lsi.us.es, e-mail[2]: jdt@argen.net

RESUMEN: un factor primordial en la gestión y estimación de proyectos consiste en conocer la toma de decisiones más adecuada, para cumplir con los objetivos del mismo. En este trabajo se propone un enfoque constructivista para la docencia en gestión y estimación de proyectos software. Este enfoque consiste en dar al alumno criterios que le permitan descubrir, por sí mismo, para cada binomio proyecto-entorno, cuáles son las reglas de decisión más adecuadas. Además de desarrollar el enfoque didáctico propuesto, se presentan los medios utilizados: el estudio de bases históricas y la simulación dinámica de proyectos.

1.- LA GESTIÓN Y ESTIMACIÓN DE PROYECTOS SOFTWARE.

Existen tres factores fundamentales en un Proyecto de Desarrollo de Software (en adelante, PDS): el coste, el tiempo y la calidad del producto obtenido. El objetivo de cualquier gestor de proyectos, siempre será minimizar el coste y el tiempo del proyecto y maximizar la calidad del producto obtenido (figura 1).

Por lo tanto, la problemática de la gestión y estimación de PDS se fundamenta en la elección de las decisiones que se han de tomar, en el transcurso del proyecto, para dirigir, en lo posible, los resultados finales hacia condiciones óptimas de coste, tiempo y calidad.
Para ello, el gestor del PDS se podrá apoyar en diferentes herramientas y/o métodos que le ayuden a realizar dicha toma de decisiones.

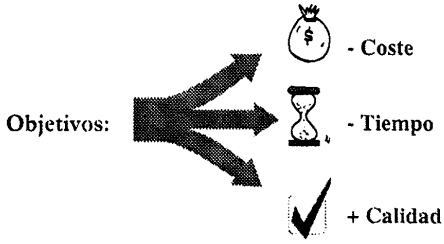


Figura 1: Objetivos del gestor de proyectos.

2.- POSIBILIDADES EN LA DOCENCIA DE LA GESTIÓN Y ESTIMACIÓN DE PROYECTOS.

A la hora de impartir docencia sobre gestión y estimación de PDS, desde nuestro punto de vista, se pueden seguir dos alternativas:

a) Enseñanza de herramientas y técnicas básicas.

Enseñar una serie de conceptos, técnicas y herramientas básicas de ayuda, tales como diagramas en red, utilidades para la gestión del personal, aplicaciones para el control de proyectos, modelos de estimación, etc..

De esta manera conseguiríamos proporcionar al alumno los medios necesarios para comprender los aspectos teóricos del contenido temático. Pero, ¿se ajusta esto a la problemática de la gestión de PDS? Pensamos que no, puesto que no se enseña a realizar la toma de decisiones y las implicaciones que conlleva.

b) Enseñanza para la toma de decisión.

El objetivo en este caso es enseñar al alumno a aprender de la experiencia de otros proyectos para que le permitan, en un futuro, obtener reglas de decisión complejas para un entorno de trabajo concreto. Esta enseñanza para que sea eficaz se debe complementar con los conocimientos teóricos del apartado anterior.

Nuestra experiencia nos indica que con esta alternativa, se consigue que el alumno sea capaz de deducir por sí mismo y con cierta rigurosidad científica, cuáles serán las políticas de gestión más acertadas para un entorno de trabajo real y las implicaciones que pudieran tener a corto-medio plazo sobre el propio proyecto.

3.- UN ENFOQUE CONSTRUCTIVISTA.

Según [Trianes y Gallardo, 98]l: "Aprender significa progresar, mediante la ayuda que la enseñanza puede proporcionar, a estructuras y procesos cognitivos más potentes y complejos". Como complemento de esta definición podemos decir que la enseñanza no es sólo proporcionar conocimientos, sino promover diversos tipos de actividad cognitiva en el alumno que le inciten a adquirir dichas estructuras y procesos cognitivos.

Por desgracia, se suelen olvidar estos conceptos pedagógicos, reduciendo la docencia, en algunas ocasiones, a meras exposiciones de conocimientos científicos. Es por lo que se ha intentado en la docencia de estimación y gestión de PDS, aportar un enfoque constructivista. Según esta teoría, el alumno forma parte activa del proceso de aprendizaje. Es decir, no es un mero espectador que recibe una serie de conocimientos que debe asimilar, sino que es parte activa, construyendo él mismo su nueva estructura mental, con la ayuda de la enseñanza.

Dentro de este espíritu, Bruner [Trianes y Gallardo, 98] creó un modelo de aprendizaje, en el que el alumno debía *descubrir* los conocimientos por sí mismo. De esta manera, la enseñanza debe ser *inductiva*, partiendo de ejemplos particulares para llegar a reglas generales que serán descubiertas por los alumnos. No es nuestro interés hacer un estudio en profundidad de los distintos modelos de aprendizaje, siempre tan controvertidos... Pero sí queremos revelar que, según nuestra experiencia, en cursos superiores y para la enseñanza de materias tales como la presente, este modelo se muestra sumamente productivo. A continuación se comenta con más detalle el enfoque propuesto: consiste en no exponer a los alumnos cuáles son las reglas de decisión generalmente aceptadas en estimación y gestión de proyectos, sino en proponer dos prácticas, mediante las cuales el propio alumno tendrá que descubrir cuáles son el conjunto de reglas y políticas de decisión, en definitiva la toma de decisiones

más adecuada, para conseguir con los objetivos del proyecto. Este método, ya aplicado en otras disciplinas, ha demostrado tener varias ventajas:

- Se consigue que el alumno aprenda más y más fácilmente mediante el proceso de descubrimiento que mediante un simple proceso memorístico.

- En la gran mayoría de los casos se consigue aumentar la motivación y participación de los alumnos.

- Se incentivan ejes transversales de aprendizaje, como la utilización de métricas, el método de trabajo, el trabajo y la discusión en equipo, etc.

4.- NUESTRA PROPUESTA DOCENTE.

Según lo visto hasta ahora, podemos plantear nuestra propuesta docente de la siguiente manera:

La parte teórica (teoría y problemas) se simultanea con la parte práctica (laboratorio), de manera que se vaya aportando los conocimientos básicos necesarios para el desarrollo de las mismas. Los bloques fundamentales de la parte teórica son:

- Naturaleza y problemática de la estimación y gestión.
- Bases Histórica y obtención de métricas.
- Modelos estáticos de estimación.
- Simulación de modelos dinámicos.
- Planificación de la calidad.
- Planificación del riesgo.

Las prácticas, relacionadas con el tema que nos ocupa, son:

- Análisis de los datos de una base histórica.
- Simulación de proyectos.

El objetivo de ambas prácticas es común: que el alumno obtenga, por sí mismo, la toma de decisiones más adecuada para estimar y gestionar su proyecto.

En la primera práctica, el alumno (constituido en grupo) dispondrá de una base histórica de PDS, que contiene datos de un número considerable de proyectos, realizados en un determinado entorno de desarrollo. A partir de dichos datos y con los medios que consideren adecuados los alumnos deberán obtener información útil para la toma de decisiones. Es decir, realizarán cálculos estadísticos sencillos para cruzar los datos de la base histórica que le permitan descubrir relaciones causales entre las decisiones y los resultados en calidad, coste y tiempo. Dichas relaciones causales se pueden, finalmente, traducir en reglas de decisión del tipo: *Cuanto mayor es la rapidez con la que se realiza la incorporación de los técnicos nuevos en el proyectos mejores resultados se han obtenido en el tiempo de entrega aunque el coste del proyecto es mayor.*

En la segunda práctica, el alumno dispondrá de un simulador de PDS basado en un modelo dinámico [Ramos y Ruiz, 99], desarrollado en nuestro departamento. Con el simulador los alumnos podrán realizar análisis de dos tipos: análisis a priori, es decir, podrán simular su comportamiento futuro de un

proyecto en función de las decisiones que se tomen en el presente y análisis post-mortem, es decir, para un proyecto ya terminado, podrán simular qué hubiera pasado si se hubieran tomado otras decisiones diferentes a las tomadas para ese proyecto.

En cualquier caso, mediante el simulador los alumnos podrán definir las políticas de gestión más acertadas para un proyecto en concreto y contrastar los resultados obtenidos de la base histórica de proyectos de la primera práctica. A continuación se presenta los aspectos más importantes de la primera práctica, objeto de este trabajo.

5.- ANÁLISIS DE UNA BASE DE DATOS HISTÓRICA.

Una *Base Histórica de Proyectos de Desarrollo Software* (en adelante, BHPDS) almacena una serie de datos de los distintos proyectos software realizados hasta la fecha. La finalidad de estas bases históricas es la de recopilar información útil de distintos PDS para poder analizarlos y deducir las relaciones causales existentes entre las distintas métricas (figura 2). De este modo se podrán extraer de la BHPDS un conjunto de normas o reglas, (útiles para nuestra empresa y equipo en concreto), que nos ayuden en la estimación y gestión de proyectos para cumplir con los objetivos del mismo. En definitiva, aprender de la experiencia de otros proyectos.

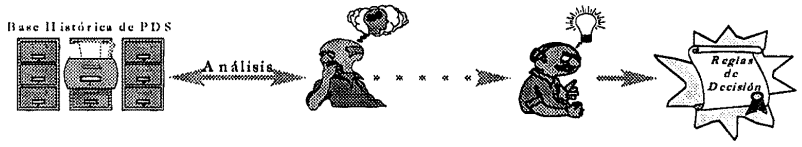


Figura 2: Pasos para la obtención de información a partir de los datos de un base Histórica de proyectos.

En este sentido, se afirma en [Pressman, 98]: "*La medición se puede aplicar al proceso del software con el intento de mejorarlo. Se puede utilizar para ayudar en la estimación, el control de calidad, la evaluación de la productividad y el control de proyectos...*".

En la práctica propuesta a los alumnos, se les proporciona una BH que posee información de unos 20 medidas distintas, realizadas sobre unos 200 PDS de un mismo entorno de trabajo y generadas por un simulador de proyectos [Ramos y Ruiz, 98]. Estas medidas se describen someramente en la tabla 1.

El sentido de esta práctica, como ya hemos indicado, es que el alumno *descubra*, por sí mismo, cuáles son las relaciones y valores de las métricas en los proyectos que han cumplido (según el criterio de la organización de desarrollo) con los objetivos de tiempo, coste y calidad. Estos objetivos (que se ajustarán a cada entorno) son del tipo:

- Se considera que el tiempo de entrega y el coste del proyecto son muy buenos si los valores finales son iguales o menores a los estimados. Se consideran buenos si los valores finales no superan a los iniciales en más de un 20 %. Aceptables si están entre un 20 y un 50 %.

- Se considera que la calidad del producto final es buena si el número medio de errores por tarea al final del proyecto (errores latentes) no supera al estimado en más de 10 %.

			MEDIDA
Iniciales	Predictivas	**Tamaño**	1. Esfuerzo estimado 2. Tiempo estimado 3. Tareas estimadas
		Complejidad	4. Dedicado a formación
	Propias del entorno o de las Políticas de Gestión	P.G. de Asignación de Esfuerzo	5. Esfuerzo de desarrollo 6. Dedicación
		P.G. de Personal	7. Tiempo de adecuación 8. Tiempo de contratación 9. Técnicos estimados
		P.G. de Condiciones del Tiempo de Entrega	10. Aplazamiento permitido
Finales	Resultados relacionados con el Tamaño		11. Esfuerzo gastado 12. Tiempo de entrega 13. LDC 14. Tareas
	Resultados relacionados con la Gestión del Personal		15. Técnicos expertos 16. Técnicos nuevos
	Resultados relacionados con el producto final obtenido		17. Desviación de calidad 18. Desviación de tiempo 19. Desviación de coste

Tabla 1: Métricas utilizadas en la Base Histórica de proyectos.

A partir de la información anterior se le sugiere al alumno el siguiente modo de operación: en primer lugar, se deben descartar aquellos proyectos que no hayan cumplido con los objetivos de calidad. A continuación se obtienen diferentes muestras de proyectos en función de los objetivos de tiempo y coste. De esta manera, se podrá trabajar con muestras de proyectos en los que el tiempo y el coste son, simultáneamente, muy buenos y/o buenos, muestras en las que el tiempo es muy bueno y/o bueno independientemente de los valores obtenidos en el coste, etc.

Una vez obtenidas las muestra de proyectos, se realizarán los estudios estadísticos y análisis correspondientes. El análisis realizado debe ir encaminados a determinar el comportamiento de la calidad, el tiempo y el coste, por separado y simultáneamente, en función de las métricas proporcionadas. La cantidad y calidad del análisis realizado, los determina el propio equipo de trabajo. Para obtener información de la BHPDS, bastará con utilizar una herramienta de hojas de cálculo que incluya estadísticas y gráficos.

Por ejemplo, si analizamos el comportamiento de la desviación en el tiempo de entrega, en función del porcentaje de técnicos expertos que utilizamos en el proyecto, obtendremos los siguientes resultados (tabla 2):

		Desviación Tiempo				
		Muy Buena	Buena	Aceptable	Mala	TOTAL
%Tec. Exp. (Real)	Bajo	0	2	2	5	9
	Medio	7	9	7	6	29
	Alto	35	5	4	4	48
	TOTAL	42	16	13	15	86

Tabla 2: Relación entre la desviación de tiempo y el porcentaje de técnicos expertos. Muestra de proyectos con tiempo muy bueno y bueno.

Presentando esta información gráficamente podemos visualizar mejor los resultados obtenidos. Por ejemplo, partir de la figura 3, se puede enunciar la siguiente relación causa-efecto: *"Cuanto mayor es el porcentaje de técnicos expertos en el proyectos menor es la desviación de tiempo, es decir, mejores resultados se obtienen en el tiempo de entrega de los proyectos"*. Podemos comprobar también que se obtienen excelentes resultados en tiempo cuando el porcentaje de técnicos expertos en el proyecto está por encima del 70%.

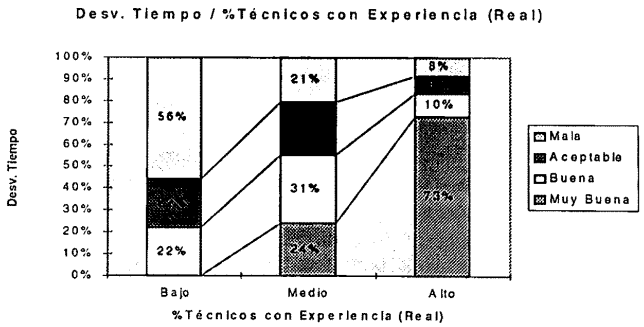


Figura 3: Relación entre el tiempo de entrega y el porcentaje de técnicos expertos en el proyecto.

Una vez terminado el estudio estadístico, se obtienen un conjunto de relaciones causales, a partir de las cuales, se pueden deducir las reglas de decisión que han llevado al cumplimiento de los objetivos propuestos. Las reglas obtenidas, para los proyectos de la BHPDS, ayudan a estimar y gestionar futuros proyectos y a conocer mejor las relaciones que se establecen entre las diferentes métricas del proyecto.
Por otro lado, los alumnos pueden comprobar la importancia que tiene para la estimación y gestión de futuros proyectos la información histórica de los diferentes proyectos que se ha realizado en su organización de desarrollo.

6.- CONCLUSIONES.

En este trabajo se ha presentado una experiencia docente que da a los alumnos la posibilidad de obtener información útil para la toma de decisiones en la estimación y gestión de proyectos a partir de una base Histórica de Proyectos. De esta manera podrán aprender del pasado y de la experiencia de otros gestores de proyectos y de la importancia que tiene la recogida de datos métricos de los proyectos en desarrollo. Finalmente destacar, el alto nivel de participación y discusión que se produce dentro de cada grupo de trabajo y la posibilidad de contrastar, posteriormente, las reglas obtenidas con un simulador de proyectos.

7.- REFERENCIAS.

[Pressman, 98] Pressman, Roger S.: *"Ingeniería del Software. Un Enfoque Práctico"*. McGraw-Hill, Madrid, 1998.

[Ramos y Ruiz, 99] Ramos, I.; Ruiz, M.: *"Los Modelos Dinámicos y la Formación en Gestión de Proyectos"*. V Jornadas sobre la Educación Universitaria de la Informática, Zaragoza, 1999.

[Ramos y Ruiz, 98] Ramos, I.; Ruiz, M.: *"A Reduced Dynamic Model to Make Estimations in the Initial Stages of a Software Development Project"*. In: Hawkings, C. et al (ed.), INSPIRE III. Process International through Training and Education, 172-185. London, 1998.

[Trianes y Gallardo, 99] Trianes, M.V.; Gallardo, J.A.: *"Psicología de la Educación y del Desarrollo"*. Pirámide, Madrid, 1998.

AUDITORÍA INFORMÁTICA DEL DESARROLLO DE APLICACIONES

Darío Usamentiaga[1], Isabel Sevilla[2], Javier Tuya[2]

[1]*Escuela Técnica Superior de Ingenieros Industriales e Ingenieros Informáticos de Gijón*
e-mail: zz98f049@opalo.etsiig.uniovi.es

[2]*Universidad de Oviedo. Departamento de Informática*
e-mail: [sevilla|tuya]@lsi.uniovi.es

RESUMEN: Este trabajo pretende continuar con la auditoría llevada a cabo el pasado año, de los trabajos realizados para la asignatura de "Ingeniería del Software" de 3°. En el nuevo estudio realizado son auditados todos los proyectos entregados por los alumnos de 4° curso de la Escuela Técnica Superior de Ingenieros Industriales e Ingenieros Informáticos para la asignatura de Ingeniería del Software I, en un ámbito de desarrollo más similar al de una organización de desarrollo. Los proyectos son realizados por equipos formados por alumnos, los cuales tienen por usuarios y equipos de aseguramiento de calidad a distintos alumnos de "Ingeniería del Software II" (de 5° curso). Los directores de los proyectos son los profesores responsables de las asignaturas. Para el desarrollo del ciclo de vida de los proyectos software, se sigue un Manual de Procedimientos que define el proceso de desarrollo y adapta la metodología Métrica Versión 2.1 al entorno en que se realizan los proyectos

1.- INTRODUCCIÓN.

La auditoría de los sistemas de información se define como "el conjunto de Procedimientos y Técnicas para evaluar y controlar total o parcialmente un sistema Informático, con el fin de proteger sus activos y recursos, verificar si sus actividades se desarrollan eficientemente y de acuerdo con la normativa informática y general existentes en cada empresa, y para conseguir la eficacia exigida en el marco de la organización correspondiente" [ACHA1994, p. 21]. En resumen, un estudio metódico con vistas a mejorar la rentabilidad, seguridad y eficacia.

Los objetivos fundamentales de la auditoría que da lugar a esta comunicación son, en primer lugar, comprobar el grado de comprensión alcanzado por los alumnos de la metodología utilizada, y la obtención de una práctica adecuada en el desarrollo software. Por otra parte, se evalúa la adecuación del Manual de Procedimientos seguido, así como las normas establecidas por los responsables de la asignatura y la estructura fijada durante el curso.

2.- METODOLOGÍA ISACA

Para la realización de la auditoría se ha utilizado la metodología propuesta por la ISACA™ (*Information Systems Audit and Control Asociation*) [ISAC2000]. Es la asociación que se ha instituido para desempeñar el papel de fuente central coordinadora de estándares de prácticas de control para tecnologías de información. Sus actividades de estándares establecen una base de calidad según la cual se miden las actividades de auditoría y de control. La ISACA ha determinado que la naturaleza especializada de la auditoría de los sistemas de información y las habilidades necesarias para llevar a cabo este tipo de auditorías, requieren el desarrollo y la promulgación de Normas Generales para la Auditoría de los Sistemas de Información y éstas son las que se han seguido.

3.- TÉCNICA UTILIZADA.

En cuanto a las herramientas que son utilizadas, la principal son los Objetivos de Control [PALA1990]. Estos objetivos de control, deben ser coherentes con la metodología utilizada, además tienen que ser específicos y muy completos de acuerdo con el tipo de auditoría que se realice. Como en este caso la auditoría es de Desarrollo de Aplicaciones, se sigue el Manual de Procedimientos [TUYA1999] para las fases de Análisis, que tiene como guía la metodología Métrica versión 2.1, y para las fases de Diseño e Integración nos basaremos principalmente en la metodología Métrica [MAP1995] debido a que el Manual en estas fases es una síntesis de ella. Un ejemplo, tanto del Manual de Procedimientos como de los Objetivos de Control, se puede ver en la Figura 1 y en la Figura 2, respectivamente.

1.2.2.3. Tarea T.RTF-ARS Revisión Técnica Formal del ARS
Revisión técnica formal del ARS. La revisión se realizará de forma independiente por parte del EGC y del USR. Una vez entregado el documento de revisión tendrá lugar una reunión conjunta de revisión con todos los integrantes del proyecto.
Para la segunda versión del ARS 1se realizará otra revisión para comprobar si todos los problemas encontrados han sido resueltos y/o se han introducido nuevos defectos. En esta ya no participará el USR y no existirá reunión de revisión

Procedimientos asociados	***Productos***
I.ERTF Elaboración de una Revisión Técnica Formal I.RRCR Realización de Reuniones Conjuntas de Revisión	D.ARS-HCR Hoja de Comentarios de Revisión del ARS D.ARS-HCU Hoja de Comentarios de Usuario del ARS D.ARS-LAC Lista de Acciones Correctivas del ARS

Figura 1. Ejemplo del Manual de Procedimientos

- **CONTROL B3-2**: Se debe especificar el modelo lógico de datos del nuevo sistema identificando las entidades de datos implicadas y las relaciones existentes entre ellas. Se debe comprobar que:

  1. Existe el modelo lógico de datos del sistema a desarrollar.
  2. El modelo lógico de datos está normalizado al menos en 3ª forma normal.
  3. Existe una descripción de las claves de acceso a las entidades, así como de los atributos característicos de las entidades definidas.
  4. Existe un catálogo de entidades.
  5. Son especificadas las relaciones entre las entidades y los almacenes.

Figura 2. Ejemplo de Objetivos de Control

4.- FASES.

Como ya se indicó anteriormente, se auditan los trabajos presentados en el año 1998/99 para la asignatura de Ingeniería del Software de 4° de la Escuela Técnica Superior de Ingenieros Industriales e Informáticos de Gijón (Universidad de Oviedo). Estos trabajos o proyectos son realizados por equipos de desarrollo formados por grupos de alumnos de la asignatura. Concretamente se trata de 16 equipos, formados por grupos de entre 5 y 8 alumnos, los cuales desarrollaron un proyecto por cada equipo. Los usuarios de dichos trabajos son alumnos de Ingeniería del Software II de 5° de la misma escuela, quienes propusieron dichos proyectos. Todos ellos fueron revisados por equipos de calidad compuestos también por alumnos de esta misma asignatura. Los directores de los desarrollos son los profesores responsables de la asignatura.

Los equipos de desarrollo deben realizar al menos dos versiones para las etapas de Análisis de Requisitos del Sistema y de Especificación Funcional del Sistema dentro de la fase de Análisis. Para la fase de Diseño deben entregar al menos una versión de los 4 documentos asociados al Diseño de los Componentes del Sistema y para finalizar un documento de Integración. Las pruebas de Integración son realizadas por los equipos de calidad. Entre cada una de las versiones entregadas se realizan los cambios necesarios que los equipos de garantía de calidad estiman oportunos. Al final del curso, deben entregar todo el desarrollo del proyecto integrado con todos lo cambios realizados y actualizados. Todo ello está planificado de antemano en el Manual de Procedimientos que deben seguir para la realización de los trabajos.

a) Estudio inicial

El alcance de este estudio es auditar todas las fases del ciclo de vida de los proyectos software desarrollados. Esto implica, por lo tanto, la auditoría del Análisis, Diseño, Construcción e Implantación de los distintos sistemas. También son auditados los documentos entregados en cada fase en cuanto a su forma y aspecto. Las pautas a seguir a este respecto, así como las distintas entregas en formato electrónico y las fechas de entrega, se encuentran debidamente contempladas y documentadas en el Manual de Procedimientos.

Los objetivos fundamentales para la realización de la presente auditoría son:

- La comprensión por parte de los alumnos de las metodologías existentes en la actualidad y la obtención de la práctica suficiente en el campo del desarrollo software y de proyectos.

- En segundo lugar, comprobar la corrección de las normas establecidas por los responsables de la asignatura de Ingeniería del Software I en la Escuela así como la estructura fijada de la asignatura durante el curso y la adecuación del Manual de Procedimientos seguido.

b) Actividad del auditor

Describimos en este apartado las acciones propias de la auditoría, en donde se especifican las técnicas concretas que han sido utilizadas y las herramientas de que nos servimos. Como ya mencionamos previamente, la técnica utilizada son la definición de los objetivos de control necesarios y suficientes para abarcar todo el ciclo de vida de los proyectos. El trabajo de campo consiste en analizar el cumplimiento íntegro de todos los objetivos de control definidos sobre las aplicaciones estudiadas, proporcionándonos una serie de indicadores sobre lo bueno o lo malo que la docencia suministra acerca de la Ingeniería del Software, además de identificar posibles carencias en los temas tratados en los cursos de 4° y 5° de la Escuela.

c) Redacción de informes

La función auditora se materializa con la entrega de los informes finales correspondientes, exclusivamente por escrito y avalando personalmente el juicio del auditor de forma documental. Esta es la única referencia constatable de toda la auditoría y el exponente de su calidad. En este informe se pone de manifiesto la situación actual, las tendencias, los puntos débiles y amenazas encontradas, las recomendaciones y planes de acción correspondientes. Es importante destacar, que el auditor sólo debe incluir los hechos importantes y consolidados. El destinatario final del informe es el cliente promotor de la auditoría, que en este caso es Isabel Sevilla, coautora de este estudio.

5.- RESULTADOS

Los resultados obtenidos en la propia auditoría son muy satisfactorios para el alumnado en cuanto al desarrollo software de proyectos informáticos. Los alumnos adquieren un alto conocimiento de la metodología Métrica versión 2.1 así como una gran experiencia en el desarrollo de aplicaciones y proyectos software. Han sido comprobados un total de 278 Objetivos de Control de los cuales el 67,96 % se han cumplido, el 10,86 % No se han cumplido y el 21,18 % no han podido ser aplicados debido a no estar contemplados en el Manual de Procedimientos seguido.

Como se puede observar en la Tabla 1, los resultados para la fase de Análisis son bastante buenos, mientras que para las fases de Diseño e Integración los resultados no son tan satisfactorios, debido al alto número de objetivos de control No Aplicables.

	Objetivos de Control Estudiados	Porcentaje de Objetivos Cumplidos	Porcentaje de Objetivos No Cumplidos	Porcentaje de Objetivos No Aplicables
ARS	55	87,27 %	12,73 %	0 %
EFS	65	91,15 %	8,85 %	0 %
DTS	46	54,08 %	4,62 %	41,30 %
DCS	43	69,04 %	17,01 %	13,95 %
DPU	30	51,25 %	15,42 %	33,33 %
PIA	39	54,97 %	6,57 %	38,46 %

Tabla 1. Resumen de los resultados obtenidos para cada fase del desarrollo

Este buen resultado seguramente es fruto del entorno en el que se desarrollan dichos proyectos, ya que por una parte son aleccionados en las clases de teoría por los responsables de la asignatura de Ingeniería del Software, y por otra parte, son guiados y evaluados por los directores de proyecto, y como complemento, aprenden de sus propios errores gracias a las revisiones de los equipos de garantía de calidad. Todo ello se puede decir que forma una estructura bastante bien definida para garantizar un alto grado de aprendizaje en las asignaturas de Ingeniería del Software.

Otro aspecto relevante encontrado es la buena aceptación por parte de los alumnos en lo que se refiere al seguimiento del Manual de Procedimientos existente para la realización de los trabajos. Aunque inicialmente les resulta algo confuso en cuanto a su estructura y contenido, una vez que se acostumbran a su uso y lo utilizan más frecuentemente, se convierte en una herramienta muy potente, que sirve como guía y apoyo para sus desarrollos. De esta forma, los alumnos se han acostumbrado perfectamente a la realización de proyectos reales con una planificación inicial bastante estricta en

cuanto a plazos de entrega, a documentos realizados y entregados, y a las distintas versiones realizadas para cada fase del desarrollo.

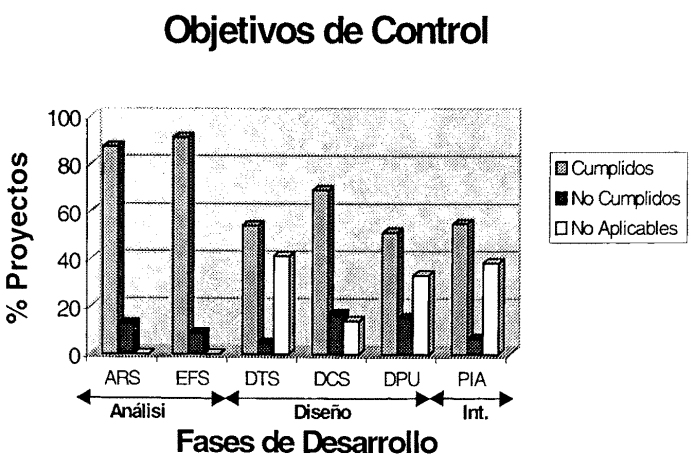


Figura 3. Resumen de Objetivos de Control estudiados para cada fase

A continuación se presenta un gráfico resumen, Figura 3, de la media de objetivos de control alcanzados satisfactoriamente para cada una de las fases del ciclo de vida del desarrollo de los 16 proyectos, tomando este valor como el 100%.

Como se puede observar en la Figura 3, los resultados obtenidos por los alumnos en la fase de Análisis son muy satisfactorios. En cuanto a la fase de Diseño e Integración, los resultados no son tan favorables. Además se puede comprobar que el Manual de Procedimientos se aleja levemente de la metodología Métrica en la fase de Diseño e Integración, como se puede observar por los objetivos de control con resultado "No Aplicable". Esto supone un riesgo evidente en el desarrollo de estos proyectos para dichas fases. Un ejemplo de la carencia de algunos aspectos importantes son los riesgos que provienen de no realizar un completo análisis del Modelo Físico de Datos o del Entorno Tecnológico de desarrollo, pudiendo provocar graves inconsistencias para el nuevo sistema.

Se debería mejorar realizando una actualización del Manual de Procedimientos, de forma que se incluyan estos apartados importantes en las fases de Diseño e Integración para asegurar el correcto funcionamiento del producto final.

6.- CONCLUSIONES.

Una vez concluidas las fases descritas anteriormente, podemos recoger e interpretar los resultados obtenidos tras la realización del trabajo de campo al aplicar el conjunto de objetivos de control a los proyectos estudiados de los alumnos.

Una primera reflexión personal que se extrae de la realización de la auditoría es la dificultad que conlleva esta actividad en cuanto a la obtención de resultados lo más eficientes posibles.

Un reto inicial es la realización de buenos cuestionarios de control, que incluyan todos los posibles casos y aspectos y que sean lo más completos posibles sin redundar en el estudio realizado. Este punto es muy importante porque de él dependen en gran medida los resultados que se puedan extraer, repercutiendo a su vez en las decisiones que se puedan tomar. Por ello, la fase de definición de objetivos de control y del trabajo de campo de cada una de las fases del desarrollo de proyectos, se ha tenido que realizar casi en paralelo ya que siempre aparecen nuevos objetivos a ser auditados, lo que se traduce en una revisión constante y refinamiento continuo de los cuestionarios.

Otra dificultad supone la propia interpretación que se dé a los distintos puntos que componen los proyectos estudiados a la hora de evaluar el conjunto de controles definidos. Es muy importante que el auditor tenga un alto grado de conocimiento general y de experiencia en el campo del desarrollo software.

7.- REFERENCIAS

[PALA1990] Palao, Manuel, *Objetivos de Control*, Ed. The EDP Auditors Foundation. Illinois, 1990

[ACHA1994] Acha Iturmendi, J.J., *Auditoría Informática en la empresa*, Ed. Paraninfo, 1994

[MAP1995] Ministerio para las Administraciones Públicas, Metodología de planificación y desarrollo de sistemas de información MÉTRICA,Versión 2.1, Ed. Tecnos, 1995.

[TUYA1999] Tuya, Javier, "Manual de Procedimientos para las prácticas de Ingeniería del Software I y II", http://opalo.etsiig.uniovi.es/~tuya/is/procedimientos/index.html, (accedido el 30-05-2000)

[ISAC2000] ISACA, The Information Systems Audit and Control Association & Foundation, http://www.isaca.org (accedido el 30-05-2000)

EBD-ULL: UNA HERRAMIENTA PARA LA EVALUACION DEL RENDIMIENTO DE SENTENCIAS SQL SOBRE DIFERENTES SISTEMAS GESTORES DE BASES DE DATOS

Lucio Cruz, Pedro González, Virginia Gutiérrez,
Jesús Alberto González, Daniel G. Morales, José Luis Roda

Centro Superior de Informática
Universidad de La Laguna
La Laguna, Santa Cruz de Tenerife
e-mail: **vgutier@ull.es**

RESUMEN: La necesidad de evaluar el rendimiento de las sentencias SQL sobre diferentes gestores de bases de datos reales nos ha motivado a realizar una herramienta en la que los alumnos puedan comprobar empíricamente la teoría de la optimización de bases de datos. Las consultas, actualizaciones (inserciones, borrados,...) y demás instrucciones que los alumnos realizan en los problemas de clase, son facilitadas a la herramienta para comprobar su rendimiento. Las asignaturas de Bases de Datos y de Ingeniería del Software ofrecen al alumno la base conceptual necesaria para su utilización. La herramienta, desarrollada bajo la plataforma Java, permite evaluar no sólo diferentes sistemas gestores de bases de datos sino también diferentes arquitecturas.

1.- INTRODUCCIÓN

El Centro Superior de Informática (CSI) de la Universidad de La Laguna imparte los dos primeros ciclos de informática (Ingeniero Técnico en Informática de Gestión y de Sistemas), y el segundo ciclo de Ingeniería en Informática (www.csi.ull.es). En el ciclo superior existen varias asignaturas relacionadas entre si como son las *Bases de Datos* y las *Ingenierías del Software*. En el trabajo [Bri99], se presentan los contenidos de las asignaturas de Ingeniería del Software del CSI, y se detalla la metodología de trabajo en las colaboraciones con las empresas. En todos los proyectos que se realizan bajo convenio con las empresas, los alumnos desarrollan los diseños de las bases de datos. Es en este punto donde los alumnos aplican los conocimientos adquiridos en *Bases de Datos*, normalizando y optimizando las diferentes consultas que se realizan.

Diferentes sistemas gestores de bases de datos ofrecen funcionalidades diversas para la optimización y evaluación de sentencias SQL. Servidores como Oracle Database Server [ORA], DB2 [IBM] y Microsoft SQL Server [MIC] conforman actualmente las plataformas más extendidas. Todas ellas

implementan módulos para optimizar y evaluar rendimientos, pero en todos los casos, como solución propietaria. La herramienta debe permitir evaluar el rendimiento de sentencias *SQL* antes y después de su optimización. Además, debería permitirnos escoger entre cualquier gestor de bases de datos que deseemos evaluar. Su desarrollo bajo la plataforma Java y haciendo uso de los controladores JDBC, nos permitirá evaluar no sólo sobre diferentes sistemas gestores de bases de datos sino también sobre diferentes arquitecturas. Esta herramienta ha sido desarrollada de forma conjunta por alumnos de las asignaturas de *Ingeniería del Software* y de *Bases de Datos*, y dirigida por los profesores de ambas asignaturas.

En el siguiente apartado se presenta el programa de la asignatura de Bases de Datos. En el tercer apartado presentamos la herramienta de evaluación que hemos desarrollado y en el cuarto apartado ofrecemos las conclusiones de este trabajo.

2.- LA ASIGNATURA DE BASES DE DATOS

La asignatura de Bases de Datos es optativa de 6 créditos (3 teóricos y 3 prácticos) que se imparte en el cuarto semestre de la Ingeniería en Informática.

a) Objetivos.

Los objetivos docentes de la asignatura se agrupan en cuatro bloques que detallaremos a continuación:

- Proporcionar al alumno la formación necesaria para tener una visión amplia de los diferentes sistemas gestores de bases de datos, modelos de datos, lenguajes de consultas, así como diversas técnicas de diseño de bases de datos.
- Adquirir conocimientos y destreza en la administración de un Sistema Gestor de Bases de Datos
- Obtener una visión amplia de las nuevas tendencias de las Bases de Datos
- Conocer algunas de las aplicaciones de las Bases de Datos

b) Programa de la asignatura.

El programa de la asignatura contempla las clases teóricas, las clases prácticas y la realización de trabajos.

Los contenidos teóricos están estructurados en 5 temas. En el primer tema se introducen y repasan conceptos fundamentales de Bases de Datos. El segundo tema presenta al alumno los aspectos de diseño, las dependencias funcionales, las formas normales, algoritmos de esquemas y las dependencias mutivaluadas. El tercer tema ofrece las técnicas de implementación de sistemas de bases de datos: catálogos, optimización de consultas y procesamiento de transacciones. Aspectos de seguridad de las bases de datos, como control de concurrencia, control de acceso y Criptografía, se imparten en el cuarto tema. En el último tema, ofrecemos al alumno nuevas tendencias en los modelos de datos: bases de datos orientadas a objetos, bases de datos paralelas, bases de datos basadas en la lógica, o sistemas de bases de datos geográficas.

Las prácticas de la asignatura se dividen en tres grandes bloques. Uno orientado al sistema gestor de bases de datos ORACLE: vemos como administrar un sistema gestor de bases de datos relacionales,

utilizando para ello los comandos SQL*Plus propios del Oracle, en conjunción con sentencias SQL y el lenguaje procedimental PL/SQL, aplicándolos en diversas consultas SQL. En el otro bloque de prácticas se ven las diferentes técnicas de diseño de una base de datos relacional, utilizando la herramienta EasyCase para obtener un diseño conceptual de la base de datos y Reno para conseguir un esquema relacional normalizado. En el último bloque, planteamos optimizar consultas SQL, obteniendo para ello el árbol de consulta canónico más eficiente guiado por reglas heurísticas de optimización. Consideramos, por ejemplo, la siguiente consulta:

"nombres de los profesores, no doctores, que imparten docencia en una asignatura titulada Informática", para la cual una posible formulación en SQL es:

```
Select Nombre, Apellidos
From PROFESORES, ASIGNATURAS, PLAN_DOCENTE
where Título="Informática" and NRP_Profesor=NRP and Codigo_asig=Codigo and
Grado="Lcdo";
```

Nos damos cuenta la necesidad de optimizarla. Para ello traducimos la consulta a una representación interna, la convertimos en forma canónica, elegimos los procedimientos candidatos de bajo nivel (teniendo en cuenta la existencia de índices, el agrupamiento físico de los registros,..), y, por último, generamos planes de consultas y escogemos el más eficiente y menos costosa.

En referencia a la realización de trabajos, los alumnos deberán exponer un trabajo donde se compararán diferentes sistemas gestores de base de datos, modelos de datos, así como una aplicación para la optimización de consultas, además de entregar un trabajo práctico consistente en el diseño e implementación de una base de datos, optimizando diversas consultas y utilizando las herramientas vistas en prácticas.

c) Bibliografía.

La bibliografía más relevante es la que se adjunta a continuación. Además los alumnos para los trabajos que realizan, tienen en Internet una fuente inagotable.

El libro "*Sistemas de Bases de Datos. Conceptos fundamentales*" de los autores Elmasri y Navathe, constituye el texto principal de la asignatura. Los libros de Korth [Kor93] y de Date [Dat93] aportan al alumnado una visión más sencilla de los sistemas de bases de datos. El libro de Bertino y Martino [Ber97] ofrece contenidos del tema de Sistemas de Bases de Datos Orientadas a Objetos.

Las guías de Oracle [Abb95a], [Abb95b] y las de SQL [ORA1], [ORA2] son fundamentales para el desarrollo del primer bloque de prácticas, siendo necesaria su utilización para una correcta administración del sistema gestor de bases de datos Oracle.

3.- LA HERRAMIENTA *EBD-ULL*

La herramienta permitirá a los alumnos contrastar los rendimientos de las sentencias SQL antes y después de realizar las correspondientes optimizaciones.

a) Objetivos.

- Ofrecer al alumno una herramienta visual de evaluación de sentencias SQL donde poner en práctica los conocimientos teóricos.
- Comprobar con la herramienta que las optimizaciones realizadas en los ejercicios de clase son realmente efectivas.
- Ofrecer un entorno común para la evaluación de sentencias sobre diversas bases de datos. Permite guardar las sentencias en una tabla de sentencias ejecutadas para su posterior recuperación sobre otras bases de datos o para realizar nuevas sentencias.
- Comprender los diferentes controladores Java existentes para el acceso a las bases de datos: JDBC:ODBC, JDBC clase 1, 2, 3 y 4.
- Comparar sus rendimientos de los diferentes controladores.

b) Estructura de la herramienta.

EBD-ULL ha sido realizada en Java con la herramienta de desarrollo de aplicaciones VisualAge de IBM. Se ha dividido en tres partes. La primera parte permite la elección del tipo de controlador JDBC a utilizar y crear la conexión con la base de datos. La Figura 1 muestra la pantalla para la realización de las conexiones con la base de datos.

La segunda parte, ofrece un conjunto de características relacionadas con las sentencias que se van a diseñar o recuperar de la base de datos de consultas. Una vez diseñada la consulta, la herramienta nos permite su almacenamiento sobre una tabla interna. La Figura 2 presenta la pantalla correspondiente a estas funciones.

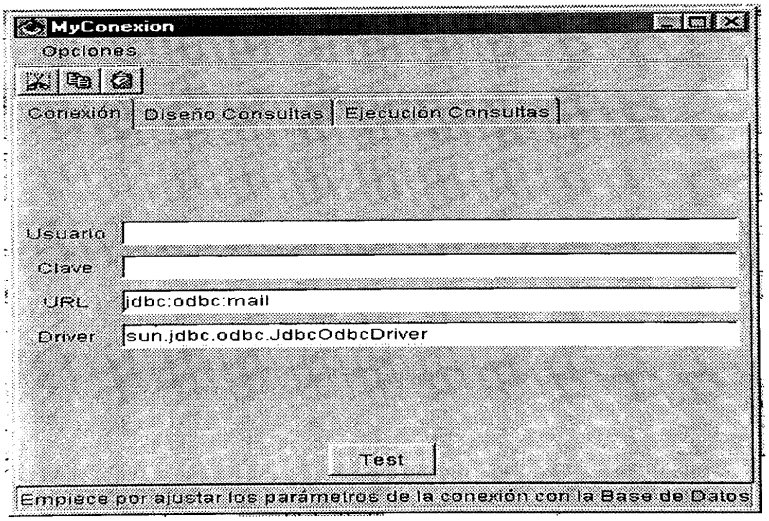


Figura 1: Selección de la conexión y del tipo de controlador.

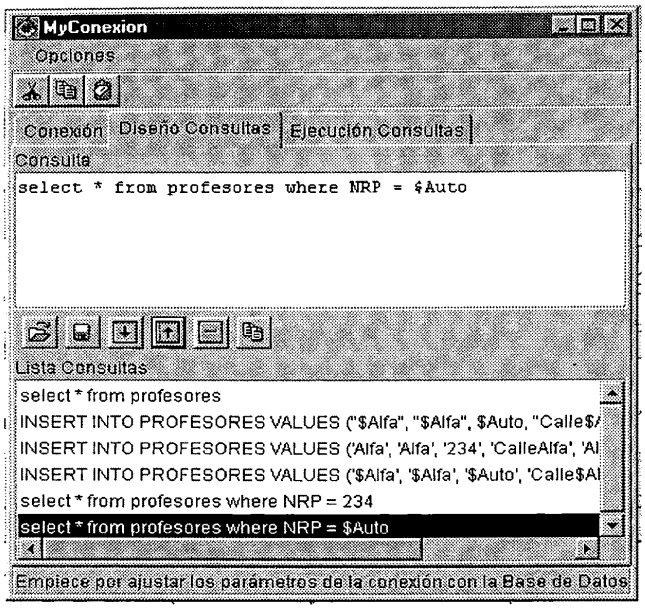


Figura 2: Diseño, almacenamiento o recuperación de la sentencia.

La tercera parte ofrece los resultados de la ejecución de la sentencia diseñada. Se ofrecen los resultados de la sentencia y los tiempos obtenidos: máximo, mínimo y media de un conjunto de repeticiones que podemos seleccionar. La Figura 3 ofrece la pantalla del módulo de los resultados.

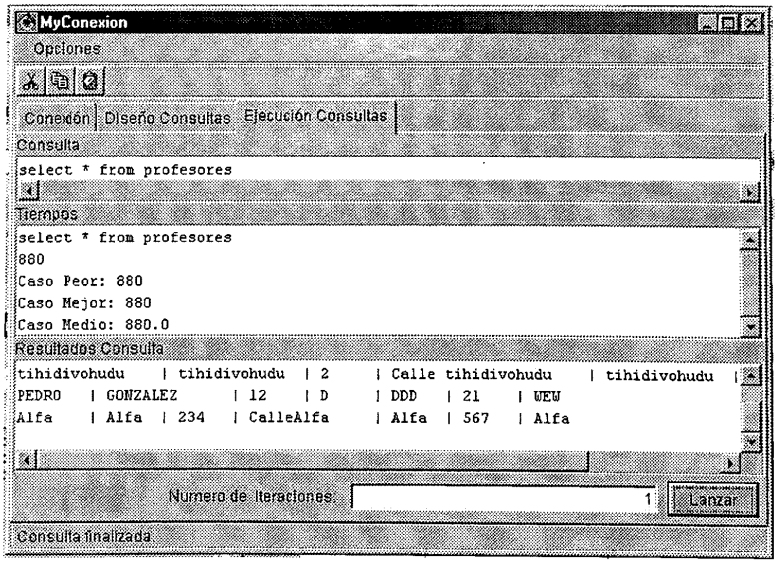


Figura 3: Resultados de la ejecución y tiempos obtenidos.

4.- CONCLUSIONES

- Hemos desarrollado una herramienta de evaluación de sentencias SQL independiente del gestor de bases de datos y de la arquitectura subyacente.
- Ofrece al alumno la posibilidad de contrastar los conocimientos teóricos sobre como optimizar las sentencias SQL.
- Permite al alumno realizar ejecuciones de sentencias SQL sobre diferentes bases de datos.
- Permite evaluar el rendimiento de los diferentes controladores JDBC existentes.

REFERENCIAS

Bri99 J. Brito, J.A. González, D.G. Morales, J.L. Roda. Ingeniería del Software en la Universidad de La Laguna: Colaboraciones con las Empresas. Jornadas sobre la Enseñanza Universitaria de la Informática. La Almunia. Zaragoza. Octubre 1999.

[Elm97] R. Elmasri & S. Navathe. Sistemas de Bases de Datos. Conceptos fundamentales. Addison-Wesley, 1997.

[Kor93] H.F. Korth & A. Silberschatz. Fundamentos de Bases de Datos. Mc-Graw Hill, 1993.

[Dat93] C. J. Date. Introducción a los Sistemas de bases de datos. Volumen 1. Addison-Wesley. Iberoamericana, 1993

[Ber97] E. Bertino & L. Martino. Sistemas de Bases de Datos Orientadas a Objetos. Conceptos y Arquitecturas. Addison-Wesley & Diaz de Santos, 1997.

[Abb95a] M. Abbey & M.J. Corey. Oracle. Guía de aprendizaje. Mc-Graw Hill, 1995.

[Abb95b] M. Abbey & M.J. Corey. Oracle. Manual del Administrador. Mc-Graw Hill, 1995.

[ORA1] SQL*Plus. User Guide and References. Versión 3.1. ORACLE, 1996.

[ORA2] PL/SQL. User Guide and References. Versión 2.0. ORACLE, 1996.

[IBM] www.ibm.com

[MIC] www.microsoft.com

[ORA] www.oracle.com

[SUN] www.sun.com

UN MÉTODO DOCENTE PARA LA REALIZACIÓN DE PRÁCTICAS DE INGENIERÍA DEL SOFTWARE

Manuel Mejías, Isabel Ramos, Octavio Martín, María J. Escalona

Dpto. Lenguajes y Sistemas Informáticos. Universidad de Sevilla
e-mail: risoto@lsi.us.es

RESUMEN: En este artículo se plantea la importancia que tienen los trabajos prácticos en las asignaturas relacionadas con la Ingeniería del Software, se analizan las circunstancias en las que las mismas se desenvuelven y se propone un método para llevarlas a cabo.

1.- INTRODUCCIÓN.

La Ingeniería del Software constituye un área de interés fundamental en todos los estudios de informática. Su importancia radica en que es en esta materia donde se integran y se aplican todos los conocimientos que los alumnos van adquiriendo con el estudio del resto de materias incluidas en los planes de estudios de las titulaciones de informática.

No obstante su importancia, por regla general la consideración de la ingeniería del software en los planes de estudio resulta insuficiente para abordar todos los aspectos que una formación completa de los alumnos debe contemplar. Esto se ve acentuado al analizar el tiempo dedicado a la aplicación práctica de los conceptos por parte de los alumnos.

Nuestra experiencia docente en el campo de la Ingeniería del Software nos dice que el inicio de los alumnos en esta materia es complejo y les supone un elevado esfuerzo de adaptación de mentalidad a la hora de enfrentarse a la resolución de problemas de complejidad no tan limitada como los que se encuentran desde que empiezan sus estudios de informática.

Esta dificultad se debe a nuestro entender a dos circunstancias complementarias: 1) el hacer frente a la resolución de problemas más generales, pasando de resolver problemas concretos que tienen una funcionalidad limitada, a la resolución de problemas que precisan una descomposición importante de su funcionalidad para obtener problemas más pequeños que puedan ser abordados de una manera realista y adecuada; y 2) el enfrentarse a la resolución de los problemas de una forma global desde que se concibe la idea de solucionarlo hasta que se obtiene un producto concreto y su mantenimiento, pasando por todos los estadios propios de las fases de producción de un sistema software.

La forma de abordar esta situación es poner a disposición de los alumnos los medios y los entornos formativos adecuados para ayudarles a realizar la transición en la forma de pensar. Es en este punto donde podemos encuadrar el trabajo objeto de este artículo

2.- ENTORNO EDUCATIVO.

Antes de presentar de forma concreta el objeto del trabajo conviene indicar el entorno en el que el mismo se sitúa.

Nuestro objeto de estudio se centra en una asignatura de Ingeniería del Software ubicada en el segundo cuatrimestre del segundo curso de la titulación de Ingeniería Técnica de Gestión, y más concretamente en la realización de las prácticas de dicha asignatura.

Se trata de una asignatura troncal de seis créditos. Es la primera asignatura que cursan los alumnos en relación con la ingeniería del software, existiendo a continuación dos asignaturas más dedicadas a esta materia. El objetivo de esta asignatura es introducir a los alumnos en los conceptos de la Ingeniería del Software y en el estudio detallado de la Fase de Análisis en la construcción de los sistemas software, siguiendo el enfoque estructurado [MAP 1995a] [MAP 1995b], utilizando los diagramas de casos de uso como técnica de ayuda para la recolección de requisitos [Durán 1999].

Tal como nos dice la experiencia, el aspecto que supone mayor esfuerzo para los alumnos es aplicar los conceptos estudiados a la resolución concreta de problemas, y es en este sentido donde estamos haciendo un mayor esfuerzo para encontrar la mejor solución para los alumnos.

3.- PLANTEAMIENTO DE LAS PRÁCTICAS.

El planteamiento que vamos a presentar es el resultado de varios años de experiencia docente y sería aplicable, con adaptaciones, a cualquier asignatura relacionada con la Ingeniería del Software, e incluso a algunas otras que no estuvieran directamente relacionadas con la materia.

Nuestro propio convencimiento, y las circunstancias que el entorno impone, las cuales se concretan en el carácter cuatrimestral de la asignatura, un elevado número de alumnos, insuficiente número de profesores y limitada dedicación por parte de los alumnos debido al elevado número de asignaturas que deben cursar simultáneamente, nos llevan a plantear la realización de las prácticas en grupo, lo cual entendemos que es muy beneficioso puesto que además de permitirnos la realización del trabajo práctico en el contexto anteriormente indicado, permite a los alumnos el desarrollar su capacidad de trabajo en equipo.

De una forma global, la metodología para la realización de este trabajo práctico en equipo se concreta en los pasos indicados en la figura 1.

Paso 1. Constitución del equipo de trabajo y propuesta, por parte de éste, del tema a desarrollar.

En este punto los alumnos se agrupan formando equipos de trabajo de cinco miembros cada uno de ellos. Cada equipo rellena una ficha de formación del grupo indicando los componentes del mismo y proponiendo un sistema software para el que hay que realizar la etapa de análisis.

Paso 2. Organización de los grupos de trabajo y asignación de profesor tutor.

Una vez que todos los grupos han entregado la ficha correspondiente, el profesor responsable de la asignatura organizará los grupos de trabajo asignando a cada grupo un número de identificación y el profesor tutor encargado de cada uno de ellos.

Paso 3. Evaluación de las propuestas de trabajo por el profesor tutor.

En este momento cada profesor de la asignatura evaluará las propuestas de trabajo realizadas por los diferentes grupos que ha de tutelar, con el objetivo de adaptar, cuando sea necesario, las diferentes propuestas a las exigencias planteadas con la realización de la práctica para conseguir los objetivos académicos que con ella se han planteado.

Paso 4. Confirmación del trabajo a realizar.

Una vez evaluadas las propuestas por los profesores tutores, el resultado de esa evaluación será comunicado a los diferentes grupos para que estos puedan desarrollar su trabajo.

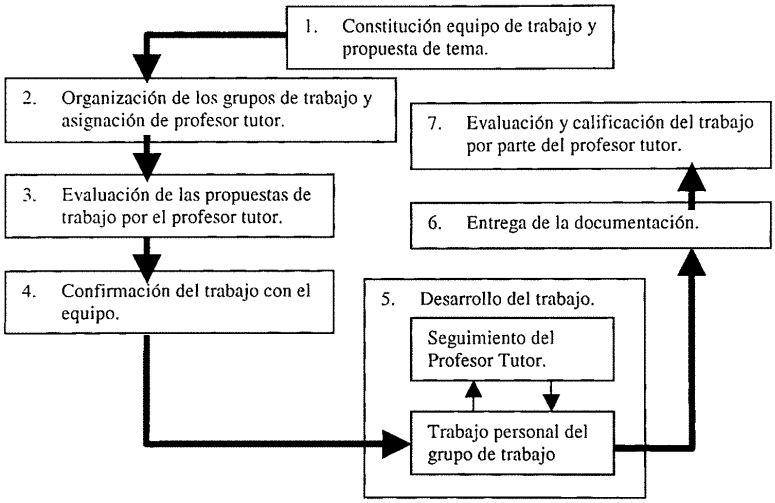


Figura 1: Proceso de desarrollo de prácticas en Ingeniería del Software

Paso 5. Desarrollo del trabajo.

Esta es la etapa ejecutiva del trabajo práctico, en ella cada equipo de trabajo realiza el análisis de requisitos y la especificación funcional del sistema software que el equipo ha propuesto.

En esta etapa se produce un seguimiento del trabajo desarrollado por el equipo por parte del profesor tutor, quien le asesorará ante las dudas que se les presente y les orientará en el desarrollo del trabajo, lo cual se realizará mediante sesiones de tutoría planificadas con el profesor. Debido a las circunstancias actuales en que se están desarrollando estas prácticas, estas sesiones de tutoría se realizan a iniciativa del equipo de trabajo y tienen por contenido el que resulte de interés en cada momento para cada equipo.

Paso 6. Entrega de la documentación.

Una vez desarrollado el trabajo, el equipo entregará la documentación elaborada para el análisis del sistema software al profesor tutor correspondiente, en la fecha indicada para ello previamente.

Paso 7. Evaluación y calificación del trabajo por parte del profesor tutor.

Con la documentación entregada por los equipos de trabajo, los profesores tutores realizarán la evaluación y calificación del trabajo práctico, considerándose para la misma criterios como adecuación a las normas de documentación de la práctica (máximo 1.5 puntos), aplicación de las técnicas estudiadas en clase (máximo 2.5 puntos), número y calidad de asistencias a las sesiones de tutoría (máximo 1.5 puntos), complejidad del sistema elegido (máximo 1.5 puntos), volumen de trabajo (máximo 1 punto) y calidad de la presentación de la documentación (máximo 2 puntos). Con estos criterios se asigna una calificación numérica global al trabajo práctico que supondrá el treinta por ciento de la calificación global de la asignatura.

4.- DOCUMENTACIÓN A ELABORAR EN EL TRABAJO PRÁCTICO.

Para realizar el trabajo práctico se hace entrega a los alumnos de un documento de Normas desarrollado por los profesores de la asignatura. En dicho documento [LSI 2000], entre otros aspectos, se describe la estructura y el contenido de los documentos a elaborar cuando se realice el análisis estructurado del sistema propuesto por cada grupo de trabajo.

Los dos documentos a generar durante la realización de la práctica son:

I. Documento de Requisitos del Sistema (DRS)

La estructura de este documento consta de los apartados indicados en la figura 2. La parte fundamental de este documento la constituye la definición del catálogo de requisitos que debe satisfacer el sistema a desarrollar. Estos requisitos son de tres tipos: a) de almacenamiento de información, que se expresarán de forma tabular; b) requisitos funcionales que se expresarán mediante diagramas de casos de uso y descripción tabular de los casos de uso; y c) requisitos no funcionales del sistema que se expresarán también de forma tabular.

Portada
Lista de cambios
Índice (*paginado*)
Lista de figuras
Lista de tablas
1. Ámbito y alcance del proyecto
    1.1. Objetivos principales
    1.2. Departamentos implicados
    1.3. Composición del equipo de trabajo y usuarios participantes
    1.4. Restricciones
    1.5. Planificación inicial
2. Catálogo de Requisitos del Sistema
    2.1. Requisitos de almacenamiento de información
    2.2. Requisitos funcionales
        2.2.1. Diagramas de casos de uso
        2.2.2. Definición de actores
        2.2.3. Casos de uso del sistema
    2.3. Requisitos no funcionales
        2.3.1. Requisitos de seguridad
        2.3.2. Requisitos de rendimiento
        2.3.3. Otros requisitos
3. Matriz de rastreabilidad
4. Glosario de términos *(opcional)*

Apéndices *(opcional)*

Figura 2: Estructura del Documento de Requisitos del Sistema

II. Documento de Especificación Funcional (DEF)

En su estructura se distinguen los apartados indicados en la figura 3, siendo los puntos más significativos: a) la descripción del modelo de procesos del sistema según la notación de los diagramas de flujos de datos hasta el nivel funcional de especificación y utilizando representaciones tabulares para describir los distintos elementos que aparecen en dichos diagramas y el diccionario de datos; b) la definición del modelo de datos mediante un diagrama entidad/relación, así como la descripción tabular de las entidades de datos y de las relaciones entre entidades, y c) la representación del comportamiento dinámico del sistema, para lo que se definirá el catálogo de eventos, la matriz entidad/evento y el diagrama de estados que represente el comportamiento dinámico para aquellas entidades de datos que lo requieran.

Portada
Lista de cambios
Índice (*paginado*)
Lista de figuras
Lista de tablas
1. Modelo de procesos del sistema
   1.1. Definición del sistema
      1.1.1. Diagrama de contexto
      1.1.2. Descripción de entidades externas
   1.2. Definición de subsistemas
      1.2.1. Diagrama de flujos de datos
      1.2.2. Descripción de subsistemas
   1.3. Especificación de subsistemas
   1.4. Descripción de los almacenes de datos
   1.5. Descripción de los flujos de datos
   1.6. Diccionario de datos
2. Modelo de datos del sistema
   2.1. Diagramas de entidad/relación
   2.2. Descripción de entidades
   2.3. Descripción de interrelaciones
3. Modelo de eventos del sistema
   3.1. Catálogo de eventos
   3.2. Matriz entidad-evento
   3.3. Diagramas de estados
4. Interfaces de usuario
   4.1. Mapa de pantallas/ventanas
   4.2. Especificación de pantallas/ventanas
   4.3. Matriz procesos/pantallas
   4.4. Especificación de Informes del sistema
   4.5. Matriz procesos/informes
5. Glosario de términos *(opcional)*

Apéndices *(opcional)*

Figura 3: Estructura del Documento de Especificación Funcional.

FD-<*id*>	***<nombre flujo de dato>***	
Descripción	Este flujo de dato corresponde a <*concepto al que corresponde el flujo de dato*>	
Requisitos asociados	<*Requisitos asociados*>	
Composición	La información que circula por el flujo de dato [consta de los siguientes datosl se distribuye en los siguientes flujos de datos]:	
	Nombre	**Tipo**
	[<*nombre del dato*> | <*nombre del flujo de datos*>]	[*dato simple* | *dato complejo* | *flujo de datos*]
	...	...
Comentarios		

Figura 4: Plantilla de descripción de flujos de datos.

A lo largo de este apartado se ha hecho referencia al uso de una técnica tabular para describir diferentes elementos incluidos en la documentación generada. Como ejemplo significativo de la misma, en la figura 4 se incluye la plantilla de tabla utilizada para describir los flujos de datos. Las descripciones de todas las plantillas de tablas utilizadas se incluyen en la Norma para la práctica referida anteriormente [LSI 2000]

5.- CONCLUSIONES

Según lo indicado en este trabajo y como fruto de nuestra propia experiencia docente, se puede afirmar que el desarrollo de trabajos de aplicación práctica en asignaturas de Ingeniería del Software es muy importante, pero que las circunstancias en las que la mayoría de las veces la docencia de estas asignaturas se desarrolla no son las más idóneas, por lo que hay intentar buscar un método docente que permita alcanzar los mejores objetivos aún considerando las circunstancias actuales. En este trabajo presentamos una propuesta a este método docente que se está aplicando con resultados positivos en la docencia de diversas asignaturas de Ingeniería del Software en la Facultad de Informática y Estadística de la Universidad de Sevilla.

REFERENCIAS

[Durán 1999] A. Durán, B. Bernárdez. *Metodología para la Elicitación de Requisitos de Sistemas Software*. Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla. Sevilla, 1999.

[LSI 2000] Departamento de Lenguajes y Sistemas Informáticos. *Norma para Elaborar la Documentación de la Práctica de Ingeniería del Software de gestión I*. Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla. Sevilla, 2000.

[MAP 1995a] MAP. *Metodología de Planificación y Desarrollo de Sistemas de Información. MÉTRICA versión 2.1. Guía de Referencia*. TECNOS, 1995.

[MAP 1995b] MAP. *Metodología de Planificación y Desarrollo de Sistemas de Información. MÉTRICA versión 2.1. Guía de Técnicas*. TECNOS, 1995.

ENTORNOS DE DESARROLLO DE SOFTWARE: CONTENIDOS Y ENCUESTA DE SATISFACCIÓN[5]

José Angel Carsí

[1]*DSIC-EUI-UPV (Departamento de Sistemas Informáticos y Computación - Escuela Universitaria de Informática – Universidad Politécnica de Valencia)*
e-mail: pcarsi@dsic.upv.es

RESUMEN: En este artículo se presentan los contenidos de una asignatura en la que se pretende enseñar las características comunes a los entornos de desarrollo de software actuales. Los contenidos tienen un enfoque generalista con un importante contenido teórico pero sin dejar de lado un enfoque práctico de aplicación inmediata. Finalmente se muestran los resultados de una encuesta de satisfacción realizada a los alumnos que cursan la asignatura.

1.- INTRODUCCIÓN.

Existe un fuerte debate sobre cuáles deben ser los contenidos de los temarios de las asignaturas en la Universidad no sólo en España, sino también fuera [Jon95]. Las posiciones varían entre aquellos que abogan por una orientación básicamente teórica de los contenidos y aquellos que creen que los contenidos deben ser fundamentalmente prácticos. Ambas posturas tienen argumentos de peso:

- Los primeros argumentan que si se conocen los fundamentos teóricos de cualquier tecnología, luego la aplicación práctica de dichos conocimientos no requiere más que el estudio de la tecnología destino, siendo productivos en un tiempo mínimo.

- Los segundos dicen que la situación actual del mercado laboral reclama personas listas para ser productivas de manera inmediata. En el pasado, las empresas ofertaban cursos de especialización, en los que se enseñaba a los recién llegados los contenidos necesarios para la empresa. En la actualidad las PYMES no pueden ofertar dichos cursos de entrada exigiendo a los candidatos conocimientos que deben adquirir por su cuenta si desean ponerse a trabajar.

A la hora de elegir los contenidos de una asignatura, el responsable de la misma debe tomar partido en un sentido u otro. Aunque, posiblemente la mejor elección consista en una postura

[5] Este trabajo ha sido parcialmente soportado por la Escuela Universitaria de Informática de la Universidad Politécnica de Valencia.

intermedia entre la enseñanza de los contenidos teóricos necesarios para obtener una educación generalista y la aplicación práctica de dichos contenidos con las herramientas industriales que demande el mercado laboral.

En la Facultad de Informática de la Universidad Politécnica de Valencia se imparte desde hace dos años una asignatura (optativa del 10º cuatrimestre) cuyo nombre es "Entornos de Desarrollo de Software" con tres créditos de teoría y tres de prácticas. En dicha asignatura hubo que tomar la decisión de elaborar unos contenidos que fueran: atractivos para los alumnos, acordes con el mercado laboral y en los que se impartieran unos conocimientos que permitieran a los alumnos y futuros responsables de departamentos de informática en las empresas la toma de decisiones en su trabajo.

Una de las alternativas sobre temario en las que primero se pensó para una asignatura con dicho nombre, consistía en hacer una presentación de todos los entornos de desarrollo de software actuales con sus características, defectos y ejemplos. Rápidamente, fue desechada esta opción ante la imposibilidad de presentar dicha información de forma actualizada. No hay más que ojear las revistas técnicas del sector (Byte, PC World, PC Magazine, PC Actual, Sólo Programadores, Programación Actual, ... por citar algunas de las más comunes, disponibles en prácticamente cualquier quiosco) para darse cuenta de la enormidad de la tarea.

De manera que, la única alternativa posible consiste en analizar los entornos de desarrollo que existen actualmente, ver los puntos que tienen en común y centrarse en ellos. De esta forma, la enseñanza tiene un enfoque generalista, pero crítico. Para adquirir destreza en las tecnologías, en las prácticas se aplican los conocimientos adquiridos en las clases teóricas con un entorno de desarrollo concreto, pero utilizando el entorno como una herramienta y no como un objetivo.

A pesar de lo anterior, no hay que descartar la necesidad de estar al día en cuanto a los entornos que van saliendo, sobre las nuevas versiones, sus características, etc. Sobre todo teniendo en cuenta que en informática es tremendamente fácil quedar desfasado en unos años. Para evitarlo, la única solución es leer y continuar la formación durante toda la vida. Posiblemente, una de las mejores fuentes de información disponibles son las revistas mencionadas en el párrafo anterior (no sólo esas, hay bastantes más). Para intentar que los alumnos adquieran la costumbre de leer mensualmente alguna revista del sector se les exige que, durante el periodo que dura la docencia de la asignatura, todos los meses lean alguna y preparen un artículo como si fueran a presentarlo al resto de compañeros. Al final del cuatrimestre, se les da la oportunidad, a aquellos que lo deseen, de contar alguna de las presentaciones preparadas al resto de la clase.

Con respecto al temario de la asignatura, tras realizar un estudio sobre cuáles son las principales demandas del mercado [Jon95, Dou96, Tuc96, Gar97] y cuáles son las características que comparten los entornos de desarrollo actuales, finalmente se escogió el siguiente temario.

2.- CONTENIDOS TEÓRICOS.

A continuación se muestran los contenidos de la asignatura "Entornos de Desarrollo de Software" justificando su presencia en la asignatura.

Tema 1: Sistemas Abiertos. En un entorno tan competitivo, heterogéneo y voluble como el actual sólo con la correcta elección de los estándares se puede conseguir que las aplicaciones se acaben en los plazos previstos, puedan interoperar con otras y sean fácilmente ampliables. En este tema se presentan los siguientes contenidos:

Definición de sistema abierto. ¿Qué es un estándar?. Oficinas de estandarización. Sistemas abiertos frente a propietarios. El factor tiempo en la estandarización.

Bibliografía: [ANSI], [Benson], [Com94], [IEEE], [ISO]

Tema 2: Calidad. Relacionado con la competitividad del mercado, sólo aquellos productos de calidad tienen alguna posibilidad de triunfar. Calidad entendida como la adecuación del producto a las necesidades de los usuarios y al cumplimiento de determinados factores de calidad. En este tema se presentan los puntos siguientes: Qué es un producto de calidad. Qué es un sistema de calidad. Dos modelos de mejora de procesos: el CMM y el estándar ISO 9000-3.

Bibliografía: [Inc95], [Tob97]

Tema 3: Cliente-Servidor. Según encuestas realizadas [Dou96], este es uno de los dos tópicos más demandados por las empresas de informática (el otro es la orientación a objetos), siendo imprescindible en el curriculum de cualquier ingeniero de informática. En este tema se presenta: Justificación del modelo C/S. Coste y beneficios. Fundamentos de C/S. Características de las aplicaciones C/S. Distribución de componentes. Arquitectura de aplicaciones de 2 y 3 niveles. Ingeniería del software para C/S.

Bibliografía: [Com94], [Gar97], [Pre97]

Tema 4: Middleware. Para construir aplicaciones C/S multicapa es necesario el uso de software que facilite su construcción y se encargue de los problemas relacionados con la distribución. Dado el marco en el que se deben construir aplicaciones en la actualidad, la gran variedad de sistemas que deben interoperar y la necesidad de realizar las aplicaciones de manera que sean fácilmente portables. ese hace imprescindible el uso del *middleware*. En este tema se presenta: Qué es el *middleware*. *Frameworks* y dos ejemplos: Los monitores de transacciones y el *Lotus Notes*.

Bibliografía: [Ber96]

Tema 5: El World Wide Web en la construcción de aplicaciones. Si hay alguna tecnología que despunta hoy en día es el Web. No falta mucho para el día en que la mayoría de las aplicaciones que se desarrollen deban exponer su funcionalidad por el Web y prácticamente todos los entornos de desarrollo tienen extensiones para la construcción de aplicaciones que hagan uso de Internet. Los ámbitos de aplicación de las aplicaciones para Internet están relacionados con temas de educación, comercio electrónico, integración de los clientes y proveedores en los sistemas de información de las empresas, etc. En este tema se presentan las características del desarrollo de aplicaciones para Internet.

Bibliografía: [Pou97]

Tema 6: COM. Como ejemplo de uno de los modelos de componentes que se pueden utilizar en la construcción de aplicaciones multicapa que hagan uso del Web para ofrecer su funcionalidad, se cuenta con detalle el modelo de componentes de Microsoft. En este tema se presenta: Descripción general del modelo. Cómo definir la funcionalidad de los componentes: Interfaces. Automatización OLE. Bibliotecas de tipos. La plataforma Activa de Microsoft.

Bibliografía: [Rog97], [Moc98]

Por supuesto en un solo cuatrimestre, no se puede ver con profundidad todos los temas reflejados en los contenidos, sino se hubieran introducido previamente en otras asignaturas. En alguno de los temas (los iniciales), se hacen continuas referencias a los contenidos de otras asignaturas en la que se presentaron, relacionando unos con otros y orientando a los alumnos en caso de que necesiten más información.

2.- CONTENIDOS PRÁCTICOS.

El objetivo de las prácticas es que los alumnos adquieran destreza en los contenidos teóricos presentados. Como ejemplo de entorno de desarrollo actual se utiliza Delphi 4.0. Las razones que nos impulsaron a escoger Delphi, de entre los diferentes entornos de producción de software fueron:

1. Delphi permite el desarrollo rápido de aplicaciones[6] y proporciona un entorno visual cómodo y sencillo para la construcción de aplicaciones de uso general que puede ser útil en la vida profesional de los alumnos en el futuro.

2. El lenguaje de programación de Delphi (*Object-Pascal*), es un Pascal extendido para incorporar los conceptos objetuales que creemos que es sencillo y didáctico.

3. Soporta las últimas tecnologías (Web, COM, CORBA, ...), de manera que se puede hacer cualquier tipo de prácticas con él.

Cuando los alumnos llegan a está asignatura no conocen nada sobre Delphi, aunque sí tienen conocimientos sobre análisis y diseño OO porque han sido presentados en otras asignaturas. Es por ello, que las primeras prácticas se dedican a la presentación del entorno: 1ª Práctica: Introducción a Delphi. , 2ª Práctica: El Modelo de Objetos de Delphi, 3ª Práctica: Bases de Datos en Delphi.

Creemos que en el futuro los alumnos llegarán a esta asignatura con mayores conocimientos sobre Delphi ya que las asignaturas que se imparten en cursos anteriores dentro de la unidad docente están siendo sometidas a un proceso de revisión y van a introducir Delphi en diversas asignaturas, con lo que las primeras prácticas se podrán reducir o incluso eliminar para centrarnos en lo que es el objetivo de las prácticas.

4ª Práctica: Construcción de Aplicaciones para Internet. En esta práctica se presenta como construir aplicaciones que exporten su funcionalidad a través del *World Wide Web* utilizando los navegadores como interfaces gráficas de las aplicaciones.

5ª Práctica: Práctica Final. Una vez presentados todos los conceptos necesarios se les pide que construyan una aplicación multicapa en la que se utilicen los navegadores como interfaz gráfica de la aplicación, construyan servidores de aplicaciones con

[6] *RAD (Rapid Application Development)*

Delphi que ofrezcan la funcionalidad de la aplicación y accedan a diversas bases de datos para almacenar la información.

Con el objetivo de construir un espíritu crítico en los alumnos con respecto a lo que realicen, se les exige que justifiquen el diseño de la aplicación a partir de los contenidos vistos en clase: partes de la aplicación, nº de capas, posibilidad de redistribución de funciones, justificación de la interfaz gráfica. Además, se debe acompañar la aplicación del manual de usuario de la aplicación. Siendo está parte, una de las que más puntúa a la hora de dar una nota a las prácticas.

3.- CRITERIOS DE EVALUACIÓN.

Al tener la asignatura que se presenta un contenido práctico importante (3 créditos) la nota de los alumnos se obtiene en un 50% a través de un examen teórico y en el otro 50% a través de la práctica final desarrollada. Siendo necesario para calcular la nota tener aprobado tanto el examen teórico como la práctica a desarrollar y haber preparado tres presentaciones de tres artículos, relevantes según los contenidos de la asignatura, escogidos entre las revistas del sector. Si finalmente los alumnos se deciden a contar al resto de compañeros alguna de las presentaciones realizadas, la nota se incrementa para tener en cuenta el esfuerzo que supone la presentación en público.

4.- ENCUESTA DE SATISFACCIÓN.

La asignatura tiene un alto grado de aceptación entre los alumnos que la cursan, quedando reflejado en la encuesta que se le pasa al final del cuatrimestre. En la encuesta se les pregunta sobre sus opiniones sobre cada uno de los temas presentados, incluida la práctica, pidiendo que los valoren de 1 a 5. En la figura 1 se muestran los resultados de las encuestas realizadas en el año 1998 y 1999.

En las gráficas se puede apreciar que la suma de los porcentajes de las valoraciones del 3 al 5 es, en todos los casos menos uno, superior al 50% alcanzando el 91% en el tema C/S en el año 1999.

Los temas de "Sistemas Abiertos" y "Calidad del Software" son los que sacan la puntuación más baja los dos años. Las razones habría que buscarlas en la poca importancia que se suele proporcionar a estos temas a lo largo de la carrera, no percibiéndolos como importantes por parte de los alumnos.

Las prácticas de la asignatura son uno los puntos que mejor parados quedan en la encuesta, dando un grado máximo de satisfacción (un 5) un 42% de los alumnos en el año 1998 y un 54% en el 1999. La principal razón de este elevado porcentaje seguramente estriba en que los alumnos valoran más los contenidos prácticos que los teóricos, ya que ***es en las prácticas donde realmente asimilan los contenidos***.

Ante la pregunta de si consideran los contenidos impartidos en la asignatura interesantes para su futuro profesional, el 90% en el año 1998 y el 100% en el año 1999 contestaron afirmativamente.

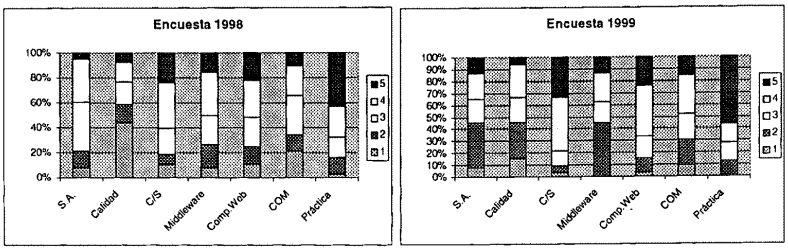


Figura 1: Resultados de las encuestas en los años 1998 y 1999.

5.- CONCLUSIONES.

Se han presentado los contenidos y las prácticas de la asignatura "Entornos de Desarrollo de Software" impartida en la Facultad de Informática de la Universidad Politécnica de Valencia.

Una asignatura en la que se pretende formar a los alumnos en las características comunes que comparten los entornos de desarrollo actuales de forma crítica. De manera que sean capaces de enfrentarse ante cualquier desarrollo de software en su futura vida profesional y que conozcan: de qué herramientas disponen y dónde tienen que buscar información para no quedar desfasados en un campo tan variable como es el de las herramientas de desarrollo. Se les exige que lean revistas del sector y preparen presentaciones. Además, se les ofrece la oportunidad de presentar algún trabajo con el objetivo de que practiquen algo que no es usual: hablar en público y defender sus ideas ante un auditorio.

Finalmente, se han presentado los resultados de las encuestas realizadas a los alumnos en los últimos años, quedando claro el alto grado de satisfacción de los mismos ante los contenidos de la materia.

BIBLIOGRAFÍA

[ANSI] *An Introduccion to ANSI*, http://www.ansi.org/

[Benson] Benson S., *Open Systems - Theory and Reality*

[Ber96] Philip A. Bernstein, (1996), *Middleware*, Communications of the ACM, Febrero.

[Com94] Varios, (1994), *Fundamentos Cliente/Servidor*, Documentos Computerworld, IDG Communications S.A.

[Dou96] Patricia J. Douglas, George M. Alliger, Robert Goldberg, (1996), *Client-Server and Object-Oriented Training*, IEEE Computer, Junio.

[Gar97] David Garlan, David P. Gluch, James E. Tomayko, (1997), *Agents of Change: Educating Software Engineering Leaders*, IEEE Computer, Noviembre.

[IEEE] *The IEEE is...*, http://www.ieee.org/

[Inc95] Darrel Ince, (1995), *Software Quality Assurance*, McGraw Hill.

[ISO] *Introduction to ISO*, http://www.iso.ch/

[Jon95] Capers Jones, (1995), *Gaps in Programming Education*, IEEE Computer, Abril.

[Moe98] José Eduardo Mohedano, Manuel de la Herrán, (1998), *ActiveX y ASP*, Sólo Programadores, Especial Monográfico 2, Enero.

[Pou97] D. Pountain, J. Montgomery, (1997), *Componentes Web*, BYTE, Septiembre.

[Pre97] Roger Pressman, (1997), *Ingeniería del Software: Un enfoque práctico*, McGraw-Hill.

[Rog97] Rogerson D., (1997), *Inside COM*, Microsoft Press.

[Tob97] Bruce Tober, (1997), *El factor calidad*, Byte, Abril.

[Tuc96] Allen B. Tucker, (1996), *Strategic Directions in Computer Science Education*, ACM Computing Surveys, Diciembre.

LA ASIGNATURA DE INFORMATICA APLICADA A LA GESTION DE EMPRESAS EN DIPLOMATURA EN CIENCIAS EMPRESARIALES

Pedro Luis Pérez Serrano[1]

[1]*Facultad de CC. Económicas y Empresariales, Departamento de Informática, Universidad de Extremadura*
e-mail: ***plperez@unex.es***

RESUMEN: El presente trabajo pretende mostrar no el programa de la asignatura de Informática Aplicada dentro de los estudios de Diplomatura en Ciencias Empresariales, sino de explicar cómo se imparte este programa y las conclusiones sacadas una vez terminado el período docente de la misma, siempre analizando las conclusiones tanto positivas como negativas. Simplemente mostramos el programa para cumplimentar el artículo.

1.- INTRODUCCIÓN

La asignatura de Informática Aplicada es una asignatura cuya presencia en la Facultad de Ciencias Económicas y Empresariales de Badajoz se ha visto incrementada en los últimos años, como consecuencia de la modificación y actualización de los nuevos planes de estudio. Así, podemos ver que dicha asignatura se encuentra en la totalidad de las titulaciones que se imparten en dicha Facultad, de tal forma que es troncal en la Diplomatura en Ciencias Empresariales, obligatoria en la Licenciatura en Economía y optativa en las titulaciones de Diplomatura en Relaciones Laborales y Licenciatura en Administración y Dirección de Empresas. En este artículo nos centraremos en la asignatura troncal, si bien el programa es el mismo para el resto de las titulaciones donde se hace presente dicha asignatura. El motivo de ser obligatoria u optativa dependió de la realización de los nuevos planes de estudio de esta Facultad, ya que se pretendió que se hiciese presente en todas las titulaciones de una forma u otra.

La asignatura de Informática Aplicada en Diplomatura en Ciencias Empresariales, es de carácter cuatrimestral, concretamente se imparte en el segundo cuatrimestre, de acuerdo con las nuevas directrices marcadas por el Consejo de Universidades respecto a los nuevos planes de estudio. Consta de 6 créditos, divididos en 4,5 créditos de teoría (tres horas) y 1,5 créditos de prácticas (1 hora).

2.- OBJETIVOS

El objetivo básico buscado en el programa desarrollado es, por una parte con respecto a la teoría, que el alumno adquiera el nivel de conocimientos general del proceso de implantación de un sistema informático, ya que se puede dar la situación de que en la empresa donde trabaje se necesite implantar un sistema informático y él sea el responsable del mismo, aportando la información necesaria para su realización (p. e. se tenga que desarrollar para la empresa un programa de control de costes y él sea el encargado de proporcionar la información necesaria para que los analistas realicen la aplicación). De esta forma él será el encargado de establecer la comunicación con los analistas, aportando nuevas ideas y revisando lo realizado así como aclarar posibles dudas a los analistas en la definición de los procesos. Por lo tanto con el programa desarrollado para esta asignatura, se le enseña al alumno las metodologías que se utilizan para el desarrollo de sistemas, haciendo hincapié en aquellas etapas en las cuales intervienen ellos, como son en la etapa de estudio previo, realización del informe de alcance y objetivos, análisis y pruebas. En el programa comentamos algunas metodologías más utilizadas y debatidas. En resumen, se intenta de que el alumno aprenda de forma general el lenguaje que se utiliza en el desarrollo de sistemas informáticos, para de esta forma establecer una mejor comunicación y por tanto entendimiento entre los analistas y el cliente (ya que muchas veces una mala comunicación analista-cliente es la base del fracaso de muchos proyectos), en este caso el alumno, teniendo siempre presente que el objetivo no es que el alumno desarrolle el mismo la aplicación.

Con respecto al programa desarrollado para la práctica, se pretende que el alumno adquiera conocimientos sobre aquellas utilidades comunes en cualquier empresa, necesarias para realizar los trabajos más elementales.

Para alcanzar estos objetivos aplicando este programa, a continuación se explica cómo desarrollo las clases tanto de teoría como de práctica durante el período docente.

3.- METODOLOGÍA DOCENTE

3.1. Clases de teoría

En la primera clase de la asignatura, al alumno se le explica el contenido de la misma, cuál va a ser el programa, explicando brevemente los capítulos, cómo van a desarrollarse las clases tanto de teoría como las de prácticas y cuál va a ser el tipo de examen y cómo se va a corregir. Por lo tanto el primer día de clase el alumno recibe todo la información que necesita para el correcto seguimiento de la misma.

Al tener la asignatura 6 créditos (4,5t + 1,5p) a los alumnos les corresponde tres horas de teoría a la semana. La distribución de las horas en la semana se realiza impartiendo dos horas seguidas un día y una hora suelta otro día. La clase de dos horas se realizan sin descanso, ya que tengo comprobado que hacer un descanso en medio hace por una parte distraer al alumno y perder la concentración y por otra los descansos siempre se alargan mas de la cuenta, con lo cual lo programado para dar ese día nunca se cumple. Esta decisión no se ha tomado a la ligera, sino que siempre en consenso y consulta con los alumnos, los cuales están de acuerdo de darlas seguidas, y de esta forma tienen un margen de tiempo tanto en la entrada como en la salida.

Para las clases, se optó bien por dejar los temas a los alumnos o bien por la toma tradicional de los apuntes. Después de varias pruebas, el método que sigo es el de dejar los apuntes antes de

empezar a dar un tema. El alumno lo recibe (que se deja en reprografía de la Facultad) antes de impartirlo, avisándole con tiempo de que ya está el nuevo tema, con lo cual puede leerlo antes de explicarse y estar predispuesto a posibles dudas. A pesar de esto, los apuntes son un guión a seguir, lo cual no implica que no se pueda explicar o comentar aspectos u conceptos que no vengan en los mismos (en este caso se le advierte al alumno de la importancia o no de lo que se va a comentar, así como la indicación de bibliografía complementaria). La ventaja es que el alumno siempre puede faltar a alguna clase, con la seguridad de que lo que se explica está reflejado en los apuntes y sabe que con estudiárselos podrá fácilmente (dentro de un margen) aprobar. Evidentemente tiene sus inconvenientes y uno de ellos, que está más que comprobado, es que nos encontramos muchas veces con que el alumno está simplemente subrayando en los apuntes lo que se explica, si bien advierto que puede entrar cualquier apartado que aparezca en los mismos. Hay que tener presente que el examen de teoría es de tipo test, con lo cual el hecho de tener todos los apuntes asegura el tener opción a contestar a todas las preguntas (o al menos al 90%, ya que siempre se suelen poner alguna pregunta que se explica en clase). De todas formas, siempre se hará hincapié en aquellos aspectos que sean importantes y que el alumno debe de aprender (vengan o no en los apuntes). Siempre que existe una duda o comentario, se abre un debate en el cual se intenta que los mismos alumnos resuelvan la duda.

A pesar de la aparente complejidad del programa, el alumno capta bastante bien todo lo que engloba al proceso de desarrollo de un sistema informático (si bien no entramos en aquellas etapas que son más de carácter puramente informático como programación) ya que se les explica de forma clara y siempre apoyado en ejemplos cercanos a su entorno (gestión de almacenes, contabilidad, ...). Para la explicación de la teoría me ayudo de transparencias, las cuales también las dejo, a opción del alumno y a veces de exposiciones hechas con ordenador utilizando un cañón de vídeo. Cuando se les explica la fase de diseño (por ejemplo), el objetivo es de que comprendan y sepan interpretar un DFD, no que sepan hacerlo. Para su comprensión, la forma que he comprobado que es (de momento la mejor) es precisamente a que aprendan las cuatro reglas que deben de cumplir, de tal forma que aprenden a detectar posibles errores en los DFD, con lo cual aprenden a comprenderlo sin prácticamente ningún problema.

El apartado que más dificultad les supone es el referente en la fase de análisis, al explicar la especificación de los procesos con el lenguaje estructurado (*mientras, repetir, caso, si sino*), de tal forma que la realización de un algoritmo les resulta complicado de entender y realizar. En conclusión, para el programa del curso 00/01, eliminaré esta parte y haré más hincapié en el tema de bases de datos relacionales, ya que se puede ligar dicho tema con la parte de práctica donde se verá la base de datos relacional.

3.2. Clases de prácticas

Para la realización de las prácticas a las cuales a cada alumno le corresponde una hora a la semana (1,5 créditos) se dispone de un aula con 22 equipos mas uno para el profesor. El número de alumnos por grupo por lo tanto está definido por el número equipos, de tal forma que como máximo son dos alumnos por ordenador, ya que por cuestiones de horario no se pueden dividir en más grupos para que de ésta forma tocasen a un alumno por equipo.

Para orientar las prácticas, se le hace al alumno rellenar una encuesta que luego me sirve para ver el nivel que tienen los alumnos desde el punto de vista de usuario y qué programas saben usar así como qué programas les gustaría aprender y la posibilidad de poder realizar trabajos o no fuera del horario de prácticas. Dicha encuesta consta de cinco preguntas: *¿tienes ordenador en casa?*, *¿tienes conocimientos de informática?*, *¿qué programas sabes utilizar?*, *¿qué esperas aprender de la asignatura?* y *¿qué te gustaría aprender?*. En general, en casi un 89%, la opinión

generalizada era la de por una parte la necesidad de saber utilizar internet (correo, navegación) y por otra la de saber utilizar paquetes integrados (hoja de cálculo, bases de datos, procesador de texto,...). Para la realización de las prácticas he incorporado hace dos años el uso de un cañón de video, con lo cual se ha visto muy mejorada la calidad de las mismas y además se motivó a que en otras asignaturas de la titulación donde hay prácticas utilizasen dicho cañón.

La ventaja del uso del cañón es que al mismo tiempo que explico bien sea internet, hoja de calculo, etc. el alumno también lo va realizando en su ordenador, con lo cual por una parte el alumno está centrado en su trabajo y por otra parte, va realizando el ejercicio conforme se va explicando. Además todo lo que se realiza en las clases prácticas el alumno debe de entregarlo en un disquete el día del examen (como requisito imprescindible para poder realizar el mismo), con lo cual se trabaja y aprende bastante en las clases de prácticas.

Este curso he observado que el alumno a empezado a utilizar más el ordenador una vez que ha adquirido conocimientos, sobre todo de hoja de cálculo e internet, ya que se ha visto incrementado la realización de trabajos en otras asignaturas utilizando los conocimientos adquiridos en esta. De todas formas, la parte referente a internet el próximo curso no será incluida en el temario ya que prefiero profundizar más en la definición y mantenimiento de una base de datos y dejar esta parte para la asignatura de libre elección que se oferta de Informática Básica.

En definitiva, las clases prácticas aportan unos conocimientos adicionales (siempre fácilmente asimilables por parte de los alumnos), los cuales en parte (bases de datos p. e.) están ligados con temas de la teoría. Podría haberse pensado en dar alguna aplicación como por ejemplo MS-Project, pero no lo he considerado ya que el objetivo no es (como se ha recalcado) el que ellos desarrollen aplicaciones y siempre hay que tener presente que se trata de una titulación de económicas no de informática.

De todas formas siempre nos encontramos con la falta de más horas de prácticas, ya que la mayoría de los alumnos carecen de ordenador propio y no les es fácil el acceso a uno para la práctica de las distintas aplicaciones estudiadas y sobre todo para el uso de internet. Para subsanar un poco esta situación, la mitad de las horas de tutorías (3) las realizo en la sala de prácticas, con lo cual el alumno puede entrar y practicar y si tiene alguna duda resolvérsela en ese momento.

3.3. Realización del examen

Para la realización del examen (como ya he comentado anteriormente) es condición imprescindible la entrega del disquete con todas las prácticas realizadas durante el curso. El examen de teoría consiste en un examen de tipo test de 25 preguntas, tal que las preguntas contestadas mal cuentan –0,1, las preguntas acertadas cuentan 0,4 y las dejadas en blanco no cuentan ni positiva ni negativamente. El alumno se queda con una copia de las respuestas que luego comparará con la solución del examen que se publica unos minutos después de terminar el examen. El motivo por lo que uso este sistema es precisamente para las revisiones de examen. El alumno calcula su nota comparando sus respuestas con la solución. De esta forma al ver la lista de notas puede ver si hay anomalías o no en el cálculo de la misma. Así, el alumno ya va predispuesto a la revisión, y en el 99% de los casos reclamados, se había producido un error en el cálculo por parte del profesor o del lector óptico (si se había cumplimentado el examen en hojas codificadas). Este sistema lo han ido incorporando otros profesores de esta Facultad, debido a la eficiencia del mismo. La nota final se calcula con la suma de la nota de teoría y la de prácticas una vez revisado el disco del alumno. En caso de que el alumno tenga una nota (en

total) entre 4,5 y 4,9, si el alumno quiere (tiene interés en aprobar), se le realiza un examen práctico (en la revisión del examen) dándole su disco de prácticas y haciendo que modifique alguna, con lo cual se puede observar también si la ha realizado él o no.

4.- PROGRAMA TEÓRICO Y PRÁCTICO

Parte de teoría: Para cumplir los objetivos mencionados anteriormente se ha desarrollado este programa (siempre susceptible de ser mejorado) en el cual en el tema 1 se le introduce en el concepto de sistema de información y las características que tiene. En el tema 2 se le explica al alumno el porqué de la necesidad de una metodología, las causas que originaron el uso de las mismas y el concepto de metodología. Llegados a éste punto, el alumno ya toma conciencia de que se necesitan unas normas, directrices para la realización de un sistema informático. Hasta este tema, el alumno lleva perfectamente captados todos los conceptos. En el tema 3 una vez que el alumno sabe lo que es un sistema, qué es dato, qué es información, metodología y porqué se necesita el uso de una metodología, se comentan algunas de las más utilizadas, empezando en éste tema a indicar al alumno en qué etapas interviene junto con los analistas, sobre todo cuando se le comenta el ciclo de vida en espiral. A partir de aquí siempre que se habla de cliente, el alumno sabe que esta es su figura. En el tema 4 se explica con un poco mas de detalle una metodología en la cual comentamos las etapas en donde el cliente (alumno) forma parte, explicando en el tema 5 los DFD, ya que a menudo en las reuniones analistas-clientes, se puede intercambiar información a través de los DFD, de aquí que uno de los objetivos sea el que el alumno entienda e interprete un DFD. El tema 6 y 7 están orientados a adquirir conocimientos para ser aplicados en la práctica.

Tema 1. Introducción a los Sistemas.
1. Introducción.
2. Concepto de sistema.
3. Clasificación de los sistemas.
4. Sistemas de información y sistemas informáticos.

Tema 2. Introducción al análisis de sistemas.
1. Introducción.
2. ¿Porqué una metodología?.
3. ¿Quiénes intervienen en el desarrollo de sistemas?
4. Estilos de interfaces hombre-máquina.
5. El centro de Proceso de datos.

Tema 3. Ingeniería del Software: Metodologías de análisis y desarrollo.
1. Introducción.
2. Definición de Ingeniería del Software.
3. Ciclo de vida clásico.
4. Diseño en espiral.
5. Prototipos.

Tema 4. Ciclo de Vida Estructurado.
1. Introducción.
2. Actividades del ciclo de vida estructurado.
   Estudio previo.
   Análisis.
   Diseño.

Codificación.
Pruebas.
Elaboración de procedimientos.
Conversión de bases de datos.
Implantación.

Tema 5. Diagramas de Flujos de Datos.
1. Introducción a los DFD.
2. Elementos de un DFD.
3. Niveles de un DFD.
4. Reglas para confeccionar un DFD.

Tema 6. Modelado de datos.
1. Introducción.
2. El modelo entidad-relación.
3. Elementos de un diagrama entidad-relación.
4. Cardinalidad de las relaciones.

Tema 7. Introducción a las Bases de Datos.
1. Introducción.
2. Sistemas de almacenamientos tradicionales.
3. Componentes de una base de datos.
4. Características de las bases de datos.
5. Tipos de bases de datos.

Parte de Práctica: Para desarrollar este programa he tenido en cuenta las necesidades, peticiones de los alumnos (datos obtenidos de la encuesta mencionada anteriormente), y las aplicaciones que están de más uso en la actualidad. La relación entre la teoría con la práctica se encuentra a partir del tema 6, donde una vez que el alumno estudia el modelo entidad-relación, sabe qué es una clave, qué es una relación, qué es una base de datos, etc, pasamos a la práctica para crear una base de datos relacional y realizar consultas sobre la misma.

Tema 1. Internet.
1. Introducción a Internet.
2. Fundamentos técnicos.
3. Principales servicios.
4. Ventajas e inconvenientes.
5. La economía en Internet.

Tema 2. Bases de datos.
1. Introducción.
2. Diseño de una base de datos
   Modelo conceptual.
   Modelo relacional.
3. Implementación de una base de datos.
   Tablas.
   Relaciones.
   Datos.
   Consultas.
   Formularios.
   Informes.

Tema 3. Hojas de cálculo.
1. Introducción.
2. Fórmulas y funciones.
3. Gráficos y diagramas.
4. Ordenación.
5. Posiciones absolutas y relativas.
6. Inclusión de gráficos y tablas desde editores de texto.

5.- CONCLUSIÓNES

En definitiva, pese a la aparente complejidad del temario, sobre todo el de teoría, el resultado de aprobados con respecto al de suspensos es muy superior, si bien el objetivo de la asignatura se centra en la comprensión de cómo es todo el proceso de implantación de un sistema de información, no de que ellos aprenda a desarrollarlo (esto se le deja bien claro el día de la presentación de la asignatura). Además se ha intentado con éste programa realizar una asignatura lo más interesante posible para el alumno, como lo demuestra el alto porcentaje de asistencia, un 89% a clases de teoría y un 100% a clases de prácticas (este alto porcentaje viene por la realización de la encuesta de tal forma que se adapta la práctica a las peticiones e inquietudes de los alumnos), si bien hay que mejorar cada año el programa para suplir las deficiencias que se puedan detectar ó para adaptarlo al nuevo entorno cambiante en el cual nos movemos. En cuanto a la parte de prácticas, la adquisición de conocimientos tanto de bases de datos como sobre todo de hojas de cálculo, anima al alumno a realizar trabajos por ordenador de otras asignaturas en las cuales desarrollan los conocimientos que van adquiriendo a lo largo del cuatrimestre.

6.- REFERENCIAS BIBLIOGRÁFICAS

[Haw 88] Introducción al Análisis de Sistemas. I. I Hawryuzkiewycz. Ed. Anaya, 1988

[Ign 97] Sistemas y Tecnologías de la información para la Gestión. Ignacio Gil Pechuán. Ed. MacGraw-Hill, 1997

[Ray 00] Sistemas de Información Gerencial. Raymond McLeod. Ed. Pearson Educación, 2.000

[Ken 91] Análisis y Diseño de Sistemas. Keneth E. Kendall. Ed. Prentice-Hall, 1991

[Pia 96] Análisis y Diseño Detallado de Aplicaciones Informáticas de Gestión. Mario G. Piattini. Ed. RAMA, 1996

[Pre 97] Ingeniería del Software. Roger S. Pressman. Ed. MacGraw-Hill, 1997

[Ari 93] Ciclo de Vida Estructurado Orientado a Procesos. Juan Arias, Manuel Díaz, Amparo Martínez. Ed. Análisis I.P.L., 1993

[Dia 93] Análisis y Diseño de Sistemas Estructurado por Datos. Manuel Díaz, Juan Arias, Amparo Martínez. Ed. Análisis I.P.L., 1993

[Map 88] METRICA 2.1. Ministerio de Administraciones Públicas. Ed. Tecnos, 1988

[You 93] Análisis Estructurado Moderno. Edward Yourdon. Ed. Prentice-Hall, 1993

[Han 97] Diseño y Administración de Bases de Datos. Gray W. Hansen. Ed. Prentice-Hall, 1997

[Dat 93] Introducción a los Sistemas de Bases de Datos. Volumen I. C. J. Date. Ed. Adisson-Wesley, 1993

[Chr 97] Fundamentos de Sistemas de Información. Chris Edwards, John Ward, Andy Bytheway. Ed Prentice-Hall, 1997

PROGRAMA PRÁCTICO DE UNA ASIGNATURA DE INGENIERÍA DEL SOFTWARE DISTRIBUIDO

Asier Perallos[1], Rebeca Cortazar[1]

[1]*Departamento de Ingeniería del Software ESIDE (Universidad de Deusto)*
e-mail: perallos@eside.deusto.es, cortazar@eside.deusto.es

RESUMEN: En esta contribución describimos de manera detallada el programa práctico de la asignatura *Ingeniería del Software III*, del plan de estudios implantado en 1996 en ESIDE (Universidad de Deusto). Esta asignatura es troncal y se impartirá por primera vez a los alumnos de 5° durante el primer cuatrimestre del curso 2000-2001. Mediante su docencia, nuestra intención es acercar a los alumnos los conceptos involucrados en el desarrollo de software distribuido, así como afianzar algunos de ellos mediante las clases de laboratorio que describimos.

1.- INTRODUCCIÓN

Durante el próximo curso 2000-2001, se impartirá por vez primera la asignatura *Ingeniería del Software III*, cuyo programa teórico ya presentamos en JENUI'99. En aquel artículo [1], justificamos los contenidos de la asignatura, que van orientados hacia el desarrollo de aplicaciones distribuidas, tanto desde un punto de vista técnico como desde el punto de vista de los aspectos metodológicos. Por otra parte, el plan de prácticas quedó someramente planteado, puesto que era un tema que en aquel momento no había sido elaborado. Lógicamente, durante este curso académico hemos abordado la labor de preparar los contenidos de las sesiones de laboratorio cuyo objetivo, contenido, organización y metodología se detallan a continuación.

2.- PROGRAMA PRÁCTICO

El objetivo fundamental de las sesiones de laboratorio es facilitar al alumno la comprensión de una parte de los contenidos teóricos estudiados [2]. Las prácticas permitirán que el alumno compruebe por su cuenta el funcionamiento real de los conceptos explicados en clase y mostrarán una visión práctica de éstos. Al mismo tiempo, le darán la oportunidad de familiarizarse con tecnologías novedosas y por tanto, muy necesitadas de profesionales en el mercado, logrando que los alumnos terminen la carrera debidamente preparados para satisfacer la demanda del mercado laboral.

Las sesiones de laboratorio se basarán en el desarrollo de aplicaciones distribuidas centrándose en tres áreas: la comunicación entre objetos remotos, la tecnología de componentes y el acceso remoto a bases de datos. Concretamente, las tecnologías usadas para la realización de las prácticas serán RMI, CORBA y JDBC [3], todo ello sobre la plataforma Java.

La selección de estas tecnologías se fundamenta en los siguientes motivos:

- Son tecnologías actuales, no excesivamente complejas de utilizar y que se complementan muy bien con los conceptos teóricos estudiados, permitiendo experimentar con ellos y contrastarlos.

- El uso de estas tecnologías no requiere de grandes recursos software ni hardware. El software necesario está basado en la plataforma Java y se encuentra disponible de forma gratuita, por lo tanto se posibilita que las prácticas puedan ser realizadas fuera de la universidad.

- Los conocimientos necesarios para su utilización se amoldan perfectamente al perfil de nuestros alumnos. Éstos conocen todos los conceptos de análisis y diseño orientado a objetos al haber cursado la asignatura troncal de *Ingeniería del Software II*, poseen conocimientos de programación orientada a objetos al haber cursado la asignatura de *Tecnología de la Programación* y conocen la plataforma Java al haber realizado múltiples prácticas basadas en ella a lo largo de la carrera.

- Por último, estas tecnologías, junto con otras del temario teórico, son lo suficientemente extensas y novedosas como para animar a los alumnos a que realicen proyectos de fin de carrera basados en ellas.

Los créditos prácticos de la asignatura serán cubiertos mediante la realización de tres prácticas que se desarrollarán siguiendo la siguiente pauta: en primer lugar, se analizará el funcionamiento de la tecnología en estudio mediante un ejemplo trivial (como el clásico 'hola mundo') y a continuación se investigará un ejemplo más complejo. Sobre este último se plantearán modificaciones que los estudiantes deberán implementar y entregar de manera obligatoria. A continuación, se expone el contenido de cada una de las prácticas.

a) La primera práctica

Tendrá como objetivo la invocación de métodos de un objeto remoto a través de RMI. En primer lugar se expondrán los fundamentos necesarios para poder utilizar RMI y se comprobará su funcionamiento con un ejemplo trivial como el típico 'hola mundo'.

Seguidamente, se presentará una sencilla aplicación distribuida consistente en un Servidor de Donaciones al que se conectan varios clientes a través de RMI para realizar Donativos. Utilizaremos este software base como ejemplo de la tecnología RMI y como práctica plantearemos una modificación: se debe añadir una nueva funcionalidad al servidor que permita a los clientes consultar la cantidad total donada que ha sido recogida hasta ese momento. Esta modificación requerirá que los alumnos inserten un nuevo método en la interfaz remota del servidor y que lo invoquen desde los clientes a través de RMI (Figura 1).

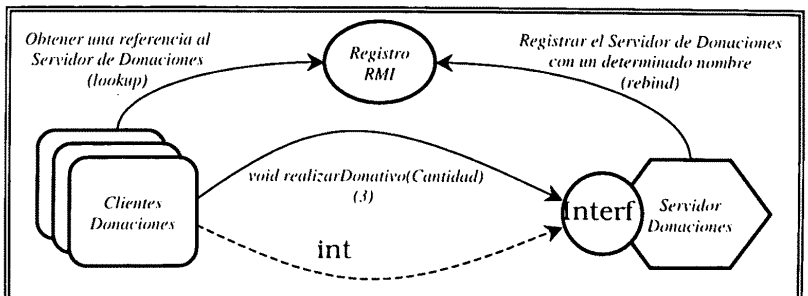


Figura 1: Esquema de la primera práctica.

Los objetivos a cubrir en esta práctica son que los alumnos se familiaricen con un mecanismo sencillo de comunicación entre objetos remotos como es RMI y que conozcan todos los elementos que hacen posible esta comunicación como son entre otros, el registro RMI y las interfaces remotas.

b) La segunda práctica

La segunda práctica estará basada en la tecnología de componentes centrándose en el modelo de componentes CORBA [4]. Partirá de la misma aplicación de Donativos que la usada en la práctica anterior pero implementada mediante componentes CORBA en lugar de RMI. También utilizaremos esta aplicación como ejemplo para explicar la tecnología CORBA y como práctica plantearemos una modificación sobre ella. Dicha modificación consistirá en la implementación de un patrón *Observer* [5] que permita a los clientes ser notificados en todo momento de la cantidad total donada. Así, cada vez que un cliente realice un donativo, el resto de clientes verá actualizada automáticamente esa cantidad.

Para implementar este patrón se deberá establecer un Sistema de Subscripción entre los clientes y el servidor, para lo cual será necesario que los clientes sean convertidos también en componentes CORBA (Figura 2).

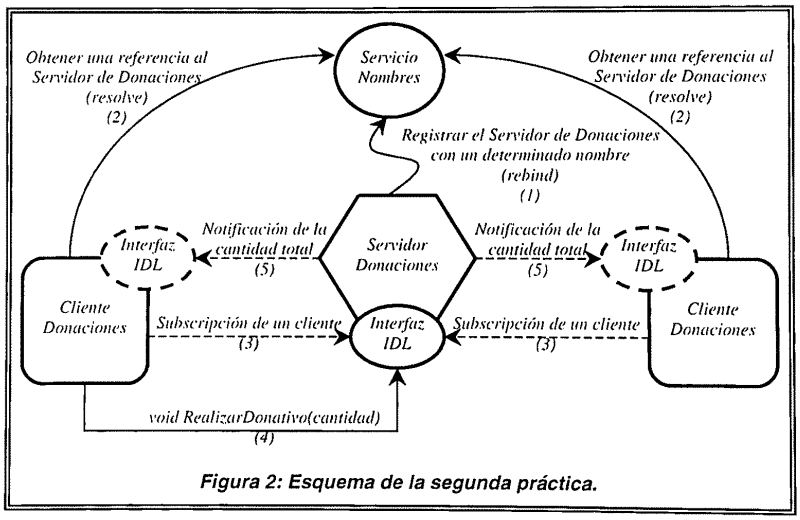


Figura 2: Esquema de la segunda práctica.

Los objetivos a cubrir en esta práctica son que los alumnos se familiaricen con la tecnología de componentes y más en concreto con el modelo de componentes CORBA y que sean capaces de reconocer dentro de una aplicación los elementos de la arquitectura CORBA estudiados en las clases teóricas.

c) La tercera práctica

Ésta tendrá dos partes diferenciadas: una primera dedicada a JDBC y una segunda parte para el planteamiento de una práctica de dimensiones mayores, integradora de las tecnologías probadas.

Para explicar el acceso a bases de datos remotas a través de JDBC, se añadirá a la aplicación de Donativos un objeto que se encargue del acceso a la base de datos de manera que nuestros alumnos dispongan de esta información como ejemplo. El nuevo objeto se encargará de, a través de JDBC, ir almacenando en la base de datos todas las donaciones que van realizando los clientes así como la fecha y la hora en que se realiza cada donación (Figura 3).

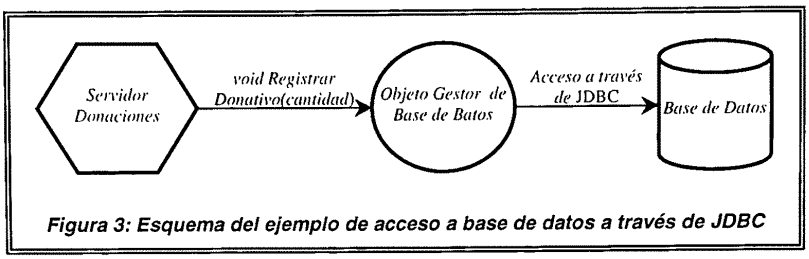


Figura 3: Esquema del ejemplo de acceso a base de datos a través de JDBC

No plantearemos ninguna ampliación sobre este ejemplo ya que consideramos que van a utilizar los accesos a bases de datos con suficiente detalle en las ampliaciones que se propongan en la siguiente etapa.

En esta segunda parte se integrarán todas las tecnologías usadas en las prácticas anteriores: RMI, CORBA y JDBC. Su finalidad será desarrollar una aplicación distribuida de pedidos de libros (tipo Amazon). Para ello, se proporcionará a los alumnos el núcleo de la aplicación ya implementado y su labor será realizar una serie de ampliaciones sobre él. El funcionamiento de la aplicación será el siguiente: los clientes se conectarán al servidor remoto a través de RMI para realizar sus pedidos. Este Servidor de Pedidos será quien contenga toda la funcionalidad necesaria para realizar un pedido. Para ello, hará uso de una serie de componentes externos que se encargarán de realizar funciones tales como realizar la Facturación del pedido, la Estimación de su fecha de entrega o la Verificación de la validez del nº de cuenta bancaria del comprador. Algunos de estos componentes serán componentes CORBA y otros implementarán interfaces remotas RMI. Además, el servidor almacenará toda la información relativa a los compradores, los libros y los pedidos en una base de datos remota a la que accederá a través de JDBC. Sobre el núcleo de la aplicación los alumnos deberán realizar una serie de ampliaciones (Figura 4).

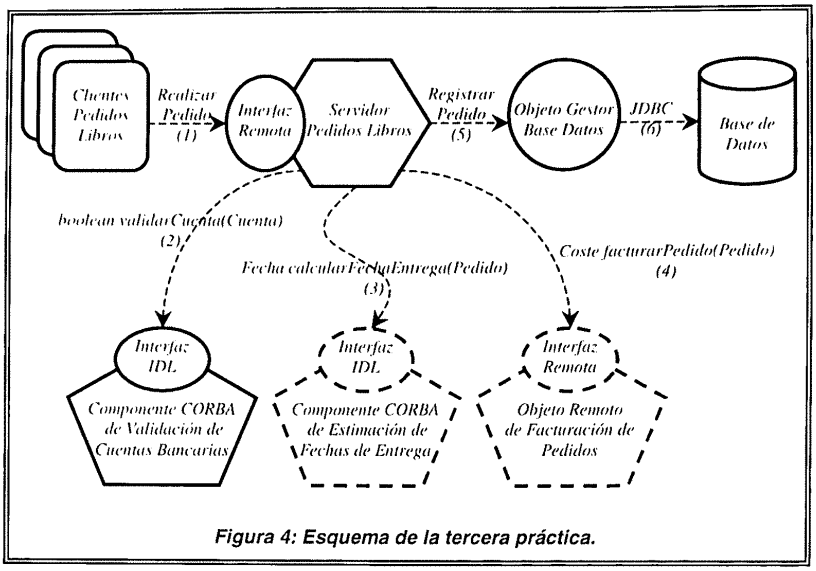


Figura 4: Esquema de la tercera práctica.

Atendiendo a sus características, las ampliaciones se pueden clasificar en tres tipos diferentes:

- El primer tipo de ampliación consistirá en utilizar los servicios proporcionados por algunos de los componentes externos con los que se debe comunicar la aplicación. En este caso, los componentes a invocar se proporcionarán ya implementados a los alumnos.

- El segundo tipo consistirá en que los alumnos implementen algunos componentes externos, tanto componentes CORBA como componentes que implementen interfaces remotas y los invoquen.

- El tercer tipo consistirá en añadir alguna funcionalidad nueva a la aplicación como, por ejemplo, la adición de nuevos compradores. Para que esto sea posible, los alumnos deberán realizar desde el cliente invocaciones RMI al servidor y deberán implementar mediante JDBC los accesos que el servidor realice a la base de datos.

Los objetivos a cubrir en esta práctica son que los alumnos sepan integrar en una única aplicación todas las tecnologías estudiadas y que comprendan la arquitectura de una aplicación distribuida de cierta complejidad así como su implementación.

3.- NECESIDADES DE SOFTWARE

La herramienta de desarrollo que se usará en las prácticas será *Java 2 SDK*. Esta herramienta incluye todo lo necesario para poder realizarlas: contiene el paquete *java.rmi* que permite realizar invocaciones remotas de métodos a través de RMI, implementa un ORB CORBA, posee un compilador de IDL a Java y proporciona el paquete *java.sql* para el acceso a bases de datos a través de JDBC.

4.- ORGANIZACIÓN DE LAS CLASES PRÁCTICAS

Las clases de prácticas se impartirán en aulas del centro de cálculo donde se dispone de un cañón de proyecciones que permitirá mostrar el código de los ejemplos y su ejecución desde el ordenador del profesor, una pizarra que servirá de apoyo en las explicaciones y ordenadores donde los alumnos podrán ir realizando tanto los ejemplos guiados (ejemplos que realiza toda la clase, paso a paso, al mismo tiempo que el profesor), como las modificaciones propuestas.
Las clases de prácticas se impartirán durante las semanas 6ª, 8ª y 10ª del cuatrimestre (meses de noviembre y diciembre) en grupos de tres sesiones cada semana; por tanto, un total de 9 sesiones de una hora de duración. Evidentemente, esto implica una coordinación entre las clases teóricas y las prácticas para garantizar que los contenidos ya han sido previamente impartidos. (Esto nos ha llevado a modificar la planificación de temas, ya que por ejemplo, originalmente se estudiaría CORBA en el mes de diciembre, trastocando así las prácticas. Hemos solucionado el problema dividiendo ese tema en dos partes: CORBA Básico (con contenido suficiente para las prácticas) y CORBA Avanzado, donde se explica el resto).

Para la realización de las prácticas los alumnos formarán grupos de 2 personas. Cada grupo deberá entregar el código fuente de las tres prácticas, siendo éste requisito indispensable para superar la asignatura. Las entregas de las dos primeras prácticas están planificadas para las semanas 7ª y 9ª del cuatrimestre. La entrega de la tercera la hemos pospuesto hasta el mes de enero por ser la que más trabajo requiere. Además, consideramos importante esta entrega paulatina de los resultados de las prácticas porque la experiencia nos ha demostrado que funciona mejor que unir todas las entregas en una y proporcionar una plazo amplio.

5.- CONCLUSIONES

Hemos descrito el programa práctico que esperamos impartir en el curso 2000-2001 en la asignatura de *Ingeniería del Software III*. Creemos que se adecua para lograr el objetivo fundamental que se persigue con esta asignatura, tanto en su vertiente teórica como práctica: enseñar los conocimientos necesarios para el desarrollo de software distribuido.

6.- BIBLIOGRAFÍA

[1] R. Cortazar, A. Barredo, J.L. Del Val, "Propuesta de Contenidos para una Asignatura de Ingeniería del Software Distribuido". V Jornadas sobre la Enseñanza Universitaria en Informática (JENUI'99), La Almunia de Dª Godina (Zaragoza), 25 y 26 de Octubre, pg:141-146.
[2] R. Orfali, D. Harkey, J. Edwards, "The essential Client/Server Survival Guide". John Wiley & Sons, 3ª Ed, 1999.
[3] G. Seshadri, "Enterprise Java Computing: Applications and Architecture". SIGS Books, 1999.
[4] Object Management Group, "The common object request broker: architecture and specification". Object Management Group. Revisión 2.3.1. Octubre 1999.
[5] E. Gamma, R. Hemlm, R. Johnson, J. Vlissides, "Design Patterns, Elements of Reusable Object-Oriented Software". Addison-Wesley, 1995.

LOS 330 CRÉDITOS Y LA INGENIERÍA DEL SOFTWARE

José Javier Dolado[1]

[1]Facultad de Informática de San Sebastián (UPV-EHU)
e-mail: dolado@computer.org

RESUMEN: La ingeniería del software está recibiendo una atención especial dentro de los grupos relacionados con la enseñanza y los diseños de curricula informáticos. En este trabajo vamos a exponer algunas ideas relacionadas con el diseño general de un curriculum orientado hacia la ingeniería del software. Las cuestiones se pueden abordar desde el punto de vista particular de la educación en nuestro país y, también, desde el punto de vista más general de la enseñanza de la informática a nivel mundial.

1. LA EDUCACIÓN EN INGENIERÍA DEL SOFTWARE

En estos momentos, la educación superior informática está influenciada principalmente por los contenidos que se imparten bajo el epígrafe "ciencias de la computación". Pero en los últimos años ha aparecido una tendencia que trata de dotar del carácter de profesión y de entidad propia a la "ingeniería del software". Aunque el contexto educativo es el de EEUU, bastantes de las cuestiones que se abordan son aplicables a la enseñanza en nuestro país. Considerar la "ingeniería del software como una profesión" supone tener una nueva perspectiva sobre qué constituye la esencia de la actividad profesional informática. Una manera de ilustrar la diferencia entre algunas disciplinas informáticas se muestra en la figura 1, propuesta por R.L. Glass, que se ha utilizado posteriormente en las propuestas SWEBOK y SE-BOK (que se describen más adelante).

Este artículo fue premonitorio de la polémica que estamos viviendo hoy en día alrededor de la disciplina de la ingeniería del software [Glass, 1992]. En él, este conocido autor ya apuntó las posibles relaciones entre las disciplinas de "Computer Science", "Software Engineering", "Information Systems" y los dominios de aplicación.

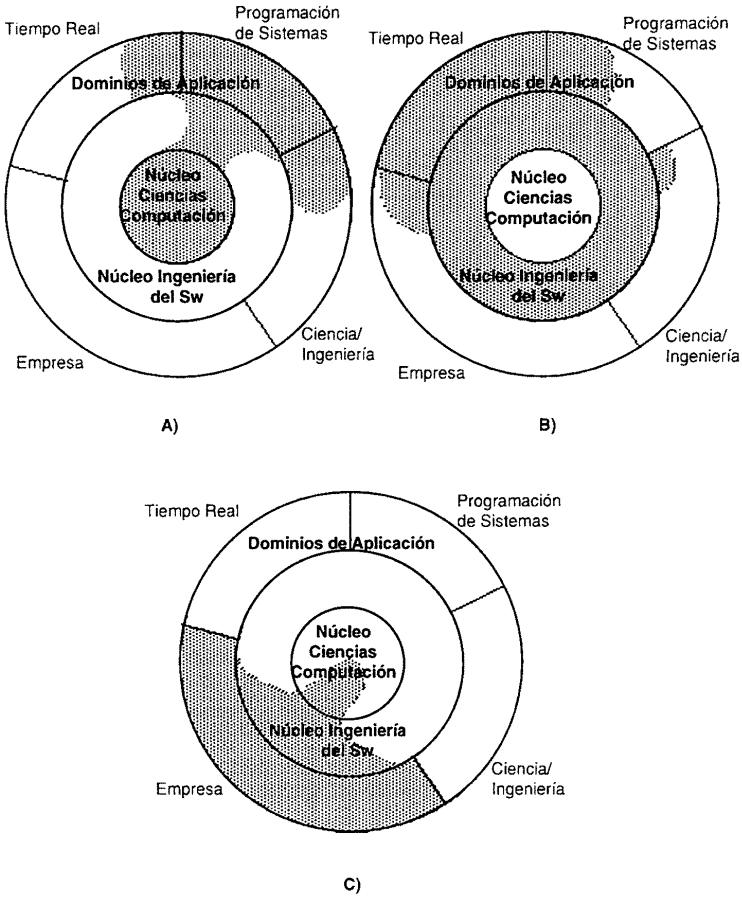


Figura 1. Núcleos y aplicaciones de las disciplinas: a) Ciencias de la Computación; b) Ingeniería del Software; c) Sistemas de Información

En la figura 1 se muestran los núcleos de la ingeniería del software, de las ciencias de la computación y los dominios de aplicación. Una clasificación general de los éstos últimos es: a) programación de sistemas; b) aplicaciones de empresa; c) aplicaciones científicas y de ingeniería y d) tiempo real. Las tres figuras nos describen qué es lo que abarcan cada una de las disciplinas. Es curioso observar cómo Glass separó los núcleos de ingeniería del software de los de las ciencias de la computación, afirmando que la primera posee su propio núcleo. Hecho éste que las publicaciones actuales están refrendando.

2. FUENTES CURRICULARES

Existen varias fuentes en las que nos podemos basar para definir la estructura de un curriculum, que a continuación vamos a enunciar. De la lectura de estos textos podemos concluir que la propuesta más ambiciosa, referida a la ingeniería del software, es la del SWEBOK, aunque no sea una propuesta curricular.

a) Propuesta SWEBOK

El *SoftWare Engineering Body of Knowledge* (ver http://www.swebok.org, SWEBOK) es un subproducto de un comité conjunto (SWECC) de la ACM y del IEEE, cuyo objetivo es "establecer los conjuntos de criterios apropiados y las normas para la práctica profesional de la ingeniería del software, sobre las que se puedan basar las decisiones industriales, la certificación profesional y los curricula".

La principal actividad de este grupo de trabajo es la de definir el núcleo del cuerpo de conocimiento de la ingeniería del software, puesto que a partir de ahí se puede definir la profesión y los diferentes curricula universitarios. Este cuerpo de conocimiento se construye, primero, a partir del "conocimiento generalmente aceptado" (CGA). Este CGA es un subconjunto de lo que sería el cuerpo total de conocimiento de la disciplina de la ingeniería del software. El objetivo del SWECC es crear una guía que permita identificar y describir el CGA. También se indica que los ingenieros del software no sólo deben conocer el CGA, sino que también deberán tener nociones de otras disciplinas. Por otra parte, identificar el CGA no equivale a establecer los curricula para la ingeniería del software. El SWECC lo deja bien claro: "Software engineering body of knowledge and curriculum are not the same".

En estos momentos se han identificado diez áreas de conocimiento de ingeniería del software, que se están detallando, de tal manera que para finales de 2001 se disponga de la descripción del "núcleo de conocimiento de la ingeniería del software:

1. Requisitos del Software
2. Diseño del Software
3. Construcción del Software
4. Pruebas del Software
5. Mantenimiento del Software
6. Gestión de Configuración del Software
7. Calidad del Software
8. Gestión de la Ingeniería del Software
9. Infraestructura de Ingeniería del Software
10. Proceso de Ingeniería del Software

Las "disciplinas relacionadas" con la ingeniería del software que propone el SWEBOK son:

- Ciencias de la Computación ("Computer Science")
- Matemáticas ("Mathematics")
- Gestión de Proyectos ("Project Management")
- Ingeniería de Computadores ("Computer Engineering")
- Ingeniería de Sistemas ("Systems Engineering")
- Gestión y Ciencias Empresariales ("Management and Management Science")
- Ciencias Cognitivas y Factores Humanos ("Cognitive Sciences and Human Factors")

b) Propuesta del Software Engineering Institute SE-BOK

El proyecto SE-BOK y las guías del Software Engineering Institute de Carnegie Mellon (http://www.sei.cmu.edu), al contrario que el SWEBOK, sí que pretenden desarrollar curricula basados en el cuerpo de conocimiento que ellos definen. Esta propuesta se ha realizado casi simultáneamente al trabajo del SWEBOK, [Hilburn, 1999] para el cuerpo de conocimiento de la ingeniería del software. De la lectura de tal informe observamos que bastantes áreas de conocimiento SE-BOK se solapan con las del SWEBOK. Básicamente SE-BOK divide los contenidos en las categorías de Ingeniería del Producto e Ingeniería del Proceso. Además añade una categoría de Dominios del Software y otra de Fundamentos de la Computación. Cada una de ellas las divide a su vez en Áreas de Conocimiento y éstas a su vez en Unidades de Conocimiento

c) Computing Curricula 2001

El CC-2001 es la otra iniciativa conjunta ACM/IEEE "Year 2001 Model Curricula for Computing: CC-2001", que pretende definir los contenidos del área *Computer Science*.

d) Otras referencias

Complementariamente se pueden utilizar las referencias de [Parnas, 1998] [Greening, 2000], aunque este último compendio de trabajos es excesivamente general. Han aparecido números especiales dedicados a la profesión y educación en la ingeniería del software en: IEEE Software, Nov/Dec, 1999; Information and Software Technology, vol. 40, n4, 1998; Journal of Systems and Software, vol. 49, n2, 1999 e IEEE Computer, Mayo, 2000, además de [ASE, 1998].

4.- ESTRUCTURA CURRICULAR GENERAL

Puesto que nuestro problema es de índole práctica, nos preguntamos ¿Cuántos créditos se pueden impartir teniendo en cuenta principalmente la propuesta del SWEBOK? La respuesta no

es fácil, puesto que por encima de conocimiento fundamental se sitúa el avanzado y el especializado, y esto permite tener curricula con una extensión flexible. Vamos a describir la estructura general que puede proporcionar, al menos, el núcleo del cuerpo de conocimiento del SWEBOK, incluyendo también algunas recomendaciones del SE-BOK. Es difícil proponer un plan de estudios simplemente mimetizando las guías del SE-BOK, puesto que la formación universitaria es diferente en cada país.

Lo que sí podemos hacer es proponer una distribución general de áreas de conocimiento, en créditos y porcentajes, que incluyan las ideas del SWEBOK, SE-BOK y CC-2001. La figura 2 representa (aproximadamente) los 330 créditos (también aproximadamente) de una ingeniería en informática orientada hacia la ingeniería del software, repartidos de la siguiente manera:

A. Cuerpo de conocimiento de la ingeniería del software, incluyendo el núcleo del cuerpo de conocimiento, conocimiento avanzado y determinados aspectos de conocimiento especializado: 25 % de los créditos o unos 80 créditos.
B. Ciencias de la computación, con los contenidos que se definen en el CC-2001: 36% de créditos (120).
C. Ingeniería de computadores: 30 créditos o un 9% aproximadamente
D. Ciencias matemáticas y físicas: 18% o unos 60 créditos
E. Ingeniería de sistemas: un 2,5% (8 créditos)
F. Gestión de proyectos: según se define por el Project Management Institute (un 5%, 16 créditos). Se debe tener en cuenta que parte de estos contenidos ya se incluyen en el área de "Gestión de la ingeniería del software" del SWEBOK.
G. Ingeniería de calidad: que va englobada dentro de la parte de calidad del SWEBOK
H. Gestión y ciencias empresariales: 2%, aproximadamente.
I. Otros: un 2,5 % aproximadamente, para cursar cualquier otro contenido.

4. CONCLUSIÓN

Ya hemos indicado que la gran cuestión reside en si se define o no (o cómo se define) la disciplina de "ingeniería del software". La emergencia de la ingeniería del software puede proporcionar grandes beneficios a la comunidad de profesionales informáticos. Y la orientación de la educación hacia la profesionalización es otro hecho que va a marcar las pautas educativas de los próximos 20 años. No obstante, existen algunas posturas divergentes como la que mantiene P.J. Denning, que cree que existe "the Profession of Computing", en la que se englobarían las diferentes especialidades informáticas. Sin embargo, no es ésta la tendencia general. Actuar de esta última manera obligaría a un cambio radical en las actuaciones que siguen tanto el SE-BOK, como el SWEBOK, que son las que van a marcar las pautas en los próximos años.

Agradecimientos

Recibido apoyo de UPV-EHU 141.226 EA083-98 y CICYT TIC99-0351.

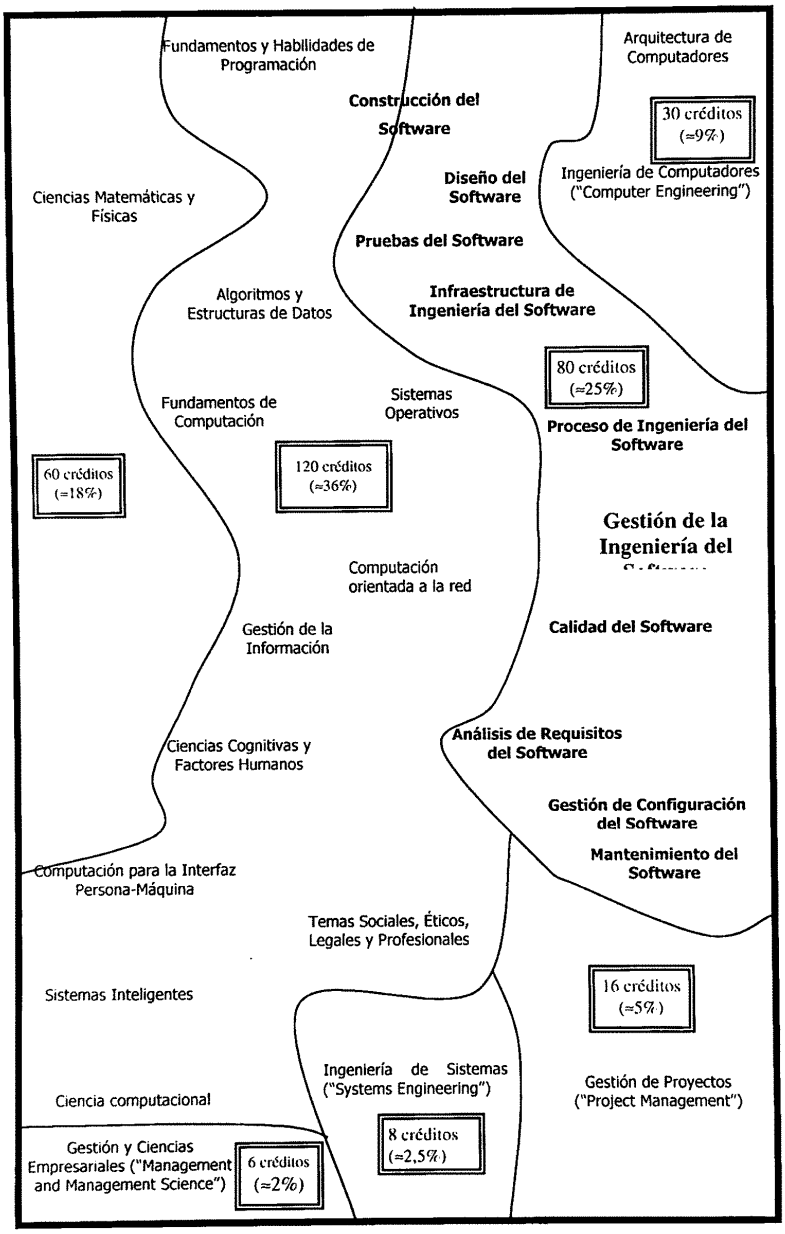


Figura 2. Posible estructura curricular orientada a la ingeniería del software.

5.- REFERENCIAS

[ACM, 1999] ACM, ACM Panel on Professional Licensing in Software Engineering Report to Council (8 de julio,1999), http://www.acm.org

[Andrews, 1999] D. Andrews, Software engineering education in the 21st century, Information and Software Technology 41 (1999) pp. 933-936

[ASE, 1998] Annals of Software Engineering, Special issue on software engineering education, vol. 6, 1998.

[Bagert, 1999] D.J. Bagert, T.B. Hilburn, G Hislop, M. Lutz, M. McCracken y S. Mengel.

Guidelines for Software Engineering Education, Version 1.0. Technical Report, CMU/SEI-99-TR032, Carnegie Mellon University.

[Bourque, 1999] Pierre Bourque, Robert Dupuis, Alain Abran, James W. Moore, and Leonard Tripp, The Guide To The Software Engineering Body Of Knowledge, IEEE Software, November/December 1999

[Glass, 1992] R.L. Glass, "A Comparative Analysis of the Topic Areas of Computer Science, Software Engineering, and Information Systems", J. Systems Software, 1992, 19:277-289

[Greening, 2000] T. Greening (ed.) Computer Science Education in the 21st Century, Springer-Verlag, 2000

[Hilburn, 1999] Thomas B. Hilburn, Iraj Hirmanpour, Soheil Khajenoori, Richard Turner and Abir Qasem, A Software Engineering Body of Knowledge Version 1.0, April 1999, Technical Report CMU/SEI-99-TR-004 ESC-TR-99-004.

[Hilburn, 1999] T. Hilburn et al., Guidance for the development of software engineering education programs, Journal of Systems and Software, vol. 49 (1999) pp. 163-169

[Parnas, 1998] D.L. Parnas, "Software Engineering Programmes are not Computer Science Programmes", IEEE Software, November/December, 1999.

ÁREA DE INNOVACIÓN DOCENTE Y EXPERIENCIAS EN LA ENSEÑANZA DE INFORMÁTICA

Responsable: Dr. Pedro Blesa
(Universitat Politécnica de Valencia)
E-mail: pblesa@dsci.upv.es
Responsable: D. Antonio Martí Campoy
(Universitat Politécnica de Valencia)
E-mail: amarti@disca.upv.es

ADAPTACIÓN CURRICULAR DE LA ASIGNATURA TEORÍA DE ALGORITMOS.

José Joaquín Aguilera García[1], José Ramón Balsas Almagro[1]

[1]*Departamento de Informática. Universidad de Jaén.*
Escuela Politécnica Superior
Avda. Madrid, nº 35. 23.071 Jaén
e-mail: {jjaguile, jrbalsas}@ujaen.es

RESUMEN: La asignatura Teoría de Algoritmos de los estudios de Ingeniería Técnica en Informática de Gestión de la Universidad de Jaén ha evolucionado desde su primera implantación en el plan de estudios de 1992. En este trabajo describimos una propuesta metodológica para dicha asignatura y exponemos las experiencias adquiridas durante sus años de docencia.

1. INTRODUCCIÓN.

La asignatura Teoría de Algoritmos pertenece al Plan de Estudios de 1997 de la Ingeniería Técnica en Informática de Gestión impartido en la Escuela Politécnica Superior de la Universidad de Jaén. Es una asignatura optativa, recomendada para el tercer curso, con una carga docente de 4.5 créditos teóricos y 3 créditos prácticos, distribuidos en el primer cuatrimestre. En los dos primeros cursos de la titulación se estudian ciertas materias que establecen conceptos previos y necesarios para la asignatura en la que se centra este trabajo.

En el antiguo Plan de Estudios de 1992, los contenidos de Teoría de Algoritmos se incluían en otra asignatura optativa con el mismo nombre, pero de 9 créditos, impartida en el segundo cuatrimestre del segundo curso.

Los objetivos de ambas asignaturas eran los de establecer las bases necesarias que permitiesen al alumno desarrollar algoritmos más eficientes, tanto en espacio (memoria consumida) como en tiempo de ejecución empleado, independientemente del campo de aplicación requerido.

La experiencia obtenida con la docencia en las asignaturas de los dos planes de estudios nos ha permitido establecer la propuesta docente que presentamos a continuación.

Para ello organizamos el trabajo de la siguiente forma. En el apartado dos se describen brevemente los objetivos de la asignatura. En los puntos tres y cuatro se ven los pasos seguidos y la situación actual a la que hemos llegado. Para finalizar, en los últimos apartados exponemos las conclusiones obtenidas, proponemos algunos trabajos futuros en proceso de realización y la bibliografía básica recomendada para la asignatura.

2. OBJETIVOS DE LA ASIGNATURA.

Los objetivos de la asignatura de Teoría de algoritmos son los siguientes:

Estudio sistemático del diseño y análisis de algoritmos.

Aprender a analizar la eficiencia de algoritmos lineales y recursivos.

Proporcionar al alumno las herramientas básicas que necesitará para desarrollar eficientemente sus propios algoritmos, sea cual fuere el campo de aplicación requerido.

Estudio de algunas técnicas fundamentales que se utilizan para diseñar algoritmos eficientes:

Algoritmos Greedy (Voraces).
Técnica Divide y Vencerás.
Algoritmos basados en Programación Dinámica.
Técnicas de Grafos. Algoritmos Backtracking.
Algoritmos Probabilistas.
Algoritmos Genéticos.

3. EVOLUCIÓN DE LA ASIGNATURA.

En el primer plan de estudios de Ingeniería Técnica en Informática de Gestión de la Universidad de Jaén (1992), la asignatura Teoría de Algoritmos se impartía en el segundo cuatrimestre del segundo curso, con una carga lectiva de 9 créditos (6 teóricos y 3 prácticos).

En el año 1997 dicho plan de estudios tuvo que reformarse para adaptarse a la normativa actualmente vigente: Real Decreto 1497/1987 y las sucesivas modificaciones dadas por el Real Decreto 1267/1994, el Real Decreto 2347/1996, el Real Decreto 614/1997 y el Real Decreto 779/1998.

Estas modificaciones afectaron a nuestra asignatura. Además, los profesores propusimos que se ofertara en un curso superior, con el fin de que el alumno se enfrentara a ella habiendo estudiado y asimilado previamente todos los conceptos necesarios relativos a estructuras de datos. Éstos se cursan en las asignaturas de Estructura de Datos y de la Información I y II, distribuidas respectivamente en el primer y segundo cuatrimestre de segundo.

Surgió así la nueva asignatura, con el mismo nombre, pero con distinta carga lectiva y en distinto curso. La nueva Teoría de Algoritmos tiene 7.5 créditos (4.5 teóricos y 3 prácticos) y se imparte en el primer cuatrimestre del tercer curso.

En el curso académico 1998/99 se implantaron simultáneamente los planes de estudios de los cursos de segundo y tercero. El número de alumnos de la asignatura fue de 18, ya que solamente se matricularon algunos repetidores y aquéllos que, estando en tercero, no la habían cursado el año anterior.

El temario teórico fue el siguiente:

Tema I: Introducción. Análisis de eficiencia de los Algoritmos.
Tema II: Algoritmos Greedy.
Tema III: Algoritmos Divide y Vencerás.
Tema IV: Algoritmos basados en Programación Dinámica.

Tema V: Algoritmos Backtracking.
Tema VI: Algoritmos Probabilistas.
Tema VII: Algoritmos Genéticos.

El temario de las prácticas fue el siguiente:

- Prácticas de Eficiencia de Algoritmos:

  Unidad I: Medición empírica de tiempos de ejecución.

- Resolución de problemas utilizando las distintas técnicas explicadas en teoría.

  Unidad II: El problema del cambio de monedas.
  Unidad III: Multiplicación de matrices: Algoritmo de Strassen.
  Unidad IV: Multiplicación de series de matrices.
  Unidad V: El juego del "tres en raya".
  Unidad VI: Búsqueda del k-ésimo menor elemento de un array.

Se mantuvo la idea original de la antigua asignatura del plan del 92. Para implementar los algoritmos los alumnos podían usar el lenguaje de programación que ellos quisiesen. No se les imponía ninguna restricción en este aspecto. Los lenguajes utilizados, como lógicamente cabría pensar, fueron Turbo Pascal y, mayoritariamente, C, ya que son los que los alumnos estudian y aprenden en las asignaturas de Metodología y Tecnología de la Programación I y II y Estructura de Datos y de la Información I y II.

Como novedad, y dado el bajo número de alumnos, las prácticas se realizaron de forma individual, pues era fácil hacer un seguimiento personalizado de cada alumno.

4. ORGANIZACIÓN DE LA NUEVA ASIGNATURA.

Después de la experiencia obtenida impartiendo varios años la asignatura del Plan del 92 y la experiencia y resultados obtenidos en el curso 98/99 se plantearon algunos cambios en la asignatura.

a) Teoría.

La primera modificación fue el cambio de orden del tema de algoritmos Greedy, con el fin de que antes de que el alumno estudiase las heurísticas Greedy, ya hubiera resuelto algunos problemas típicos de forma óptima.

Otro cambio fue adelantar el tema de algoritmos genéticos. Antes siempre se había impartido al final del cuatrimestre y se habían apreciado bastantes deficiencias en la asimilación de sus contenidos.

El nuevo temario teórico, con su planificación, quedó como sigue:

Tema I: Introducción. Análisis de Eficiencia de Algoritmos. (8 h.)
Tema II: Algoritmos Divide y Vencerás. (6 h.)
Tema III: Algoritmos basados en Programación Dinámica. (8 h.)

Tema IV: Algoritmos Greedy. (6 h.)
Tema V: Algoritmos Genéticos. (5 h.)
Tema VI: Algoritmos Backtracking. (6 h.)
Tema VII: Algoritmos Probabilístas. (6 h.)

b) Prácticas.

En la nueva organización se ha tenido en cuenta el carácter práctico de la asignatura (3 créditos). El peso de las prácticas frente a la teoría lo hemos plasmado en la ponderación de la nota final de la asignatura, asignando a la nota de prácticas un 40% del total de la misma (4 puntos). Ahora las prácticas ya no se califican como Aptas o No Aptas, con lo que se pretende motivar al alumnos para una realización mejor y más detallada.

Las prácticas se vuelven a realizar por parejas, debido al incremento del número de alumnos hasta 29. Están planificadas temporalmente para que todo el trabajo que deba realizar cada equipo pueda desarrollarse completamente en sesiones semanales de 2 horas asignadas a tal fin. No obstante el alumno tiene la posibilidad de preparar la práctica previamente a su realización utilizando el cuadernillo que se les entrega al principio del curso. En él se encuentra descrito el proceso a seguir para la realización de la práctica, así como los contenidos teóricos necesarios para su correcta realización y algunos ejemplos prácticos resueltos que les sirvan como referencia. Posteriormente aparecen descritos los problemas que se deben resolver y una lista de los contenidos que tendrán que entregar en la memoria.

Para favorecer la labor continua del alumno en el desarrollo de las prácticas se ha planificado un calendario de entrega. Cada práctica se entrega de manera individual, no dejando esto para el final del cuatrimestre. La fecha de entrega no coincide con la realización de la misma, sino que se dan 2 semanas adicionales para que el alumno disponga de tiempo suficiente.

Algunas prácticas disponen de apartados opcionales que permiten a los alumnos que lo deseen aumentar la calificación final obtenida.

La experiencia nos había demostrado que el alumno tiende a abandonar la asignatura en los primeros meses ante el esfuerzo que supone tener que entregar unas prácticas periódicamente, junto con la presión que ejerce la carga del resto de asignaturas. Además, unido a su carácter optativo y al impartirse en el primer cuatrimestre, algunos alumnos se matriculaban y, si no les convencía o se sentían agobiados, hacían alteraciones de matrícula todavía dentro de los plazos establecidos.

Para aumentar la motivación y evitar el abandono de la asignatura, la entrega de cada uno de los bloques de prácticas no es rígida y si se realiza en la fecha asignada supone una puntuación de las mismas sobre 12 puntos en lugar de sobre 10. Si la práctica no se entrega en su plazo se podrá hacerlo hasta el día del examen, pero sin bonificación.

Periódicamente se colocan en el tablón de anuncios las notas, lo que permite a los alumnos realizar un seguimiento de sus resultados y conocer los fallos cometidos (a través de las tutorías) con el fin de no volver a repetirlos en las siguientes entregas.

Por último, en vez de proponer para cada tema un sólo problema, se deja a cada grupo que escoja de entre dos posibles.

El temario de prácticas quedó de la siguiente forma:

- Prácticas de Eficiencia de Algoritmos:

  Unidad I: Medición empírica de tiempos de ejecución.

- Resolución de problemas utilizando las distintas técnicas explicadas en teoría.

  Unidad II: Multiplicación de Polinomios. Organización de Torneos.
  Unidad III: El problema del cambio de monedas. Árboles Binarios de Búsqueda Optimos.
  Unidad IV: Heurísticas Greedy. El coloreo de un grafo. El viajante de comercio.
  Unidad V: El viajante de comercio. El problema de la mochila.
  Unidad VI: El problema de la mochila. El problema de las 8 reinas.
  Unidad VII: El problema de las 8 reinas. Búsqueda del k-ésimo menor elemento de un array.

5. CONCLUSIONES Y TRABAJOS FUTUROS.

La experiencia obtenida durante el curso 99/00 nos ha permitido extraer las siguientes conclusiones:

- El cambio de orden de los temas ha proporcionado una mayor comprensión de los conceptos.
- La flexibilidad conseguida con las prácticas ha permitido al alumno adaptarse gradualmente a su desarrollo, con lo que se ha despertado el interés general y hemos conseguido un incremento importante de participación y resultados frente a cursos anteriores.
- Este hecho, junto con el anterior, ha supuesto para el alumno una reducción de la complejidad de la asignatura evitando con ello la tentación de abandonarla.
- Se ha notado un aumento notable de la asistencia a tutorías motivada por la evaluación continua de las prácticas.

Como trabajos futuros, destacar que estamos diseñando un sistema automático e interactivo que permita la publicación automática de temarios en Internet e incorpora técnicas de procesamiento de lenguaje natural para la creación y evaluación automática de exámenes.

El sistema pretende ser un complemento a las tareas de tutorización: la exposición de contenidos; la resolución de preguntas expresadas en lenguaje natural y generación automática de FAQ; la generación automática de preguntas de examen y posterior evaluación por el sistema; la retroalimentación al alumno de indicadores sobre su aprendizaje mediante la presentación de resultados de su propia evaluación y la aportación de estadísticas globales al profesor sobre el aprovechamiento de los alumnos.

6. BIBLIOGRAFÍA.

La bibliografía principal propuesta para la materia a impartir es la siguiente:

Brassard G., Bratley P. *Fundamentos de Algoritmia*. Prentice Hall. 1997.

Horowitz E., Sahni, S., Rajasekaran S. *Computer Algorithms*. Computer Science Press. 1998.

Cormen T.H., Leiserson C.E., Rivest R.L. *Introduction to Algorithms*. The MIT Press. 1992.

Joyanes L., Zahonero I. Estructura de Datos. Algoritmos, Abstracción y Objetos. MCGraw Hill. 1998.

Heileman, G. L. Estructuras de Datos, Algoritmos y Programación Orientada a Objetos. McGraw Hill. 1998.

Mark Allen Weiss. *Estructuras de Datos y Algoritmos*. Addison-Wesley Iberoamericana. 1995.

Aho A.V., Hopcroft J.E.,Ullman J.D. *Estructuras de Datos y Algoritmos*. Addison-Wesley Iberoamericana. 1988.

DISEÑO DE UN CURSO INTERACTIVO Y ADAPTATIVO DE PROCESADORES DE LENGUAJES

Manuel Alfonseca, Rosa María Carro, Estrella Pulido, Pilar Rodríguez

E.T.S. Informática. Universidad Autónoma de Madrid
e-mail: {Manuel.Alfonseca, Rosa.Carro, Estrella.Pulido,Pilar.Rodriguez}@ii.uam.es

RESUMEN: El objetivo de este trabajo es describir una experiencia de innovación docente a través de la red que se está llevando a cabo en la Escuela Técnica Superior de Informática de la U. Autónoma de Madrid para la docencia de la asignatura de Procesadores de Lenguaje. Las características principales de esta experiencia son (a) la utilización de una herramienta para la creación de cursos adaptativos y (b) la incorporación de ejercicios sobre gramáticas que requieren una evaluación compleja.

1.- INTRODUCCIÓN.

El objetivo de este trabajo es describir una experiencia de innovación docente a través de la red que se está llevando a cabo en la Escuela Técnica Superior de Informática de la U. Autónoma de Madrid. Esta experiencia consiste en la creación de un curso de Procesadores de Lenguajes que se encuentre disponible a través de Internet, de modo que pueda contribuir al aprendizaje de la materia por parte de los estudiantes, complementando la docencia impartida en las aulas. Este curso constará de explicaciones teóricas, ejemplos ilustrativos y ejercicios que el estudiante podrá realizar para comprobar su comprensión del tema objeto de estudio. En la actualidad el curso diseñado es de carácter adaptativo [1] y proporciona información a los estudiantes sobre el tipo de errores cometidos durante la realización de algunos de los ejercicios propuestos [2].

La materia de Procesadores de Lenguajes es una de las asignaturas troncales de la carrera de Ingeniería Informática y tiene por objeto la enseñanza de los distintos métodos utilizados para traducir o ejecutar programas escritos en cualquier lenguaje de programación. Estos métodos pueden utilizarse para construir aplicaciones tan diversas como los compiladores, los intérpretes o los analizadores de fórmulas. La complejidad de la materia, que requiere el dominio de diversas técnicas de análisis morfológico, sintáctico y semántico, hace que sea necesario introducir programas especializados para realizar simulaciones o ejercicios relacionados con estos temas.

La siguiente sección describe brevemente el sistema TANGOW, utilizado para la implementación del curso de Procesadores de Lenguajes. En la sección 3 se destacan las principales características del curso creado. Finalmente, se incluyen algunas conclusiones y trabajo en curso.

2.- EL SISTEMA TANGOW.

Para la creación del curso de Procesadores de Lenguajes y su distribución a través de la red, se ha utilizado el sistema TANGOW (Task-based Adaptive learNer Guidance On the Web). Este sistema permite la generación dinámica de cursos adaptativos accesibles a través de Internet, y el seguimiento de los estudiantes durante el proceso de aprendizaje [3].

La principal característica de los cursos generados con TANGOW es la adaptabilidad. Los estudiantes son guiados durante el proceso de aprendizaje, de modo que tanto los conceptos presentados a cada uno de ellos como los contenidos de las páginas HTML mostradas son seleccionados en cada momento para cada estudiante en particular. Los aspectos considerados por el sistema durante el proceso de guía son: el perfil del estudiante, las acciones realizadas previamente por el mismo durante su interacción con el curso y la estrategia de aprendizaje activa.

Un curso de estas características queda definido mediante la especificación de un conjunto de tareas docentes o unidades básicas de conocimiento y un conjunto de reglas que describen las relaciones entre las tareas docentes. Una tarea docente puede ser teórica o práctica, atómica o compuesta, y puede tener asociados fragmentos HTML con los contenidos de la tarea. Por su parte, una regla muestra cómo se descompone una tarea compuesta en subtareas. El proceso de guía consiste en seleccionar, para cada estudiante y en cada momento, el conjunto de tareas que éste puede o debe realizar, y construir las páginas HTML más apropiadas, a partir de los distintos fragmentos HTML disponibles en cada tarea, que habrán sido proporcionados por el diseñador del curso.

Pueden existir distintas reglas para mostrar la descomposición de una misma tarea, teniendo cada una de ellas una condición de activación diferente. Si éste es el caso, para comprobar si una tarea compuesta puede ser realizada por el estudiante y qué subtareas deberán ser realizadas, se evalúan las condiciones de activación de las reglas y se elige la descomposición descrita por aquella regla cuya condición de activación se satisfaga. Las condiciones de activación están relacionadas con el perfil del estudiante y sus acciones previas, lo que permite definir dinámicamente la estructura del curso para cada estudiante concreto.

La arquitectura del sistema TANGOW (figura 1) está basada en la utilización de los servicios Web para recibir las peticiones de los estudiantes y para presentar las páginas HTML durante la realización de los cursos.

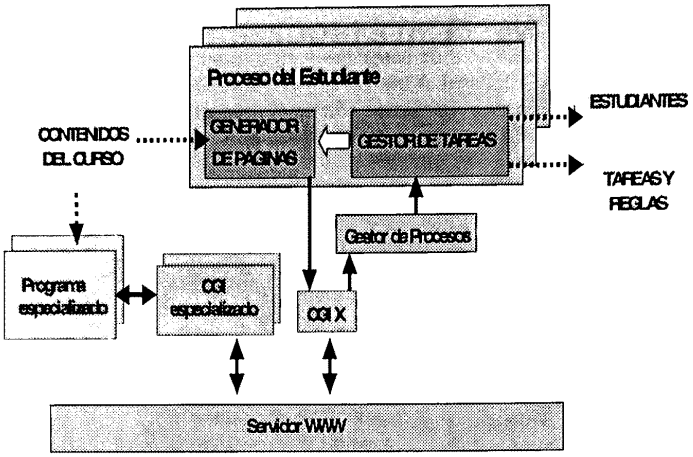


Figura 1. El sistema TANGOW.

La información necesaria para ofrecer una guía a los estudiantes durante la realización de los cursos se encuentra almacenada en las bases de datos (1) *Estudiantes*, que contiene información sobre los perfiles de los estudiantes y sus acciones en el proceso de aprendizaje, (2) *Tareas y Reglas Docentes*, que contiene las tareas y reglas docentes definidas para cada curso, y (3) *Contenidos del Curso*. En el espacio reservado a los *Contenidos del Curso* se almacenan todos los fragmentos HTML (que pueden contener textos, imágenes, vídeos, sonidos, animaciones, simulaciones, applets, etc.), a partir de los cuales se construirán las páginas HTML presentadas a los estudiantes. Estos fragmentos están clasificados según algunas características de sus contenidos, como el idioma, el nivel de dificultad o la edad del estudiante a quien van dirigidos, de modo que puedan seleccionarse los fragmentos más apropiados para cada tipo de estudiante. Pueden estar relacionados con contenidos teóricos, ejemplos o ejercicios sobre la materia a estudiar.

En la actualidad el sistema TANGOW permite incluir directamente en el curso ejercicios tipo test. En el caso de ejercicios que requieran una evaluación más compleja es posible utilizar programas especializados que se integran en el sistema a través de CGIs diseñados específicamente para cada tipo de ejercicio. Estos CGIs comprueban la solución proporcionada por el estudiante, y le envían como respuesta una página con información sobre los resultados y botones de navegación para continuar con el curso. Cuando el estudiante pulsa uno de estos botones, la información sobre los resultados obtenidos es enviada al *Gestor de Procesos*, que es único para todos los estudiantes. Este programa, junto con el *Gestor de Tareas* y el *Generador de Páginas*, constituye el mecanismo básico de ejecución del sistema.

Las peticiones realizadas por los estudiantes llegan al *Gestor de Procesos* a través del programa CGI principal, que se mantiene a la espera de las páginas HTML que deberá mostrar a los estudiantes. Por su parte, el *Gestor de Procesos* envía la petición de cada estudiante a su correspondiente *Gestor de Tareas*, que es el encargado de guiarle durante la sesión de aprendizaje. Este programa mantiene la información sobre el perfil del estudiante y las acciones que va realizando durante la ejecución del curso. En función de esta información y de la descripción del curso almacenada en la base de datos de *Tareas y Reglas*, el *Gestor de Tareas* selecciona las tareas que el estudiante puede realizar en cada momento, enviando al *Generador de Páginas* la información necesaria para que este pueda generar

dinámicamente, a partir de los fragmentos almacenados en los *Contenidos del Curso*, las páginas HTML que serán presentadas.

3.- EL CURSO DE PROCESADORES DE LENGUAJES.

Una vez considerados los aspectos más relevantes del sistema TANGOW, se van a describir algunas de las características del curso de Procesadores de Lenguajes diseñado.

a) El contenido del curso.

Como base teórica para el curso se parte de los apuntes de la asignatura, preparados por uno de los autores y desde 1996 accesibles para los alumnos en Internet en forma de diez archivos HTML, cada uno de los cuales corresponde a una de las lecciones en que se divide el curso, tal y como se imparte en la asignatura troncal de tercero de nuestro plan de estudios. Dichas lecciones son:

[1] Introducción
[2] Tablas de identificadores
[3] Análisis morfológico
[4] Análisis sintáctico
[5] Análisis semántico
[6] Generación de código
[7] Optimización de código
[8] Intérpretes
[9] Gestión de errores
[10] Gestión de memoria

En su forma inicial, cada una de estas lecciones está formada por un solo archivo que contiene texto HTML continuo, que se subdivide internamente en una serie de apartados a distintos niveles, utilizando las marcas <Hn>. Para su adaptación al sistema TANGOW, las lecciones se han subdividido en archivos más pequeños, de modo que la información contenida en cada uno de ellos se convierta en una unidad didáctica independiente.

Además de ejercicios tipo test, el curso contiene dos tipos de ejercicios especializados: ejercicios cuyo enunciado se genera automáticamente y ejercicios propuestos por el estudiante. En el primer caso, los ejercicios generados, o escogidos entre varios previamente especificados, se proponen al estudiante para que éste lo resuelva. El estudiante debe introducir el resultado obtenido, que es corregido y calificado automáticamente por el sistema. En el segundo caso, es el estudiante el que propone un enunciado relativamente libre, aunque sometido a ciertas limitaciones, para que el sistema lo resuelva y se lo explique.

Estos dos tipos de ejercicios se pueden plantear como complementarios para la práctica de un determinado tema. A continuación se presentan sendos ejemplos de ambos tipos de ejercicios, asociados a la lección correspondiente al análisis sintáctico.

b) Ejemplos de ejercicios.

El primer ejemplo corresponde a un enunciado propuesto por el sistema al estudiante: "Dada la gramática definida por las siguientes reglas de producción: `S::=aSb` y `S::=ab`, encontrar la tabla de relaciones de precedencia. ¿Es una gramática de precedencia simple? Si no lo es, explicar por qué". La figura 2 muestra la página asociada a este ejercicio.

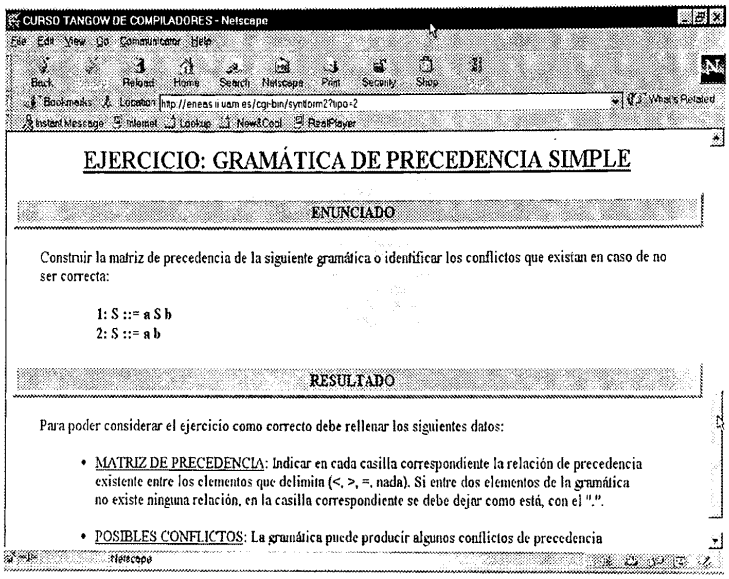


Figura 2. Gramática propuesta por el sistema.

El estudiante debe resolver el problema propuesto y escribir la solución obtenida en un formulario. El sistema espera como solución una tabla de relaciones de precedencia, así como la enumeración de los conflictos existentes, en caso de que la gramática no sea de precedencia simple. El estudiante puede escribir la matriz en formato relativamente libre, en el sentido de que el resultado de cualquier aplicación de una permutación a las filas y a las columnas será considerado válido por el sistema. La tabla que el estudiante debe proponer ha de ser equivalente a la siguiente:

	S	a	b
S			=
a	=	<	=
b			>

En un ejercicio del segundo tipo, asociado asimismo a la lección sobre análisis sintáctico, se proporciona al estudiante un formulario para que escriba la gramática que desea analizar, pudiendo elegir entre obtener la matriz de relaciones de precedencia, o bien generar la tabla del análisis sintáctico correspondiente a una gramática SLR(1), detectando en ambos casos posibles errores debidos a que la gramática no cumple las condiciones requeridas. La figura 3 muestra la pantalla asociada a este tipo de ejercicio.

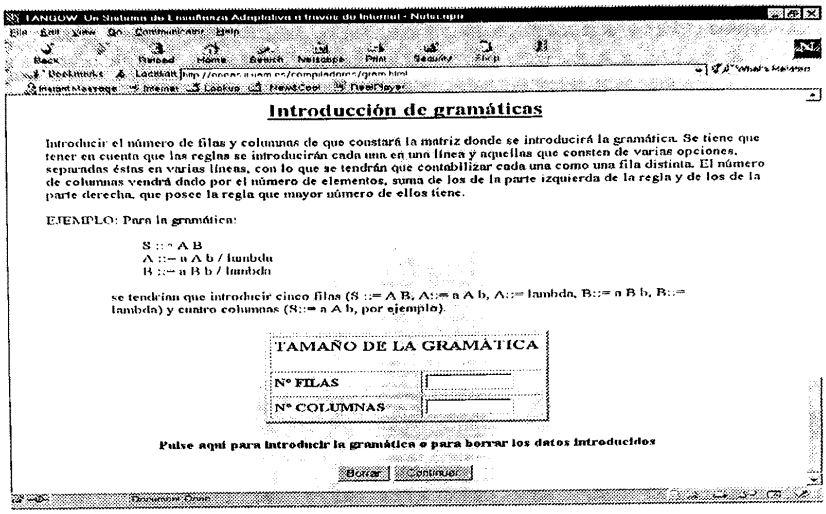


Figura 3. Gramática propuesta por el estudiante.

Para resolver ejercicios de ambos tipos se ha desarrollado un programa APL que se invoca a través de un CGI especializado, lee de un archivo los datos del ejercicio y realiza los análisis y procesos necesarios, generando como resultado un nuevo archivo que contiene la matriz de relaciones de precedencia, la tabla del análisis SLR(1), y la información necesaria para detectar errores y colisiones. Un segundo programa utiliza este archivo para generar la página HTML con las explicaciones correspondientes.

4.- CONCLUSIONES Y TRABAJO EN CURSO

En este artículo se ha presentado un curso interactivo de procesadores de lenguaje en el que los temas presentados en cada momento se adaptan al perfil y a las acciones de cada estudiante. El curso se ha desarrollado de acuerdo a la metodología de diseño basado en tareas y reglas docentes de TANGOW. Se ha hecho un especial hincapié en los ejercicios sobre gramáticas, para cuya realización se han desarrollado nuevos programas que han sido integrados en el sistema, permitiendo recopilar información sobre las acciones del estudiante durante la realización de ese tipo de ejercicios y utilizarla durante la presentación del curso.

Este curso ha sido utilizado durante la etapa final de la asignatura en el curso 1999/2000, habiendo sido valorado positivamente por los estudiantes que lo han utilizado. Durante el curso 2000/2001 se espera que el sistema sea utilizado de forma más generalizada, lo que permitirá evaluar la utilidad del curso como complemento a la docencia de la asignatura. También se incluirán simulaciones que faciliten la comprensión de algunos temas [4].

5.- REFERENCIAS

[1] P. de Bra. "Design Issues in Adaptive Web-Site Development", Second Workshop on Adaptive Systems and User Modeling on the WWW at the 7th International Conference on User Modeling. Canadá. Junio 1999, pp. 29-39.

[2] A. Patel . Kinshuk, D. Russell. "A computer based intelligent assessment system for numeric disciplines". Information Services and Use, 18(1-2). ISSN 0167-5265. 1998, pp. 53-63.

[3] R.M. Carro, E. Pulido, P. Rodríguez. "Dynamic generation of adaptive Internet-based courses", Journal of Network and Computer Applications, 22. 1999, pp. 249-257.

[4] M.Alfonseca, J.de Lara, E.Pulido. "Semiautomatic Generation of Web Courses by Means of an Object-Oriented Simulation Language", Simulation, special issue on Web-based Simulation, 73:1, Julio 1999, pp. 5-12.

ENSEÑANZA DE LA PROGRAMACIÓN BASADA EN ESQUEMAS

José Manuel Burgos Ortiz[1], Javier Galve Francés[1], Julio García Martín[1], Miguel Sutil Martín[2]

[1]*Profesores de la Facultad de Informática de la Universidad Politécnica de Madrid*
e-mail: {jmburgos,jgalve,juliog}@fi.upm.es

[2]*Profesor de Formación Profesional en Informática en el IES "Ciudad Escolar"*
e-mail: msutil@arrakis.es

RESUMEN: La mayoría de los cursos y textos de programación no resaltan el conocimiento clave de la programación ni prestan atención a las habilidades y destrezas que el experto maneja para escribir programas. Este trabajo presenta una exploración de ese conocimiento para dotarlo de estructura. Se estudia la idea de esquema de programación como unidad válida para organizar el complejo conocimiento de un primer curso de programación poniendo especial interés en las relaciones entre problema, solución y programa.

1.- MOTIVACION.

La realidad actual de los programas que realizan nuestros estudiantes nos muestra un panorama decepcionante: código ilegible, poco o mal documentado y con una fuerte tendencia a representar ideas "ingeniosas" que esconden malas decisiones de diseño. En muchos casos, el origen del problema hay que buscarlo en los primeros momentos del periodo de formación del candidato a programador.

La mayoría de los cursos y textos de programación asumen como enfoque incuestionable el de dejarse llevar por la exposición de todas las construcciones sintácticas que constituyen el lenguaje e ir así mostrando ejemplos de programas que hacen uso de ellas. En este enfoque, que denominaremos *guiado por la sintaxis*, se enseña a programar al mismo tiempo que se enseña un lenguaje de programación, con lo que se mezclan los conceptos de la computabilidad con la sintaxis del propio lenguaje. Los problemas se muestran como los ejemplos necesarios para ilustrar las construcciones sintácticas y las soluciones se presentan como programas acabados que resuelven los problemas. De este modo, esta colección de problemas deslabazados y programas de última versión se convierten en el material del que disponen los estudiantes para resolver problemas nuevos.

El presente trabajo pretende buscar el conocimiento clave de un primer curso de programación y dotarlo de una estructura que lo haga coherente y manejable. Nos basaremos en los conocimientos obtenidos por investigadores del aprendizaje, el análisis de diversos trabajos relacionados con éste y la experiencia de los autores.

2.- UN MODELO DE APRENDIZAJE DE LA PROGRAMACIÓN BASADO EN ESQUEMAS.

Los sicólogos cognitivos han establecido lo que se conoce como *teoría de esquemas* para explicar los procesos de aprendizaje [AKB79]. Según esta teoría, el conocimiento se estructura y organiza en nuestra mente mediante unidades o "trozos" de conocimiento llamados *esquemas*. Un esquema es un conjunto de conocimientos extraídos de repetidas experiencias que han sido agrupados debido a sus relaciones y su efectividad para resolver, con un aire común, diversos problemas. El aprendizaje se puede ver entonces como la construcción de conocimiento mediante la creación de nuevos esquemas o mediante la combinación de esquemas preexistentes.

Lo que distingue al experto del novicio en cualquier área del saber es que el conocimiento en la mente del experto está formado por una estructura de esquemas que los jerarquiza y ordena, permitiendo una fácil recuperación, una utilización versátil y fructífera y la atracción de otros esquemas para engranarlos y así formar entramados más amplios y ricos[CS73].

El conocimiento de los programadores expertos también ha sido caracterizado mediante el modelo cognitivo anterior como consistente en secuencias de *acciones estereotipadas*, perfiladas como abstracciones intermedias de solución. Estas acciones estereotipadas han sido tratadas, modeladas y nombradas de diversas maneras por autores distintos: *esquemas de programación* [GRI81], *plantillas* [CL92a] [CLS93], *planes* [SOL85], *soluciones enlatadas* [SOL86] o *patrones* [PRO00] [ETW96]. Diversos estudios sugieren que la representación del conocimiento en la forma de esquemas de programación son de gran ayuda para los estudiantes a la hora de escribir programas [AR85] [LD85].

3.- ORGANIZACION DE LOS ESQUEMAS.

El objetivo central de este trabajo es el de encontrar un repertorio de esquemas que reúna el rico y complejo conocimiento que se transmite en un primer curso de programación, y permita ser transmitido de manera efectiva a programadores noveles. Las investigaciones hechas sobre el modo en que se organiza en la mente el conocimiento de programación sugieren que los programadores novicios tienen representaciones más superficiales, o sintácticas, de este conocimiento, mientras que los expertos manejan estructuras más profundas, semánticas, abstractas y conceptuales. Cuanto más abstractas y conceptuales sean las categorías que se manejen, más pequeño será su número, su nivel de organización será más alto y permitirá detectar más fácilmente las similitudes profundas entre los esquemas. Otra cualidad fundamental que deben tener los esquemas es que estén cohesionados, es decir, que estén interconectados de diversas maneras para estructurar el conocimiento, pues esto ayuda extrordinariamente a asimilar, memorizar, e integrar lo aprendido, así como acceder a dicho conocimiento en nuestra mente y utilizarlo de manera más versátil y fructífera. Por último, otra característica deseable de los esquemas es que permitan y faciliten la generación de soluciones de mayor dimensión mediante la combinación de los esquemas, así como la incorporación de nuevos esquemas.

Categoría		Especificación del Problema
Solución Directa		El problema es igual o semejante a otros previamente resueltos o bien es una combinación sencilla de ellos.
Análisis de Casos		En el problema hay que considerar casos distintos. El resultado toma valores distintos en función de los datos de entrada.
Colecciones	*Acumulación*	El problema consiste en acumular una expresión aplicada a cada elemento de una colección. Se exige un recorrido de la colección.
	Búsqueda	El problema consiste en hacer una búsqueda para saber si algún elemento de una colección cumple una propiedad o para saber cuál la cumple. Se exige un recorrido de la colección.
	Extremales	El problema consiste en determinar qué elemento de una colección maximiza o minimiza una expresión. Se exige un recorrido de la colección.
	Construcción	El problema consiste en construir una colección de salida como resultado de aplicar una expresion a cada elemento de una colección de entrada. Se exige un recorrido de la colección.
Lectura y Escritura		El problema consiste en leer o escribir datos en un medio externo.

a) Una Taxonomía de Problemas.

El primer paso para conseguir la estructuración de los esquemas consiste en categorizarlos desde la perspectiva más abstracta posible: la de los problemas [BGG00b].

Los problemas anteriores son los problemas que podríamos calificar como básicos. A ellos habría que añadir otros más avanzados, como los que siguen:

Categoría	Especificación del Problema
Juntar	El problema consiste en tomar elemento a elemento de dos colecciones y generar otra colección formada por la combinación de ellas.
Separar	El problema consiste en tomar una colección y generar dos a partir de ella.
Ordenar	El problema consiste en ordenar una colección de elementos.

b) Principios, Pautas y Plantillas.

Hemos establecido los siguientes ámbitos para poder identificar y tratar separadamente los esquemas que resumen el conocimiento de cada parcela relativo a cada una de las fases del proceso de desarrollo (análisis, diseño, implementación y pruebas):

- Principios: Las premisas, enunciadas de la manera más sencilla posible. Con ellas establecemos un modelo aproximado o marco de trabajo para esta fase del desarrollo. Responden a la pregunta ¿qué tengo que tener en la cabeza a la hora de aplicar los esquemas?.

- Pautas: Los métodos, las orientaciones metodológicas que encauzan la manera de usar los esquemas. Son las que dan respuesta a la pregunta ¿cómo y cuándo aplico los esquemas?

- Plantillas: Las herramientas, los esquemas operativos que resumen el conocimiento más concreto de cada fase. Responden a la pregunta ¿qué repertorio de esquemas tengo para aplicar?

4.- ESQUEMAS DE PROGRAMACIÓN.

Los esquemas se expanden a lo largo de las etapas de desarrollo para recoger conocimiento específico de cada etapa. Hemos considerado que los *patrones de programación* son un buen modelo para expresarlos [BGG00a]. En los últimos cinco años, este campo ha tenido una fuerte expansión, extendiéndose su uso no sólo a la enseñanza de la programación [AST98] [CL99] sino a la comunidad de programadores y de ingenieros de software [GHJV95]. Su principal aportación es un vocabulario de diseño común y el fomento de la reutilización de soluciones a problemas comunes. Antes de enunciar los esquemas de cada etapa, se pueden establecer los principios y pautas generales comunes a todos ellos [GEH81].

Principio General:

- *Refinamiento Progresivo*: El problema se debe partir en subproblemas para que, una vez resueltos, por combinación, den solución al problema original.
- *Modelado con tipos y descomposición en subtipos.*

Pautas Generales:

- Los subproblemas se puedan resolver. Cada subprograma debe hacer una sola cosa.
- Un subproblema sea resoluble con el mínimo impacto posible sobre los demás.
- La solución de cada subproblema debe suponer menos esfuerzo que el problema original.
- La representación de los datos se debe posponer tanto como sea posible. Una buena pauta no siempre fácil de seguir es la de elegir la representación de los datos que haga el programa sencillo [KP78]. Agrupar y organizar los subprogramas que manejan el mismo tipo de datos

a) Esquemas de Análisis.

a.1) Principios.

- La programación es una actividad orientada a objetivos. [GRI81].
- Es necesaria una comprensión completa del problema antes de resolverlo.
- Los problemas están estructurados en una taxonomía de problemas, como "familias" de problemas.

a.2) Pautas.

- En esta etapa se debe centrar la atención sólo en lo que tiene que ver con el qué y posponer el cómo.
- Especificar los datos de entrada y de salida.
- Especificar las premisas de las que se parte y los objetivos a conseguir. Refinar y precisar las precondiciones y las poscondiciones todo lo que se pueda.
- Buscar jerarquías de datos: Las jerarquías de los datos guían las jerarquías del problema y sus subproblemas.
- Partir el problema en subproblemas y aplicar recursivamente los esquemas a cada subproblema.
- Intentar categorizar el problema en la taxonomía. Reformular las las precondiciones y las poscondiciones si es necesario para poder categorizar el problema.

a.3) Plantillas.

- *Análisis de los datos*: Una plantilla para describir los datos.
- *Problema como contrato*: Una plantilla para establecer las premisas y los objetivos del problema.

b) Esquemas de Diseño.

b.1) Principios.

- *Principio de reciclaje* [CL92b]: Se debe intentar partir siempre de plantillas preestablecidas. Se trata de ver si existe un problema *isomorfo* (análogo o semejante) que ha sido previamente resuelto para, en tal caso, delegar en él la solución por *invocación* (utilización) o *clonación* (copiar su implementación). En otros casos basta con reutilizar las ideas utilizadas para resolver problemas semejantes
- *Diseño Guiado por los Datos*: La estructura de los datos guía, en muchos casos, la estructura del programa.
- *Refinamiento iterativo*, mejoramiento progresivo o crecimiento progresivo[PAT90]: Una buena estrategia para resolver el problema consiste en simplificarlo, resolverlo y luego ampliar la solución para resolver el problema original mediante vuelta atrás y revisión.
- *Factorización*: Reemplazar expresiones que se repiten muchas veces por llamadas a un subprograma común .
- *Generalización-Especialización*: Buscar un problema más general (generalización) y luego particularizarlo o bien considerar este problema como un caso particular de otro más general que ya está resuelto (especialización). La técnica de *inmersión* se basa en esta estrategia.

b.2) Pautas.

- *Aplicación, adaptación o invento*: Si el problema ha sido categorizado aplicar el esquema de solución. Si el problema no ha sido categorizado en la taxonomía, adaptar la funcionalidad del esquema de la categoría que más se aproxime a él o intentar inventar un nuevo esquema. Si el problema no ha sido suficientemente partido en el análisis, partirlo ahora en subproblemas.
- *Combinación*: Combinar los esquemas de alguna de las siguientes maneras:
- *Composición*: A la manera de la composición de funciones de matemáticas.
- *Secuenciación*: Uno a continuación de otro.
- *Anidamiento*: Uno anidado dentro de otro.
- *Mezcla*: Aplicando los dos esquemas a la vez.

b.3) Plantillas.

<table>
<tr><th colspan="3">BASICAS</th><th colspan="3">AVANZADAS</th></tr>
<tr><th colspan="2">Plantilla</th><th>Variaciones</th><th>Plantilla</th><th colspan="2">Variaciones</th></tr>
<tr><td colspan="2" rowspan="2">Entrada-Proceso-Salida</td><td>Elemento a elemento</td><td rowspan="2">Juntar</td><td colspan="2">• Iterativa</td></tr>
<tr><td>Por Lotes</td><td colspan="2">Recursiva</td></tr>
<tr><td colspan="2" rowspan="2">Lectura y Escritura</td><td>Presentación– Lectura</td><td rowspan="2">Separar</td><td colspan="2">• Iterativa</td></tr>
<tr><td>Lectura de tipo
y Escritura de tipo</td><td colspan="2">Recursiva</td></tr>
<tr><td colspan="2">Solución Directa</td><td></td><td>Búsqueda</td><td colspan="2">Con saltos</td></tr>
<tr><td colspan="2">Análisis de Casos</td><td></td><td></td><td colspan="2">Binaria</td></tr>
<tr><td rowspan="8">Colecciones</td><td rowspan="2">Acumulación</td><td>• Iterativa</td><td rowspan="8">Ordenación</td><td>Inserción Directa</td><td>• Iterativa</td></tr>
<tr><td>Recursiva</td><td>Recursiva</td></tr>
<tr><td rowspan="2">Búsqueda</td><td>• Iterativa</td><td rowspan="2">Selección</td><td>• Iterativa</td></tr>
<tr><td>• Recursiva</td><td>Recursiva</td></tr>
<tr><td rowspan="2">Extremales</td><td>• Iterativa</td><td rowspan="2">Rápida</td><td>• Iterativa</td></tr>
<tr><td>Recursiva</td><td>Recursiva</td></tr>
<tr><td rowspan="2">Construcción</td><td>• Iterativa</td><td rowspan="2">Mezcla</td><td>• Iterativa</td></tr>
<tr><td>Recursiva</td><td>Recursiva</td></tr>
</table>

c) Esquemas de Implementación.

c.1) Principios

- El código del programa debe atenerse a la sintaxis del lenguaje de programación.
- La implementación es la expresión de la solución en el lenguaje de programación.
- *Principio de la Programación Culta* ("literate programming") [KNU84] [CL92b] [SOL86]: No basta con escribir código sintácticamente correcto. Es importante que se escriba pensando en lectores humanos, es decir, que sea fácil de leer, entender y modificar. Es importante que el código sea autocontenido, es decir, que por sí solo documente y explique al lector humano por qué el programa resuelve el problema dado.

c.2) Pautas.

- Traducir la solución al lenguaje de programación.
- Estructurar el código en piezas pequeñas.
- Permitir que el programa se pueda leer de arriba abajo.
- Usar identificadores significativos.
- Reglas de encolumnado y tipografía.

c.3) Plantillas.

Plantilla	Variaciones	Plantilla	Variaciones
Utilización	• **Constantes**	*Selección*	• **Alternativas**
	• **Tipos**		• **Rango de posibilidades**
	• **Variables**	*Iteración*	*Con contador*
	• **Funciones**		*Condicionada*
	• **Procedimientos**	*Recorrido*	• **Intervalos**
	Arrays		• **Lineal Simple(Arrays y strings)**
	Ficheros		• **Archivos**
	Punteros		• **Listas Enlazadas**

d) Esquemas de Pruebas.

d.1) Principios.

- Los problemas que han sido categorizados en la etapa de análisis conducen a un programa correcto.
- Los programas que no han sido categorizados deben ser validados mediante pruebas.

d.2) Pautas.

- Probar los posibles puntos débiles del programa usando casos típicos y extremos. Elaborar un rango de casos exhaustivo.
- Probar cada pieza del programa (subprograma) por separado, yendo de más pequeño a más grande, concentrándose en cada pieza por separado.
- Si hay estructuras de datos, probar que la construcción de las mismas sea correcta, pero separada del resto del programa.

5.- CONCLUSIONES.

En este trabajo hemos presentado una propuesta para dotar de estructura el conocimiento de un primer curso de programación y así mejorar la efectividad de la enseñanza y del aprendizaje en dicho curso. La propuesta se basa, por un lado en lo que saben los expertos y por otro en lo que se sabe acerca de cómo las personas aprenden, organizando el conocimiento en esquemas de programación divididos en principios, pautas y plantillas, específicos para cada una de las fases del desarrollo de un programa. Con estos esquemas se facilita el diseño de cursos, materiales curriculares y la creación de materiales de enseñanza complementarios. Esta propuesta no está cerrada ni acabada. Se pueden identificar otros esquemas y además, aunque muchos de los esquemas están ya expresados en forma de patrones, quedan otros por escribir.

REFERENCIAS.

[AKB79] J. R. Anderson, P. J. Kline y C. M. Beasley. A General Learning Theory and Its Application to Schema Abstraction. En G. H. Bower (ed.). *The Psychology of Learning and Motivation*, vol. 13, New York Academic Pres (1979), pp. 277-318.

[AR85] J. Anderson y B. Reiser. The Lisp Tutor. *Byte*, 10, 1985, pp. 159-175.

[AST98] O. Astrachan. Design Patterns: An Essential Component of CS Curricula, *SIGCSE Bulletin and Proceedings*, 30(1), March 1998, pp. 153-160.

[BGG00a] J.M. Burgos, J.Galve y J. García. From Problems to Programs: A Pattern Language to Go from Problem Requirements to Solution Schemas in Elementary Programming. *EuroPLoP'2000.*

[BGG00b] J.M. Burgos, J.Galve y J. García. Una Taxonomía de Problemas para la Enseñanza de la Programación, *Actas del VII Congreso Internacional de Informática en la Educación INFOREDU'2000*, Mayo 2000, La Habana (Cuba).

[CL92a] M. J. Clancy, M. C. Linn. Designing Pascal Solutions: A Case Study Approach. Computer Science Press, 1992.

[CL92b] M.J. Clancy y M.C. Linn. The Case for Case Studies of Programming Problems, *Communications of the ACM*, March 1992, 3, 3, pp. 121-132.

[CL99] M.J Clancy y M.C. Linn. Patterns and Pedagogy. Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education, 1999, 37-42.

[CLS93] M.J Clancy, M.C. Linn y P.K. Schank. Supporting Pascal Programming with an On-line Template Library and Case Sudies. *International Journal of Man-Machine Studies*, 20 (1993), pp. 1031-1048.

[CS73] W. C. Chase y H. Simon. Perception in Chess. *Cognitive Psychology*, 4 (1973), 55-81.

[ETW96] J. P. East, R. Thomas, E. Wallingford, W. Beck, J. Drake. Teaching Programming Through Patterning. *Proceedings of the 1996 Small College Computing Symposium*, April 1996, St. Cloud, Minnesota, pp. 39-45.

[GEH81] N. H. Gehani. Program Development by Stepwise Refinement and Related Topics. *BSTJ*, vol. 60, no. 3, March 1981.

[GHJV95] E. Gamma, R. Helm, R. Johnson, J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.

[GRI81] D. Gries. The Science of Programming. Springer-Verlag, Texts and Monographs in Computer Science, 1981.

[KNU84] D. E. Knuth. Literate Programming, *The Computer Journal*, 27, 1984, pp. 97-111.

[KP78] B. Kernighan, P. Plauger: The Elements of Programming Style. McGraw-Hill, 1978.

[PAT90] R. E. Pattis. A Philosophy and Example of CS-1 Programming Projects, *SIGCSE Bulletin*, vol. 22, no. 1, 1990, pp. 34-39.

[PRO00] V. K. Proulx. Programming Patterns and Design Patterns in the Introductory Computer Science Course, *Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education*,2000,80-84. http://www.ccs.neu.edu/teaching/EdGroup/

[SOL85] E. Soloway. From Problems to Programs Via Plans: The Content and Structure of Knowledge for Introductory Lisp Programming. *Journal of Educational Computing Research, 1(2)*, 1985, 157-172.

[SOL86] E. Soloway. Learning to Program = Learning to Construct Mechanisms and Explanations. *Communications of the ACM*, 29, 9 (Sept. 1986), 850-858.

LAS ESTANCIAS EN PRÁCTICAS DE LAS TITULACIONES DE INFORMÁTICA EN LA UNIVERSITAT JAUME I DE CASTELLÓ

Cristina Campos Sancho[1], Mercedes Marqués Andrés[1]

[1]*Departamento de Informática, Universitat Jaume I de Castelló.*
e-mail: camposc@uji.es, mmarques@uji.es

RESUMEN: En el presente artículo se describe el procedimiento que se sigue para la organización y seguimiento de estancias en prácticas en empresas de las titulaciones de informática de la Universitat Jaume I de Castelló. Se realiza un estudio detallado de los estancias realizadas por los alumnos durante el curso 1998/1999 y se exponen diferentes problemáticas detectadas, así como propuestas de mejora que pueden contribuir a un mayor aprovechamiento de estas estancias.

1.- INTRODUCCIÓN.

"Estancias en prácticas es una denominación que ampara las disciplinas académicamente tuteladas e integradas en los planes de estudios de la Universitat Jaume I, que consisten en la realización de prácticas mediante un sistema de permanencias regladas en organizaciones administrativas, económicas o profesionales de los sectores públicos o privados o de cualquier otra forma que acuerde el Equipo de Gobierno. Las estancias en prácticas tienen como objetivo la asimilación por parte de los estudiantes de la realidad institucional, empresarial y laboral de su entorno social en el ámbito de sus respectivas futuras profesiones." [1]

Todas las titulaciones de la Universidad Jaume I (UJI) tienen un número de créditos obligatorios dedicados a asignaturas de estancias en prácticas en empresas (EEPP), aunque la duración en horas varía entre un mínimo de 120 horas y un máximo de 450 horas dependiendo de la titulación.

La UJI ofrece dos titulaciones de informática: Ingeniería Técnica en Informática de Gestión (ITIG) e Ingeniería en Informática (II). En el primer caso existe una única asignatura de estancias en prácticas con una duración de 300 horas (30 créditos) que se ubica en tercer curso. En el segundo caso, los alumnos deben cursar dos asignaturas, una con una duración de 300 horas (30 créditos) que se ubica en tercer curso y la segunda ubicada en cuarto curso con una duración de 150 horas (15 créditos).

La UJI inició sus actividades en el curso 1991/1992, por tanto la realización de estancias en practicas en empresa de las titulaciones de informática comenzó el curso 1993/1994, cuando alumnos de ambas titulaciones cursaban su tercer año académico. Los primeros cursos fue necesario realizar un trabajo de búsqueda de empresas donde los alumnos pudiesen realizar la estancias. A medida que las titulaciones se han ido consolidando, el número de alumnos se ha estabilizado y la experiencia de profesores, alumnos y empresas ha permitido ir estableciendo normativas, rutinas en cuanto al funcionamiento y mejoras que se van incorporando paulatinamente.

Cada titulación tiene un profesor que se responsabiliza de la coordinación de la realización de las estancias de los alumnos de la misma: asignación de empresas y de profesores tutores, supervisión de la evaluación, revisión de normativas, etc.

Durante el curso 1998/1999 se preparó una normativa específica para las realización de estancias en prácticas de ITIG y de II que ha sido aprobada al inicio del presente curso académico. Esta normativa amplia algunas de las responsabilidades ya establecidas en la normativa marco, añade detalles sobre los procedimientos que se siguen en estas titulaciones y otros temas particulares como, por ejemplo, la evaluación de alumnos que ya realizan una actividad profesional o son beneficiarios de becas en empresas [2], [3].

En este informe se analizarán en particular las EEPP llevadas a cabo por los alumnos de II y de ITIG durante el curso académico 1998/1999.

2.- ANÁLISIS DE LAS SOLICITUDES DE LAS EMPRESAS

Al inicio del curso académico, la Oficina Gestora de Estancias en Prácticas (OGEP) hace llegar al profesorado que coordina las EEPP de cada titulación las solicitudes de empresas que demandan alumnos de estancias en prácticas a la Universidad.

Debido, en parte al tipo de empresas disponibles en el entorno socioeconómico de la provincia de Castellón (un sector amplio de pequeñas y medianas empresas, y pequeñas empresas de servicios informáticos), en general no puede decirse que exista una oferta variada y amplia. Aunque la demanda suele ser más elevada que el número de alumnos disponibles para realizar las estancias, estas ofertas no se ajustan normalmente al nivel y tipos de tareas que los alumnos solicitan o que, tal vez, sería de desear para los alumnos de estas titulaciones.

Cuando las empresas realizan una solicitud de alumnos de EEPP deben indicar cuántos alumnos requieren y de qué titulación, qué tipo de tareas desarrollarán y el cuatrimestre y el horario de realización de las estancias.

Las ofertas de las empresas se analizan con el fin de evaluar su adecuación para la realización de estancias. Algunas se desestiman tras hablar con los responsables de las empresas, teniendo en cuenta criterios como, por ejemplo, el tipo de tareas ofrecidas. Tras este análisis, las ofertas seleccionadas se publican para informar a los alumnos previamente a que éstos indiquen sus preferencias.

En la tabla 1 se muestra un resumen de las distintas tareas indicadas en las solicitudes de las empresas a principio de curso. Se ha contabilizado una oferta por cada alumno solicitado.

Tipos de Tareas	Ofertas
Apoyo y soporte a usuarios	3
Desarrollo Software	33
Docencia	4
Gráficos	10
Hardware, mantenimiento	5
Mantenimietno Software	6
Redes	6
Web, proyecto Infoville	15
Varios	12
No especifica	4
Total	**98**

Tabla 1

En general hay que destacar la gran demanda de tareas relacionadas con desarrollo de sistemas, bases de datos y programación agrupadas en el apartado Desarrollo Software. También se observa una significativa demanda de estancias relacionadas con el desarrollo de páginas web, que incluye el proyecto Infoville desarrollado en varios municipios de la provincia.

Aunque en principio las ofertas parecen escasas, a medida que avanza el curso nuevas empresas solicitan alumnos, o empresas que ya hicieron una solicitud inicial demandan una continuidad con más alumnos.

Respecto al perfil de alumno que demandan las empresas, puede decirse que en algunos casos se solicitan alumnos de una de las dos titulaciones de informática de la UJI sin conocer exactamente la formación o perfil particular. Cuando se realiza la asignación de alumnos a empresas se tiene en cuenta el tipo de tarea y la duración de la estancia, más que la titulación solicitada en la oferta, aunque nunca se asigna un alumno de una titulación diferente a la solicitada por la empresa sin confirmarlo previamente con los responsables de la misma.

Por último, es interesante comentar un hecho que afecta de manera evidente a la organización y realización de las estancias: es difícil hacer coincidir el momento y tipo de necesidad de las empresas, con el momento y requisitos de ubicación de los alumnos. Muchas empresas realizan solicitudes de alumnos de estancias en prácticas atendiendo a necesidades puntuales de su organización, para suplir un puesto de trabajo temporal, cubrir un punto alto de actividad de desarrollo informático, o como viene siendo muy común, para "hacer la página web de la empresa". Las ofertas llegan a lo largo del curso y no al principio, como sería de desear para poder incluirlas dentro de la organización docente del mismo. Como consecuencia algunas ofertas no se cubren porque los alumnos ya han sido asignados a principio de curso o cuatrimestre, y algunos alumnos deben asignarse a ofertas menos afines a sus preferencias teniendo en cuenta la disponibilidad existente en el momento de iniciar la estancia.

3.- ANÁLISIS DE LAS PREFERENCIAS DE LOS ALUMNOS

Como se ha mencionado anteriormente, los alumnos pueden consultar las ofertas disponibles para la realización de estancias y posteriormente deben indicar cuales son sus preferencias para la realización de las mismas.

Los alumnos pueden sugerir una empresa no ofertada para la realización de sus estancias previo acuerdo con la misma. Esta propuesta se analiza de la misma manera que el resto de ofertas y, si se considera adecuada, el alumno se asigna a la empresa que ha ofrecido. Como se observa en la tabla 2, el número de alumnos que propusieron la empresa donde realizar al estancia es muy elevado, aunque ha bajado durante el curso 1999/2000. Este hecho puede ser debido a que el funcionamiento en cursos anteriores era, normalmente, que el alumno propusiese la empresa.

Carácter de las Estancias	**Alumnos**	**%**
Alumnos asignados según sus preferencias	95	59%
Alumnos que propusieron empresa	53	33%
Alumnos con contrato laboral o beca	11	7%
No presentados	2	1%
Total	**161**	

Tabla 2

También es interesante mencionar el número de alumnos que ya están incorporados al mundo laboral mediante un contrato. En general, suelen ser alumnos que finalizaron sus estudios de ITIG y están cursando cuarto curso de II en el que tienen que realizar una segunda estancia. En este caso, los alumnos pueden desarrollar una memoria que les exime de la realización de estancias en otra empresa (siempre y cuando la actividad laboral esté enmarcada dentro del perfil exigido para la realización de estancias), puesto que, claramente, ya se cumplen los objetivos principales de las estancias.

En la tabla 3 se muestra la distribución de las preferencias de tareas indicadas por los alumnos que informaron sobre éstas para la realización de las estancias.

Preferencias respecto a las tareas	**Alumnos**
Desarrollo de Software	37
Docencia	1
Gráficos	7
Hardware, Mantenimiento	2
Redes	6
Web	12
Empresa concreta	20
Indiferente	10
TOTAL	**95**

Tabla 3

Puesto que las ofertas de las empresas ya estaban publicadas cuando los alumnos indicaron sus preferencias, algunos optaron por señalar exactamente la empresa donde realizar la estancia, este caso se ha contabilizado en el apartado Empresa concreta.

Además, los alumnos debían informar de su elección sobre el cuatrimestre, horario y población para la realización de la estancia y la posibilidad de desplazamiento. Este último aspecto es importante a la hora de asignar un alumno a una empresa, puesto que muchas de las ofertas corresponden a entidades ubicadas a cierta distancia de la universidad por lo que se asignan alumnos que disponen de vehículo, o bien aquellos que tienen su domicilio en la misma localidad de la empresa.

4.- ANÁLISIS DE LA REALIZACIÓN DE LAS ESTANCIAS

Para tutorizar la realización de cada estancia se asigna un profesor cuyas responsabilidades son establecer un primer contacto entre el alumno y la empresa, desarrollar un plan de trabajo junto con un supervisor designado por la empresa y mediar entre el alumno y la empresa sobre posibles incidencias. A partir del plan de trabajo desarrollado por el tutor y el supervisor de la empresa se puede analizar el tipo de actividades desarrolladas por los alumnos que, algunas veces, pueden variar o ampliarse respecto a las tareas que la empresa planteó en su oferta.

En la tabla 4 se muestran los tipos de tareas que desarrollaron los alumnos de II i de ITIG durante el curso 1998/1999. Para la elaboración de esta tabla no se han tenido en cuenta las estancias que corresponden a alumnos con contrato laboral o becas de la tabla 2.

Como puede observarse, más de la mitad de las estancias están dirigidas hacia actividades de desarrollo de software. El nivel del trabajo desarrollado por los alumnos, en ocasiones, no corresponde con el que sería de desear para un titulado medio o superior en informática. Es evidente que las empresas no pueden asignar a alumnos de tercer o cuarto curso de carrera, normalmente sin experiencia, tareas de más responsabilidad, y menos aún teniendo en cuenta el carácter temporal del trabajo.

Tareas realizadas por los alumnos	**Alumnos**	**%**
Atención a usuarios, instalación Software	12	8%
Desarrollo de Software	80	54%
Docencia, formación	3	2%
Gráficos	4	3%
Hardware, mantenimiento	9	6%
Redes, mantenimiento	13	9%
Web	21	14%
Otros	6	4%
Total	**148**	

Tabla 4

En cuanto al resto de requisitos de los alumnos para la realización de estancias como son cuatrimestre, horario y población, puede decirse que se cumplieron casi por completo.

5.- CONCLUSIONES

Las estancias en prácticas constituyen un punto fuerte en la formación de los alumnos de la UJI. Su correcto funcionamiento, así como la satisfacción de los resultados obtenidos por empresas y alumnos, son aspectos de gran importancia para los profesores responsables de la coordinación de las estancias. Es por ello que se intenta analizar los resultados y problemáticas detectadas para mejorar, dentro de unas posibilidades reales, aquellos aspectos que son susceptibles de mejora.

Es necesario que las empresas no sólo soliciten alumnos de estancias en prácticas de informática cuando surge una necesidad puntual, si no que se planteen este tipo de actividad como una colaboración entre la universidad y la empresa que permita enriquecer el funcionamiento y conocimientos de ambas partes. Aunque no puede decirse que son muchas, hay empresas que plantean

proyectos de diferentes niveles en los que incorporan alumnos de estancias de manera continuada y cuyos resultados para todas las partes implicadas son más satisfactorios.

Los alumnos, en ocasiones, suelen sentirse defraudados respecto al nivel de las tareas realizadas en las estancias y a la supervisión ofrecida por las empresas. Como se ha mencionado anteriormente, es evidente que las empresas no pueden asignar tareas de más responsabilidad a los futuros titulados, pero es de desear que cuando finalicen sus estudios al menos esta experiencia, y la que vayan adquiriendo con el tiempo, les sirva para acometer tareas con un nivel de responsabilidad y una necesidad de conocimientos acorde con su formación.

Otro aspecto fundamental sobre la organización y realización de las estancias es la dificultad de los alumnos para compaginar los horarios de clase y el horario de la estancia. Aunque se intenta que los alumnos de tercer curso tengan una mayor disponibilidad, sobre todo por las mañanas, es un hecho real que muy pocos alumnos cursan exclusivamente asignaturas de un curso. Se intenta que los alumnos dediquen a las estancias el tiempo e interés que se merecen y lo tengan en cuenta en el momento de matricularse de unas u otras asignaturas.

Por último, es interesante comentar que durante este último curso se está intentando mejorar todos los aspectos mencionados mediante sesiones de información a alumnos y tutores, comunicación directa con las empresas y divulgación de la normativa reguladora específica aprobada [2], [3]. Así mismo, se ha comenzado a plantear un proyecto interno de la UJI para desarrollar un sistema informático abierto a empresas, alumnos, tutores y coordinadores de estancias que mejore la gestión y la transferencia de información y resultados.

REFERENCIAS

[1] Adaptación de las normas reguladoras de estancias en prácticas de la Universitat Jaume I
http://sic.uji.es/uji/norm/practiques/estades.html

[2] Normativa y funcionamiento de la asignatura de Estancias en Prácticas de la titulación de Ingeniería Técnica en Informática de Gestión. Curso 1999/2000
http://sic.uji.es/uji/norm/practiques/normetig.html

[3] Normativa y funcionamiento de las asignaturas de Estancias en Prácticas de la titulación de Ingeniería en Informática. Curso 1999/2000
http://sic.uji.es/uji/norm/practiques/normeinf.html

UTILIZACION DE LAS TECNOLOGIAS INTERNET COMO RECURSOS DIDÁCTICOS FUNDAMENTALES PARA EL DESARROLLO DE LAS NUEVAS METODOLOGIAS DIDACTICAS PARA TELEFORMACION

José María de las Heras Montes[1]

[1]*ALI Asociación de Doctores, Licenciados e Ingenieros en Informática*
e-mail: ***secretec@ali.es***

RESUMEN: Internet está abriendo nuevos horizontes para el desarrollo de las actividades de la empresa y el comercio, pero posiblemente donde su incidencia sea mayor es en la Formación. La nueva Formación por Internet, ya bautizada por el CSIC como Teleformación, nos abre nuevos cáminos de investigación en la búsqueda de las Metodologías Didácticas mas idoneas a utilizar en cursos de teleformación. En la ponencia se expone la metodología didáctica utilizada en cursos de teleformación en Análisis y Diseño Informático impartidos por ALI.

1.- INTRODUCCIÓN.

La utilización de Internet en Formación tiene grandes atractivos, a tener en cuenta cuando desarrollamos cursos de teleformación. No obstante y frente a estos atractivos también debemos de reflexionar sobre las limitaciones que impone actualmente el propio medio, que condiciona el propio "proceso de desarrollo de la formación". Un alumno que asiste a un curso de teleformación deberá adoptar actitudes diferentes a las habituales en cursos presenciales

a) Atractivos de Internet para la Teleformación

Ofrece una "solución de conectividad universal" a bajo coste, gracias a la economía de escala que supone compartir recursos y la cada vez mayor infinidad de aplicaciones disponibles. Los alumnos que realizan los cursos pueden estar en cualquier parte del mundo. Solo precisan de un acceso a Internet y una cuenta de correo para mantener diálogo permanente con los profesores.

Su capacidad para albergar en su seno todo un inmenso "Continente de Conocimientos" que pueden representarse a través de las mas variadas combinaciones de formas: palabras, imagen, voz, vídeo, sonido, etc.

Fomenta la comunicación, cooperación y desarrollo del ser humano. Esta particularidad, es una de las grandes diferencias con la Formación Presencial. Durante todo el desarrollo de los Cursos por Internet

los Alumnos mantienen con los Profesores una tutoría personal con un dialogo permanente, en este sentido Internet nos facilita también que se mantenga una relación Alumno - Profesor muy similar a la mantenida en Clases Presenciales Particulares.

Permite una gran reducción del coste de los servicios a los que se acede. La reducción de los costes de los cursos impartidos en Teleformación puede llegar a ser superior a un 60%, en relación con el coste de los mismos cursos impartidos presencialmente.

b) Limitaciones actuales de Internet en Formación

Quienes desean asistir a cursos de teleformación, deben de tener presente que con esta nueva modalidad de formación "ellos son los protagonistas", con las grandes ventajas que supone, pero también teniendo en cuenta los riesgos. Cuando realizamos un curso por Internet, necesitamos adoptar algunas Actitudes algo diferentes a las que con frecuencia adoptamos en un curso presencial. En los párrafos siguientes comentamos estas diferencias:

Mayor motivación hacia el curso que realiza: en Internet, actualmente, no hay un contacto "cara a cara" de profesores y alumnos, como en una clase presencial y las dificultades ó dudas que se le presentan al alumno, no se resuelven al momento. Ante las dificultades que se vayan presentando su actitud debe ser la de utilizar los medios de comunicación con los profesores establecidos en el curso: aplicaciones específicas, correo electrónico, etc., que aunque no de forma instantánea, le darán las respuestas y orientaciones que precise. Pero no olvide que es usted, y no solo el profesor, quien ahora debe de hacerse las preguntas: ¿Se ha entendido?, ¿Queda alguna duda?

Mayor flexibilidad de horario pero también mayor autodisciplina: Sin duda la flexibilidad de horario que permite la formación por Internet, sobre todo para personas con otras ocupaciones, es una de sus grandes ventajas. "En Internet ningún alumno faltará a clase", si dentro de la gran flexibilidad de que dispone para establecer sus propios horarios se marca también su propia disciplina para realizar y obtener el mejor aprovechamiento del curso.

Descubrimiento Activo: El alumno debe de poner en marcha por si mismo, con la ayuda de los profesores, con los Programas y Referencias de Ayuda en Línea disponibles en los Cursos, etc. su propia búsqueda y ampliación personal de conocimientos sobre los temas que versa el curso que realiza y que "también están en Internet". Con estos medios los Alumnos podrán llegar a ampliar y profundizar en los conocimientos sobre cualquiera de las materias del curso, de una forma que es difícil de imaginar e imposible de llevar a cabo en un curso presencial. Esta capacidad que ofrece Internet para que cada alumno sea protagonista de la ampliación de sus conocimientos solo tiene el limite que cada cual quiera ponerse, fundamentalmente por razones de tiempo.

2.- METODOLOGIA DIDACTICA UTILIZADA

Los criterios que se han seguido para su desarrollo se basan en una enseñanza "asistida y orientada a los objetivos educativos", a través de excelente documentación y actividades programadas que permiten al profesor conocer y actuar sobre el "avance del alumno en el proceso de aprendizaje".

Con este nuevo planteamiento retomamos criterios que ya fueron utilizados con anterioridad en métodos de enseñanza programada, pero incorporando a la misma componentes fundamentales en el proceso de aprendizaje facilitados por las tecnologías Web, como son entre otros: Relación Alumno-Profesor, Incorporación de Experiencias, etc.

Esta Metodología didáctica orientada a Teleformación plantea la estructuración de los Cursos en Unidades Didácticas con Actividades Asociadas, basadas en:

Test de Autoevaluación: Estan orientados a una primera fase del proceso de aprendizaje y su finalidad es permitir que el alumno evalúe por si mismo el nivel de comprensión de los conceptos fundamentales de cada una de las unidades didácticas del curso.

Cuestiones Basicas: Se orientan a una segunda fase del proceso de aprendizaje, es decir la asimilación de los conceptos por el alumno. El alumno debe de responder a cuestiones relacionadas con el aprendizaje realizado. Los alumnos con frecuencia solicitan de los profesores aclaraciones a algunos de los conceptos involucrados en las cuestiones planteadas. La corrección por el profesor de las respuestas dadas por el alumno la permiten una primera evaluación del grado de asimilación de los coceptos tratados en cada unidad didáctica.

Ejercicios: Su finalidad puede ser múltiple. En el caso concreto del curso de Análisis y Diseño Informático se orientan a desarrollar en el alumno las capacidades de resolución de problemas del mundo real, siguiendo dos criterios: un primer criterio de aplicación de los propios conceptos de la unidad didáctica ya estudiada por el alumno en el mundo real, un segundo criterio basado en un supuesto amplio "caso", que el alumno resuelve durante todo el curso. La corrección por el profesor de los ejercicios realizados por el alumno le permiten evaluar no solo la capacidad del alumno para aplicar correctamente el aprendizaje realizado en el mundo real, sino tambien para descubrir las aptitudes de los alumnos en las distintas materias tratadas en el curso. Con este conocimiento del alumno, el profesor puede incidir directamente en la formación de cada alumno ayudandole a superar los temas que le resultan mas dificiles y ampliando los propios horizontes del curso en aquellos otros temas en los que muestra mayores aptitudes.

La Metodología desarrollada puede aplicarse en una amplia variedad de Aplicaciones Educativas que incluyan en su concepción: Test, Cuestiones, Ejercicios y Consultas, con tan solo ligeras variantes.

Otro aspecto importante de esta Metodología Didáctica orientada a Teleformación es su propio enriquecimiento a través de la propia experiencia en el desarrollo de los cursos. Actualmente se ha incluido un nuevo concepto de "programas de ayuda en línea" con el fin de facilitar a los alumnos: documentación, aplicaciones disponibles en la red relacionados con los contenidos de las unidades didácticas, ejemplos de ampliación, tutoriales en linea, referencias de interes a visitar, etc.

Con esta Merodología, pueden obtenerse importantes beneficios en las nuevas Aplicaciones Educativas a desarrollar, entre los que podemos destacar:

- Facilidad para el desarrollo y estructuración de los cursos,
- Utilización de una Metodología educativa probada con excelentes resultados,
- Importantes ahorros de costes, tanto en el desarrollo, como en la utilización posterior de los cursos.

3.- MAYOR APROVECHAMIENTO DE LOS CURSOS POR LOS ALUMNOS

Como la mayoría de las actividades que estamos realizando por Internet, la Teleformación está inmersa en un proceso de "mejora permanente". El propio desarrollo de Internet permite por una parte que los Cursos puedan ampliar sus contenidos, tanto con nuevos documentos creados por los profesores, como mediante referencias a documentos y programas creados por otras organizaciones empresariales y educativas. Por otra parte los desarrollos de nuevos cursos y la actualización de los existentes se benefician de los nuevos programas creados específicamente para Teleformación.

La experiencia, con los cursos ya impartidos es muy positiva, de hecho hemos constatado que la consecución de los objetivos educativos por los alumnos que han realizado los cursos por Internet es muy superior a la conseguida por los alumnos que han asistido a cursos presenciales. Nos hemos preguntado ¿El porqué?, puesto que en principio suponíamos que debería haber sido similar ó algo inferior. Hay una razón muy importante que nos puede ayuda a explicar este mejor aprovechamiento y que puede estar en "el tiempo dedicado por los alumnos para conseguir el aprendizaje".

En los cursos presenciales este tiempo es inflexible y lo establece el propio profesor, si por alguna razón se limita, o alguno de sus asistentes pierde alguna clase, la consecución de los objetivos educativos puede ponerse en peligro. En Internet esto no sucede y cada Alumno, según sus propias circunstancias y necesidades, establece el tiempo que debe dedicar. Este grado de libertad en el tiempo se convierte en un factor clave que le permite marcar su propio ritmo para conseguir tanto los objetivos, como el mejor aprovechamiento del curso.

Para ampliar detalles sobre el funcionamiento real en Internet de lo expuesto en este documento los interesados pueden visitar el Area de Formación de ALI en http://www.ali.es

AUTOMATIZACIÓN DE LA RECEPCIÓN, CLASIFICACIÓN Y RESPUESTA DE PRÁCTICAS Y TRABAJOS DE ASIGNATURAS

José Ramón Herrero[1]

[1]*Departament d'Arquitectura de Computadors*
Universitat Politècnica de Catalunya
e-mail: josepr@ac.upc.es

RESUMEN: La gestión de grandes volúmenes de prácticas o trabajos de asignaturas puede convertirse en una carga elevada para el profesor. Cualquier ayuda en su gestión puede contribuir a la reducción del trabajo burocrático que el profesor tiene que realizar. Este tiempo puede ser entonces dedicado a tareas más creativas, lo cual puede revertir en una mejora de la docencia y la calidad de vida del profesorado y alumnado.

En este artículo se presenta un software realizado para la gestión de trabajos entregados por los alumnos a través de correo electrónico. Se ha realizado un programa que actúa como filtro de mensajes y realiza la clasificación por grupos de los trabajos entregados. El programa es capaz de detectar e informar de un conjunto de errores de forma, guardando solamente aquellas entregas que cumplen con los requisitos que se hayan especificado. Se genera automáticamente un mensaje de confirmación con información que indica si el proceso de entrega ha sido realizado correctamente y el trabajo ha sido archivado o si, por el contrario, se han encontrado anomalías de algún tipo. Este mensaje se envía automáticamente al alumno que ha remitido el trabajo de forma que éste pueda actuar en consecuencia.

1.- INTRODUCCIÓN.

En los nuevos planes de estudio se intenta a menudo realizar un seguimiento continuado del trabajo y aprendizaje de los alumnos. Por este motivo existen multitud de asignaturas donde las prácticas han sido descompuestas en subetapas que se entregan semanal o quincenalmente. Del mismo modo se pueden pedir diversos trabajos de estudio a lo largo del curso. Es relativamente frecuente pues, que un alumno tenga que realizar 8 o 10 pequeñas prácticas y 3 o 4 trabajos por asignatura. Esto, conjugado con el elevado ratio de alumnos por profesor, implica la recepción por parte del profesor de muchas prácticas y trabajos para su evaluación a lo largo del curso.

La gestión de grandes volúmenes de prácticas o trabajos de asignaturas puede convertirse en una carga elevada para el profesor. Cualquier ayuda en su gestión puede contribuir a la reducción del trabajo burocrático que el profesor tiene que realizar. Este tiempo puede ser entonces dedicado a tareas más creativas y educativas lo cual puede revertir en una mejora de la docencia y la calidad de vida del profesorado y alumnado.

El uso de las nuevas tecnologías puede ayudarnos a reducir la carga asociada, como mínimo a las labores más sistemáticas, aunque también se han realizado intentos de evaluación automática para el caso de software [1, 2, 3]. Aunque lo ideal sería conseguir un proceso de evaluación totalmente automático, nuestra propuesta se centra en la parte de recepción de las prácticas, rechazo de aquellas que tengan un formato incorrecto, clasificación y archivo de las que siguen correctamente las especificaciones, y aviso al alumno confirmando o rechazando la entrega.

2.- AUTOMATIZACIÓN DEL PROCESO DE ENTREGA

A continuación se presenta el software realizado para la gestión de trabajos entregados por alumnos a través de correo electrónico. Se ha realizado un programa que actúa como filtro de mensajes y realiza la clasificación por grupos de los trabajos entregados. El programa es capaz de detectar e informar de un conjunto de errores de forma, guardando solamente aquellas entregas que cumplen con los requisitos que se hayan especificado. Se genera automáticamente un mensaje de confirmación con información que indica si el proceso de entrega ha sido realizado correctamente y el trabajo ha sido archivado o si, por el contrario, se han encontrado anomalías de algún tipo. Finalmente, este mensaje se envía automáticamente al alumno que ha remitido el trabajo de forma que éste pueda actuar en consecuencia.

a) Filtrado

El proceso de filtrado se basa en el título del mensaje, a partir de palabras clave que indican la asignatura o tipo de trabajo, el identificador del grupo y el número de trabajo.

Existe la posibilidad de especificar versiones, para hacer posibles las reentregas. Se puede forzar también a un formato concreto del contenido del fichero recibido o una cantidad de líneas mínimo.
Se puede exigir que las entregas se realicen en orden, o que empiecen por el número deseado.

El filtro evalúa todo este conjunto de restricciones y en caso de fallar alguna de ellas considera que la entrega no se ha realizado correctamente y genera un mensaje explicativo para el alumno.

b) Clasificación

Si en cambio todo ha ido bien, guarda el contenido del E-mail en un fichero. Opcionalmente, el fichero es comprimido para ahorrar espacio de disco.

Finalmente ese fichero es guardado en un subdirectorio específico para el grupo que realiza la entrega.

c) Logging

Además de guardar las entregas ordenadas por grupos, en un directorio diferente para cada uno, la herramienta guarda unos ficheros de *log* que sirven como resumen de lo que ha pasado.

Se crea un fichero de *log* global en el cual se puede observar el proceso de entrega de todos los grupos a lo largo del curso. Existe una línea para cada E-mail recibido, indicando la fecha, remitente, título del mensaje y resultado del proceso (OK o ERROR según proceda). Este fichero permite detectar rápidamente las entregas tardías.

De forma similar, se crea un fichero equivalente para cada grupo, conteniendo información de todos los E-mails recibidos de ese grupo. De esta forma se puede rápidamente ver la situación de las entregas de ese grupo.

d) Respuesta

Finalmente la herramienta envía al alumno el mensaje de respuesta generado, indicando si la entrega ha sido realizada correctamente o si por el contrario ha resultado errónea, en cuyo caso explica los motivos.

3.- EXPERIENCIA

A continuación mostraremos como ha sido nuestra experiencia con esta herramienta.

a) Contexto de aplicación

Este filtro de correo ha sido usado durante varios cuatrimestres en la asignatura Conceptos Avanzados de Sistemas Operativos de la Facultad de Informática de Barcelona. Esta asignatura tiene para el alumno 3 horas de teoría y una hora de laboratorio semanal a lo largo de 15 semanas lectivas.

Como complemento a las clases de teoría, los alumnos realizan 5 trabajos optativos de lectura y comentario de artículos. Por lo que se refiere al trabajo de laboratorio, realizan prácticas que posteriormente pueden ser finalizadas a lo largo de la semana. Se realizan un total de 10 prácticas. Durante la hora de laboratorio el profesor resuelve dudas y corrige.

En este contexto, resulta útil que el alumno entregue los trabajos y prácticas por correo electrónico. Este medio está al alcance de todos los alumnos desde multitud de lugares de trabajo y no requiere muchos recursos.

Con la entrega nos aseguramos de la realización de los trabajos y prácticas en los períodos estipulados, así como la posibilidad de corrección o revisión posterior por parte del profesor en aquellos casos en los que la supervisión en el laboratorio no haya sido posible o suficiente.

b) Evaluación

b.1) Profesorado

Antes de la existencia de esta herramienta los profesores recibían grandes cantidades de E-mails relacionados con estos trabajos y prácticas y que eran procesados manualmente. A menudo se recibían mensajes con errores lo cual significaba un trabajo adicional para el profesor.

El proceso de filtrado *inteligente* de estos mensajes, con todo lo que conlleva a significado una reducción notable del trabajo asociado a la corrección.

La clasificación de entregas por grupos permite al profesor encontrar rápidamente la información cuando se requiere. Los ficheros de *log* permiten obtener una visión global rápida de cómo esta evolucionando el curso o un grupo en particular.

Por todo esto, el nivel de satisfacción por parte de los profesores es muy grande.

b.2) Alumnado

Por lo que se refiere al alumnado, antes de existir este filtro, no se fiaban mucho de la realización de entregas por E-mail. Preguntaban a los profesores si habían recibido *su E-mail de la práctica X*, a lo cual el profesor normalmente no podía contestar de memoria. En el caso de recibir días después de la entrega un mensaje del profesor pidiendo una reentrega debido a que el mensaje que habían enviado estaba por ejemplo vacío, se daban casos en los que el alumno ya había borrado esa práctica, lo cual resultaba en momentos de nerviosismo o trabajo adicional.

Desde que existe el filtro y reciben una realimentación casi inmediata del estado correcto o incorrecto de la entrega se ha constatado que se sienten más confiados y tranquilos con este proceso.

Indirectamente, ha servido también para que se interesen por las herramientas que llevan a cabo todo este proceso, lo cual, para el caso de un estudiante de una ingeniería en informática resulta interesante e instructivo.

4.- CONCLUSIONES

Con este artículo he pretendido dar a conocer una herramienta que ha servido y está sirviendo para ayudar a los profesores a gestionar el trabajo relacionado con la entrega de prácticas y trabajos de asignatura, reduciendo por tanto el trabajo del mismo.

Esta aproximación no es ni mucho menos la única. Sin duda podría crearse una interficie para acceso por WWW que sin duda resultaría más llamativa. En nuestro caso, debido al tipo de aulas donde se realizan los laboratorios de la asignatura la aproximación usada ha sido la más viable.

Queda resuelta la parte que podríamos llamar *burocrática* de la entrega de trabajos. Sería interesante abordar la parte de evaluación automática, pero ésta es, sin duda, mucho menos sencilla y generalizable.

5.- AGRADECIMIENTOS

Agradezco al Departament d'Arquitectura de Computadors y a la Facultat d'Informàtica de Barcelona la ayuda prestada para la realización y presentación de esta ponencia.

6.- BIBLIOGRAFIA

[1] A. Soriano, et al. Sistema automático de ayuda a la evaluación práctica docente. JENUI´98.

[2] A.M. Zin and E. Foxley. Automatic Program Quality Assesment System. Proceedings of the IFIP Conference on Software Quality, India, 1991.

[3] S. Benford et al. Ceilidh: A course administration and marking system. Proc. Int. Conf. Computer Based Learning in Science, Vienna, 1993.

UNA EXPERIENCIA DE DISEÑO COOPERATIVO DE MATERIAL DOCENTE VIRTUAL EN INTERACCIÓN PERSONA-ORDENADOR

Julio Abascal, José J. Cañas, Miguel Gea, Jesús Lorés†, Manuel Ortega, Luis Alfonso Ureña, Manuel Velez

† Departamento de Informática e Ingeniería Industrial.
Universidad de Lleida, C/ Jaume II 69, Lleida
Tlf: 973.702.700 Fax: 973.702.702
e-mail:jesus@eup.udl.es

RESUMEN: En esta ponencia se presenta la iniciativa nacida en el seno de la asociación de Interacción Persona-Ordenador (AIPO) de edición de un corpus docente común en Interacción Persona-Ordenador (IPO), que pretende realizar un estudio de cómo debe ser la docencia de IPO y las contribuciones que se deben recoger desde un punto de vista interdisciplinar, utilizando el soporte virtual como medio de compartición de contenidos y como una iniciativa de promover y fomentar la difusión de IPO a toda la comunidad ibero-americana (por el idioma) de "forma libre" y abierta a contribuciones de todos los docentes interesados.

Como ejemplo se presenta la experiencia realizada del curso virtual de Introducción a la interacción Persona-Ordenador y su aplicación en la Universidad de Lleida.

Palabras clave: docencia virtual, interacción persona ordenador

1 INTRODUCCIÓN

El nacimiento de la Asociación de Interacción Persona-Ordenador (AIPO) ha supuesto un fuerte impulso a la visualización de esta disciplina como fuerza emergente dentro de la Universidad española. Como en toda disciplina universitaria hemos de considerar siempre la docencia y la investigación. La docencia, además en nuestro caso es un tema al que le hemos de dedicar un apartado importante dentro de nuestras líneas de trabajo, debido a la falta de material docente en castellano. AIPO supone pues un marco idoneo de debates y contactos sobre los aspectos docentes de la interacción. Dentro de nuestras primeras reuniones surgió la idea de intentar la realización de un corpus docente común. Una propuesta lógica por tanto a partir de estas conclusiones es el uso de Internet como soporte de esta información común. Dentro de estas inquietudes surgió la iniciativa de la confección de un curso de Introducción a la Interaccion Persona-Ordenador basado en Internet.

2 LA DOCENCIA VIRTUAL

La utilización de herramientas virtuales permite basar el proceso de aprendizaje no tan sólo en las actividades de tipo presencial que se desarrollan en el aula, sino también ofrecer un sistema de formación continua en el que en todo momento, se podrá transmitir información, material, recursos, experiencias.
Para desarrollar formación virtual de una manera efectiva es importante contar, por un lado, con una serie de capacidades técnicas en relación con la web y las herramientas interactivas que ofrece, y por otro, con capacidades pedagógicas y organizativas.
Es por lo que pretendemos utilizar esta plataforma como media para la creación de los contenidos y su utilización ubicua.

3 UNA JUSTIFICACIÓN DE LA DOCENCIA DE LA INTERACCIÓN PERSONA-ORDENADOR

El Informe de ACM/IEEE-CS joint Curricullum Task Force Computing Curriculla de 1991 [2] establece nueve áreas temáticas para cubrir la materia de la disciplina de informática.

La interacción Persona-Ordenador es una de las nueve áreas temáticas definidas en el curriculla de ACM, aspecto importante en el momento de situar esta docencia en los planes de estudio de las Universidades españolas.

En 1988 el Grupo de Interés Especial en Interacción Persona ordenador (ACM-SIGCHI) puso en marcha un grupo con el objetivo de hacer un diseño curricular[3]. La tarea de este comité fue la de redactar una serie de recomendaciones para educación en IPO. Este comité redactó el documento ACM SIGCHI Curricula for Human-Computer Interaction en 1992, que contiene una serie de recomendaciones para la realización de cursos de IPO.

Para poder cubrir todos los aspectos de la definición y los objetivos, la IPO ha de abarcar una gran cantidad de áreas diferentes, que incluyan distintos aspectos del ser humano y del ordenador: Informática (diseño e ingeniería de las interfases), psicología (teoría y aplicación de los procesos cognitivos y el análisis empírico del comportamiento de los usuarios), sociología y antropología (interacción entre tecnología, trabajo y organizaciones) y diseño industrial (productos interactivos)

Los temas que se escogieron en el curriculum de ACM se derivaron de la consideración de los aspectos interrelacionados de la interacción *Persona*-ordenador: (N) la naturaleza de la interacción, (U) Uso y contexto de los ordenadores, (H) características del ser humano, (C) Ordenadores y arquitectura de la interfase y (D) el proceso de desarrollo. También hay que tener en cuenta la (P) presentación de proyectos y la evaluación.

4 CONTENIDOS DEL CURSO

El curso actualmente consta de 13 capítulos, el primer capítulo es una introducción a la interacción Persona-Ordenador, donde se introduce la disciplina, el concepto de interfase y su usabilidad.

El segundo y tercer capítulo trata de estudiar las dos entidades que intervienen en la interacción, el ordenador con el estudio de los dispositivos y el estudio del ser humano desde el punto de vista la interacción.
Los estilos de interacción es un capítulo fundamental donde presentamos una perspectiva de los paradigmas de interacción y estudios comparativos.
A continuación presentamos un estudio de metáforas y su uso y diseño.

El soporte al usuario y las guias de estilo permiten completar la formación en modelos de interacción.

La internacionalización es un capítulo poco habitual en la bibliografía, pero que consideramos muy importante en un contexto globalizado de la interacción como el que vivimos.

El siguiente bloque corresponde al apartado de diseño, con un capítulo dedicado a la ingeniería de la interfase, otro dedicado a herramientas de desarrollo, diseño gráfico y estudio de evaluación de las interfases..

Finalmente se dedica un tema a las interfases de trabajo en grupo.

5 EL CURSO VIRTUAL

La estructura actual del curso es un sitio web ubicado en el servidor del grupo Griho de la Universidad de Lleida (http://griho.udl.es/ipo) que contiene toda la documentación del curso y las transparencias correspondientes a cada capítulo
El diseño del curso sigue una estructura de tres partes tal como se ve en la figura.

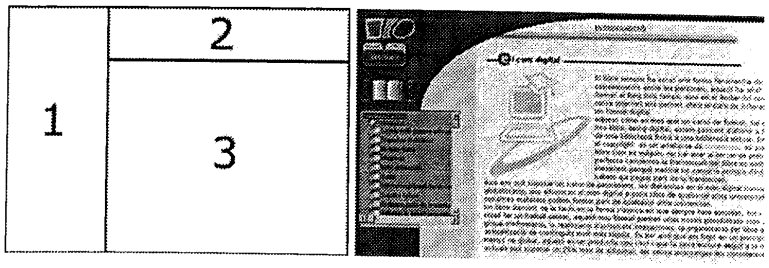


Figura 1. Diseño del libro digital

Una vez el estudiante accede al libro digital, se presenta una hoja dispuesta en tres marcos, el marco número 1 corresponde a la parte de navegación por el curso, en ella se encuentran tres diferentes niveles de navegación, el marco número 2 es el destinado a informar al usuario en donde se encuentra en cada momento y el marco número 3 es el destinado a la información del curso.

Para la realización del curso hemos estado trabajando en la confección del mismo, un colectivo de profesores de la asociación con experiencia en diversas partes del temario. Psicologia, diseño, trabajo en grupo, metodos formales, etc..

El curso permite la inclusión de documentación a traves de la modificación del indice, un fichero de tipo texto, que es visualizado por un applet de java, los contenidos han sido discutidos a través de correo electrónico y migrados al formato Html, por la facilidad que supone el uso de plantillas.

Para la redacción de los contenidos hemos utilizado los libros de Preece[4], Dix[5] y Shneiderman[6] se han utilizado realización del curso hemos utilizado, asi como abundante material de internet y de la librería digital de ACM.

6 EJEMPLOS DE APLICACIÓN. UNIVERSIDAD DE LLEIDA

El curso ha sido aplicado en la docencia presencial de la asignatura de Introducción a la interacción Persona-Ordenador de la Universidad de Lleida.

La asignatura está dividida en 2 horas teóricas y dos horas de laboratorio.
Para las clases de teoría utilizamos un aula general para todos los alumnos, debido a no poder reducir el numero de alumnos por grupo.

En estas clases utilizamos el ordenador portátil conectado a internet y el cañón proyector y un reproductor de vídeo.

Como base de la presentación se utilizan transparencias basadas en el curso digital. El esquema que seguimos es una presentación del tema, los objetivos a conseguir, una exposición de contenidos del tema y finalmente las conclusiones.

Tambíen presentamos el capítulo del libro digital y se visitan enlaces del libro que se consideren idoneos para la presentación.

Como complemento se realizan proyecciones de vídeo relativas a los temas que se presentan.

También se presenta una bibliografía y enlaces a sitios *web* visitables del tema que hemos expuesto.

Como soporte docente el alumno puede consultar el curso digital por internet o bajarlo a su ordenador para tenerlo off-line asi como las transparencias.

Tambíen se dispone de una cópia en papel, en previsión de los problemas que pueda suponer la lectura por pantalla.

7 EL PROYECTO AL COMPLETO

El libro digital es una parte de una proyecto mas completo en el que pretendemos utilizar los medios disponibles en Internet para ofrecer una posibilidad de formación completa al estudiante mediante la posibilidad de utilizar el curso digital , buscar mas información a traves de los enlaces disponibles, visionar una clase, realizar ejecercios interactivos y autoevaluarse.

8 CONCLUSIONES

La experiencia obtenida en la confección del temario la consideramos muy positiva en la ventaja que supone la creación de contenidos docentes por un grupo de profesores de diferentes universidades, expertos en aspectos determinados del temario. El uso de Internet por tanto ha permitido su realización a traves deñ uso del correo electrónico, listas de correo y transferencias de fichero. Us aspecto a considerar seria el desarrollo de una herramienta de trabajo en grupo que permitiera la actualización simultánea de materia.

La experiencia docente de uso del curso digital en los dos últimos cursos en la Universidad de Lleida ha sido muy positiva, sobre todo en el último curso por el diseño del curso digital en grupo.

No obstante es interesante comentar la necesidad todavía de imprimir el libro en papel por la dificultad de aprender toda la documentación directamente en la pantalla.
Han quedado opciones abiertas que no se han podido hacer por falta de tiempo como videconferencia de diferentes profesores para compartir la docencia con el profesor presencial.

9 REFERENCIAS

[1] Gabriel Ferraté.La universidad en el cambio de siglo. Alianza Editorial. 1998

[2] ACM/IEEE-CS joint Curricullum Task Force Computing Curriculla de 1991

[3] ACM SIGCHI Curriculla for Human-Computer Interaction. ACM 1992

[4] Preece, Jenny (1994). Human Computer Interaction. Addison and Wesley

[5] A. Dix, J: Finlay, G. Abowd, R. Beale: "Human-Computer Interaction, 2ª ed.". Prentice Hall, 1998. LSI nº 1119. ISBN: 0-13-239864-8

[6] B. Shneiderman: "Designing the user Interface: strategies for effective human-computer interaction, 3ª ed.". Addison Wesley, 1992.LSI nº 29. H5 SHN des

LA ENSEÑANZA DE LA INFORMÁTICA EN OTRAS TITULACIONES: EL CASO DE LAS FACULTADES DE CIENCIAS DEL DEPORTE E INEFS

Mercedes Macías García[1]

[1]*Profesor Asociado en la Facultad de Ciencias del Deporte de la Universidad de Extremadura.*
Área de Lenguajes y Sistemas Informáticos. Departamento de Informática.
e-mail: mmaciasg@unex.es

Resumen: El creciente papel que desempeña la informática en el mundo del deporte es espectacular. Desde la planificación y control de entrenamientos, pasando por la elaboración de estrategias o calendarios deportivos, hasta el seguimiento nutricional o de rendimiento de un deportista, todos los aspectos deportivos pueden ayudarse y potenciarse con herramientas informáticas. Es por ello que cada vez se incluyen asignaturas de informática en los planes de estudios de los futuros profesionales del deporte.

Con el presente trabajo, se analiza la presencia de asignaturas de Informática en una titulación universitaria como es la Licenciatura en Ciencias de la Actividad Física y el Deporte, y que actualmente se imparte en varias Facultades de Ciencias del Deporte e Institutos Nacionales de Educación Física o INEFs. También se verá qué departamentos de la Universidad se hacen cargo de la docencia de este tipo de asignaturas. Por último se incidirá en los objetivos perseguidos, los bloques temáticos y los contenidos de las asignaturas de informática que se vienen impartiendo en estos centros.

1.- INTRODUCCIÓN.

Actualmente, nos encontramos en plena revolución de la información, y estamos asistiendo al nacimiento de nuevas formas de trabajo y de comunicación. En definitiva, al nacimiento de una forma nueva de vivir.

Los sistemas educativos no pueden ser ajenos a estos cambios sociales. Es necesario que los asuman y los potencien. La Universidad debe ir por delante formando profesionales cada vez más cualificados, que desempeñen su labor en una sociedad donde las circunstancias no son las que eran y con toda probabilidad no serán las que son.

En al actualidad, los planes de estudio de las diferentes titulaciones universitarias hacen hincapié en la necesidad por parte de los alumnos de recibir una formación básica en Informática, que les sea útil tanto en el proceso de aprendizaje como en el futuro ejercicio de sus actividades profesionales.

Los planes de estudio han ido asumiendo las nuevas tecnologías en mayor o menor grado en toda la Universidad, bien en forma de nuevas titulaciones, de nuevas asignaturas en titulaciones ya existentes, o de reestructuración de asignaturas antiguas. Cada vez es más frecuente que asignaturas puramente teóricas de cualquier tipo, tengan créditos prácticos en Laboratorios de Informática.

2.- LA TITULACIÓN EN CIENCIAS DE LA ACTIVIDAD FÍSICA Y EL DEPORTE.

En España, se imparte la Licenciatura en Ciencias del Deporte en varias Facultades de distintas Universidades, y la Diplomatura en distintos Institutos Nacionales de Educación Física o INEFs.

Tradicionalmente, la principal salida profesional de esta titulación ha venido siendo la enseñanza, sin embargo, últimamente ha cambiado mucho el concepto de Educación Física. Se ha producido un fuerte crecimiento y diversificación en el mundo de las Ciencias aplicadas al deporte, y han surgido nuevas salidas profesionales, entre las que destacan fundamentalmente el entrenamiento y alto rendimiento deportivo, la planificación y gestión deportiva, el deporte aplicado a la salud y calidad de vida o el deporte como actividad recreativa.

Los planes de estudio se están adaptando a estos cambios que demanda la sociedad y actualmente en todas las titulaciones de ciencias del deporte conviven planes de estudios antiguos y nuevos.

a) Asignaturas de Informática.

Se han incorporado asignaturas de informática en la mayoría de las Facultades e Inefs, sobre todo en los planes de estudio nuevos y en mayor grado en aquellos centros que ofrecen itinerarios de Gestión y Recreación Deportiva así como de Entrenamiento y Alto Rendimiento. En general, las asignaturas de informática suelen ser de Fundamentos de Informática y vienen impartiéndose frecuentemente en los primeros cursos.

El gráfico siguiente representa la situación de las asignaturas de informática en todos los centros que actualmente imparten la Licenciatura en Ciencias de la Actividad Física y el Deporte. Asímismo, se diferencia el número de créditos dedicados a estas asignaturas y su carácter.

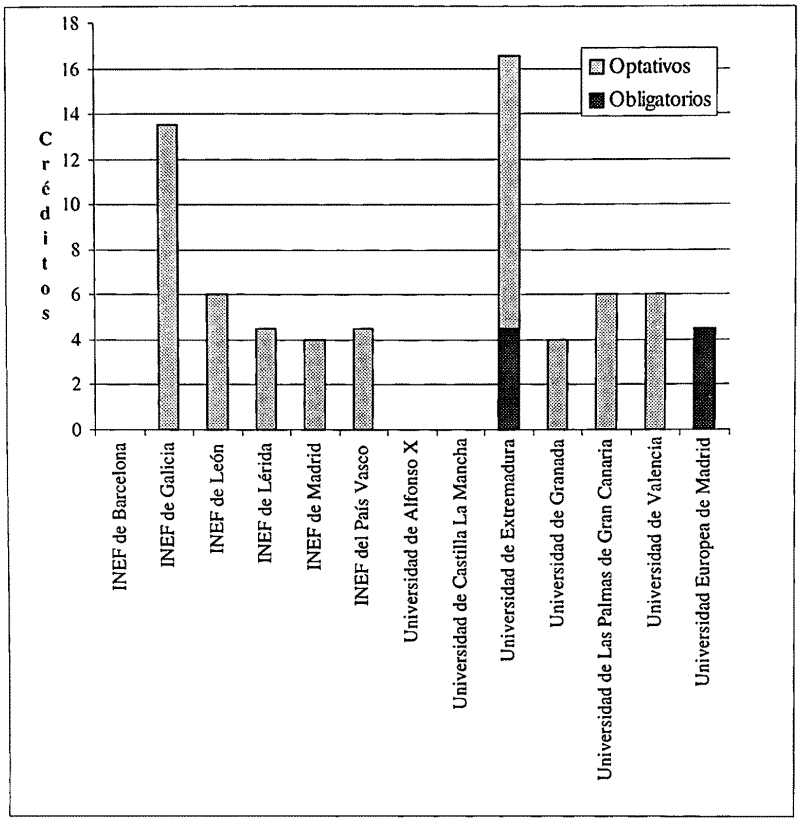


Figura 1. Créditos de Informática asignados en los distintos centros.

Se observa que la presencia de asignaturas de informática durante el curso 1999/2000 es mayoritaria y que en general, todas estas asignaturas tienen carácter optativo.

Hemos de decir, que en la mayoría de los centros, se están incorporando paulatinamente los nuevos planes de estudio, por lo cual, en muchas de ellos conviven asignaturas en extinción, pertenecientes a planes de estudio antiguos con asignaturas de planes de estudio nuevos. Será interesante observar la evolución de estas asignaturas en los próximos años.

También diremos que estas asignaturas despiertan un gran interés en los estudiantes, y tienen un número de matriculados muy alto.

b) Departamentos responsables de la docencia.

En cuanto a la asignación de estas asignaturas a los diferentes departamentos de la Universidad, se muestra la siguiente tabla:

CENTRO	DEPARTAMENTO
INEF de Lérida	El propio INEF. Universidad de Barcelona
INEF de La Coruña	Universidad de La Coruña
INEF de León	**El propio INEF** Universidad de Castilla-León
INEF de Madrid	Estructuras y análisis del movimiento humano y de las técnicas deportivas Universidad Politécnica de Madrid
INEF del País Vasco	Gestión Universidad del País Vasco
FCD de Cáceres	Informática Universidad de Extremadura
FCD de Granada	Educación Física y Deporte Universidad de Granada
FCD de Las Palmas	Informática y Sistemas Universidad de Las Palmas de Gran Canaria
FCD de Valencia	Estadística e Investigación Operativa Universidad de Valencia
Universidad Europea de Madrid	Informática

Tabla 1. Departamentos responsables de las asignaturas de Informática.

Se observa que existe una gran disparidad en la asignación de departamentos, yendo desde el departamento de Informática en algunas Universidades, hasta la situación de que el propio INEF constituye un Departamento de la Universidad a la que se adscribe.

Queremos remarcar expresamente el hecho de que tan sólo en tres centros, el departamento responsable de la docencia de las asignaturas de informática es el departamento de Informática. Tal vez debamos plantearnos este hecho que es notable y sorprendente en esta titulación pero que ocurre también en otras.

3.- FACULTAD DE CIENCIAS DE LA ACTIVIDAD FÍSICA Y DEPORTES DE LA UNIVERSIDAD DE EXTREMADURA.

En la elaboración de los temarios de las asignaturas de Informática algunas dificultades añadidas, tales como la formación académica previa del alumnado, completamente heterogénea, el número de créditos asignados, así como la dificultad de equipar y actualizar los Laboratorios de Informática con el software y el hardware más novedoso, obligan a rehacer los temarios anualmente, en función de los alumnos y de la equipación disponible. La propia naturaleza de la materia obliga a su vez a no perder de vista el avance de las nuevas tecnologías en el deporte para realimentar todo este sistema.

a) Asignaturas de Informática.

En la Facultad de Ciencias del Deporte de la Universidad de Extremadura, se imparten las siguientes asignaturas de informática:

ASIGNATURA	CARÁCTER	CRÉDITOS
Fundamentos de Informática	Obligatorio	4,5
Informática y comunicaciones	Optativo	6
Gestión informática de datos deportivos	Optativo	6

Tabla 2. Asignaturas de Informática de la FCD de la UNEX.

Al estudiar los programas de las asignaturas de informática impartidos en los distintos centros hemos observado que el 85% de los contenidos podría enmarcarse en una asignatura genérica de Fundamentos de Informática como la impartida en la Facultad de Ciencias del Deporte de la Universidad de Extremadura.

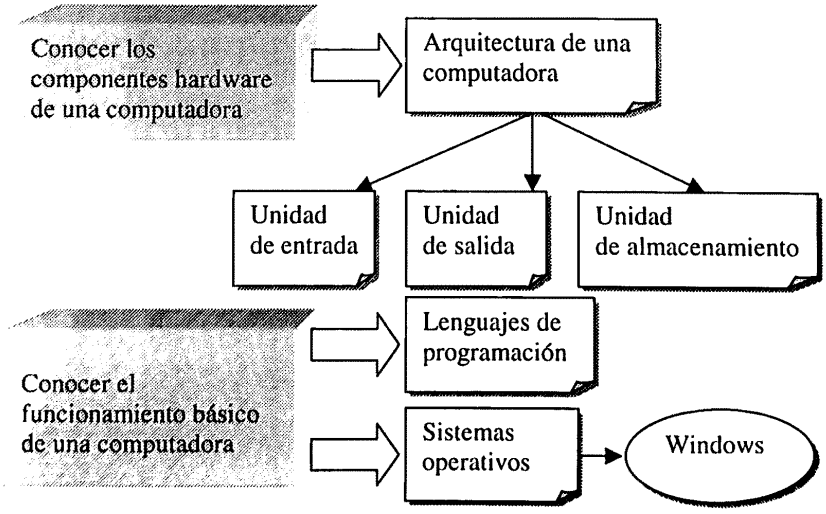
b) Objetivos de estas asignaturas.

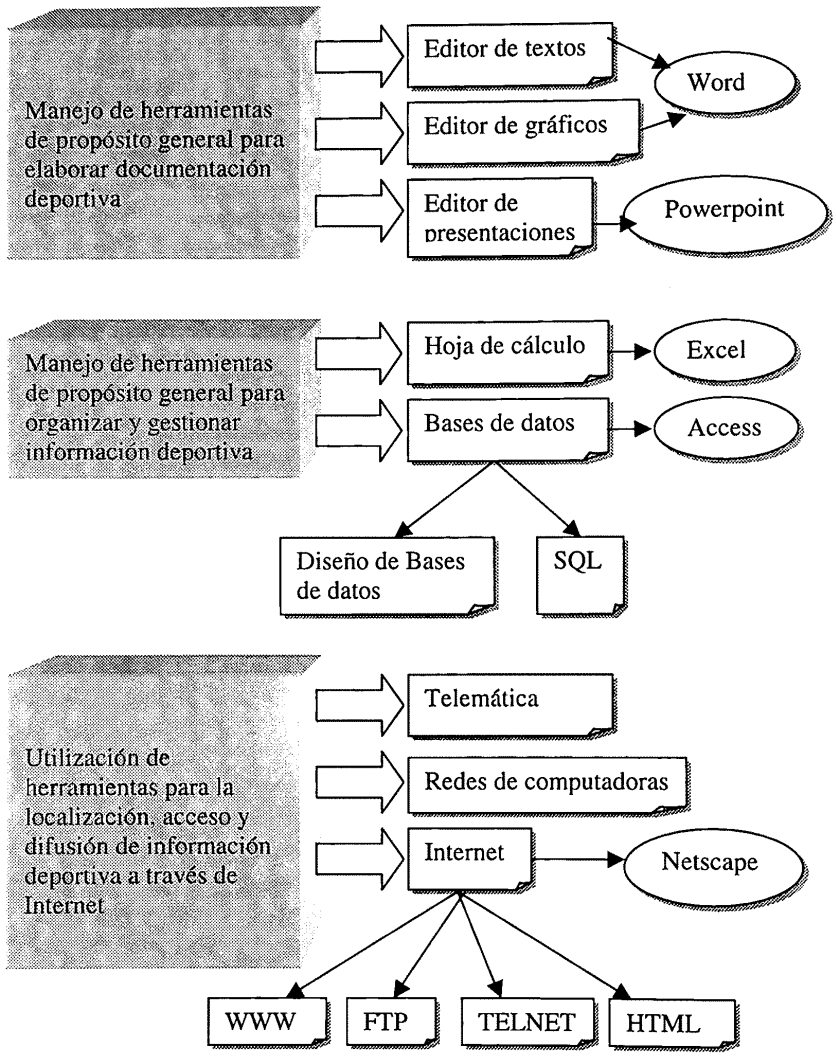
Haciendo una síntesis de los distintos programas de asignaturas de Informática impartidos en esta titulación, se puede decir que los objetivos perseguidos son los siguientes:

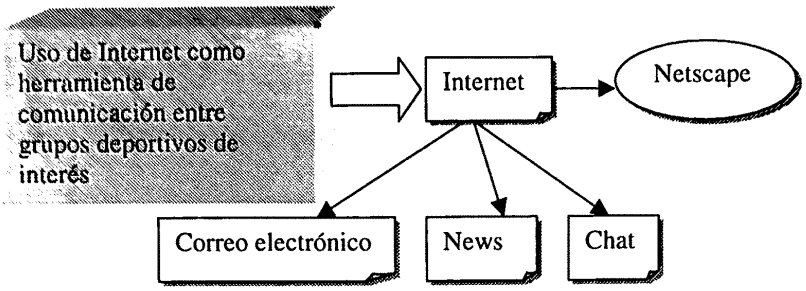
- Conocer los componentes hardware de una computadora.
- Conocer el funcionamiento básico de la máquina de tratamiento automático de la información.
- Manejo de herramientas de propósito general para elaborar documentación deportiva.
- Manejo de herramientas de propósito general para organizar y gestionar datos deportivos.
- Utilización de herramientas para la localización, acceso y difusión de información a través de Internet.
- Uso de Internet como herramienta de comunicación entre grupos de interés.

c) Bloques temáticos.

Representaremos gráficamente los distintos bloques temáticos que surgen a partir de los objetivos propuestos, así como los temas teóricos y prácticos que actualmente se imparten para llevarlos a cabo.




Manejo de herramientas de propósito general para elaborar documentación deportiva
Editor de textos
Word
Editor de gráficos
Editor de presentaciones
Powerpoint
Manejo de herramientas de propósito general para organizar y gestionar información deportiva
Hoja de cálculo
Excel
Bases de datos
Access
Diseño de Bases de datos
SQL
Utilización de herramientas para la localización, acceso y difusión de información deportiva a través de Internet
Telemática
Redes de computadoras
Internet
Netscape
WWW
FTP
TELNET
HTML



No hemos incluido aplicaciones específicas como Sistemas de Medida, Sistemas de Entrenamiento, herramientas Estadísticas o de Simulación. Estas aplicaciones específicas se suelen impartir en las asignaturas que las utilizan como herramienta de trabajo.

4. CONCLUSIONES.

El avance de las nuevas tecnologías, se va recogiendo en los planes de estudio, tanto en forma de nuevas asignaturas, como de nuevos contenidos. En la mayoría de los centros que imparten docencia en Facultades de Ciencias del Deporte e Inefs, las asignaturas de informática son aún escasas, aunque hemos observado que van en aumento tanto en número como en créditos dedicados a ellas. De momento, estas asignaturas son de muy reciente incorporación, tienen muy pocos años de rodaje y están en plena evolución y readaptación. Además el alumnado tiene un interés creciente en ellas, debido a su indudable aplicación al entorno de trabajo como futuros profesionales de la Educación Física y del Deporte.

5. REFERENCIAS.

- INEF de Cataluña. http://wwwinefc.es/
- INEF de Galicia. Universidad de La Coruña. http://www.udc.es/inef.
- INEF de León. Universidad de Castilla-León. http://www.unileon.es/
- INEF de Madrid. http://www.inef.com
- FCD Univ. Castilla-La Mancha. http://www.uclm.es/to/cdeporte.
- FCD Univ. Extremadura. http://www.unex.es/ccdeporte
- FCD Univ. Granada. http://deporte.ugr.es
- FCD Univ. Las Palmas de Gran Canaria. http://azuaje.ulpgc.es
- FCD Univ.Valencia. http://www.uv.es/~esport
- FCD Univ.Europea de Madrid http://www.uem.es/facultad/deporte.html
- Universidad de Alfonso X. http://www.uax.es/uax/.

LA GEOMETRÍA EN LA INFORMÁTICA

Manuel Abellanas*
Universidad Politécnica de Madrid
Mabellanas@fi.upm.es

RESUMEN: La Geometría Algorítmica trata problemas geométricos de forma constructiva. Actualmente ha cobrado un gran interés debido a las numerosas aplicaciones que tiene en la Informática. En este trabajo se hace una introducción a la disciplina indicando su interés en Informática y aportando enlaces de interés para la docencia y la investigación en la materia.

1.- INTRODUCCIÓN

La reconstrucción de las lindes de los terrenos en las orillas del Nilo tras las crecidas periódicas de éste fue uno de los motivos del desarrollo de la Geometría en el antiguo Egipto. En tiempos de Euclides la Geometría era considerada uno de los pilares de la ciencia. Especialmente, por lo que tenía de aplicada a los problemas que la vida ordinaria iba originando y para los que demandaba soluciones prácticas. A lo largo de la historia ha recibido más o menos atención dentro de la Matemática. En el último siglo el desarrollo de la Topología y de los espacios abstractos influyó también en la Geometría, primando el interés por la generalización y el estudio de propiedades existenciales no constructivas. Gracias al desarrollo de la Informática han aparecido numerosos problemas geométricos cuya solución es necesaria para resolver problemas informáticos. En algunos casos ha sido suficiente desempolvar problemas ya conocidos y sus soluciones. En otros muchos casos ha sido necesario hacer una reformulación de los problemas y buscar nuevas soluciones adecuadas a las demandas de la Informática. Con todo ello, no ha sido suficiente y han surgido, además, nuevos problemas geométricos al tratar de dar respuesta a problemas informáticos. Esto ha enriquecido la Geometría y está dando trabajo a informáticos y matemáticos interesados por problemas aplicados.

2.- PROBLEMAS GEOMÉTRICOS EN LA INFORMÁTICA

La Matemática es la herramienta que emplean numerosos campos técnicos y científicos para formalizar problemas y buscar soluciones. La Informática no es una excepción. Esto mismo ocurre con el caso particular de la Geometría. En ámbitos bien diferentes surgen problemas geométricos. En este apartado vamos a mostrar algunos ejemplos que surgen en el ámbito de la Informática.

Bases de datos. En el acceso compartido a bases de datos surgen problemas cuando varios usuarios están autorizados a modificar los datos. Mientras un usuario accede a una variable, ésta debe estar bloqueada al resto para evitar una lectura errónea mientras el primero modifica su valor. Si asignamos a cada usuario de una base de datos un parámetro que represente el tiempo de acceso a dicha base, los períodos que el usuario bloquea cada variable quedarán representados como intervalos. Así mismo, en el espacio producto, el producto de los intervalos

* Parcialmente subvencionado por DGES-MEC-PB98-0933

de bloqueo correspondientes a los diferentes usuarios define una región poliédrica que determina la región prohibida. Esto significa que los puntos de dicha región representan puntos de posible conflicto entre usuarios. El análisis de conflictos en el acceso a la base de datos se corresponde con el análisis geométrico de dicha región. Si el número de variables es elevado, como ocurre en la práctica, o el número de usuarios es elevado, la región tiene una elevada complejidad. Esto hace necesario el empleo de algoritmos eficientes para el análisis.

Reconocimiento de patrones. En reconocimiento de patrones, ya sea en archivos sonoros, de imagen, etc., el problema consiste en identificar o reconocer formas o patrones conocidos. Se han desarrollado numerosas técnicas en Geometría Algorítmica que sirven de soporte para ello. Desde las más simples como las envolventes convexas, a otras más sofisticadas como las envolventes alfa y diferentes métodos de agrupación de datos.

Clasificación. Relacionado con el tema anterior, los problemas de clasificación pretenden, en general, encontrar métodos automáticos para clasificar nuevos datos a partir de una colección de datos previamente conocidos. En numerosas ocasiones el problema se reduce a localizar el nuevo dato, empleando diferentes criterios de proximidad, en una subdivisión del espacio de los datos generada por los datos iniciales. Los diagramas de Voronoi[7] en sus diferentes versiones son una herramienta útil para ello.

Optimización de redes. El diseño de redes de comunicaciones también da lugar a problemas de optimización geométrica. Las triangulaciones de Deloné se emplean a menudo para diseñar redes en las que nodos próximos estén conectados entre sí. Una triangulación de una nube de puntos del plano es un grafo cuyos nodos son los puntos y sus aristas son segmentos rectilíneos que los conectan, de modo que dos de ellas no se atraviesan, siendo el número de aristas maximal (lo mayor posible). La triangulación es de Deloné si el menor de los ángulos de todos los triángulos es lo mayor posible. Esta propiedad hace que las aristas conecten puntos entre sí si son próximos. Un subgrafo importante de la triangulación de Deloné es el arbol de expansión mínimo (que es la red que conecta los nodos con la menor cantidad de cable).

Diseño de circuitos. En el diseño de las máscaras que intervienen en un circuito se emplean generalmente regiones determinadas por rectángulos. El número de rectángulos que intervienen es muy elevado. Una de las tareas importantes consiste en verificar si el diseño cumple las condiciones necesarias para su correcto funcionamiento. Algunas de ellas corresponden a condiciones geométricas, en particular de distancias mínimas entre las regiones determinadas por la unión de los rectángulos que definen las diferentes máscaras. También aquí se hace imprescindible disponer de algoritmos eficientes para el análisis de objetos geométricos.

Síntesis de imágenes. La síntesis de imágenes es una de las ramas más activas de la Informática Gráfica. En la mayor parte de los casos, las imágenes se forman a partir de un gran número de elementos geométricos sencillos. El problema consiste en estructurarlos adecuadamente para que las operaciones que posteriormente se desean realizar sobre ellos se puedan hacer empleando la menor cantidad de recursos posible. La Geometría Algorítmica aporta tanto estructuras de datos como algoritmos adecuados para la manipulación de dicha información geométrica.

[7] En su versión eucídea plana, el diagrama de Voronoi de n puntos del plano es la descomposición del plano en las regiones de proximidad de los puntos. La región de proximidad de un punto es el conjunto de puntos que son más próximos a él que a cualquiera de los demás.

Existen otras muchas áreas relacionadas con la Informática, o que dependen esencialmente de ella, en las que aparecen problemas geométricos. Por ejemplo en planificación de trayectorias en Robótica o en Cartografía Automática entre otras.

3.- LA GEOMETRÍA ALGORÍTMICA

La Geometría Algorítmica se ocupa de resolver problemas geométricos de modo constructivo. No sólo se interesa por la existencia de la solución de un problema, sino también por encontrar el método (algoritmo) para construir dicha solución. Además, y esto es propio del diseño de software en general, se ocupa de buscar algoritmos eficientes. Es decir, que requieran pocos recursos. El volumen de datos que se maneja en muchas aplicaciones es muy elevado, lo que obliga a buscar no una solución, sino una buena solución algorítmica para el problema. Los dos parámetros que miden generalmente la complejidad de un algoritmo son el espacio y el tiempo. El espacio es la cantidad de memoria necesaria para almacenar tanto los datos de entrada del problema, como los datos que se generan durante el proceso del algoritmo, así como los de la solución. El tiempo depende directamente del número de operaciones básicas que el algoritmo efectúa al resolver el problema. La forma de medir el tiempo para no depender de la máquina empleada consiste precisamente en medir el número de operaciones básicas.

La Geometría Algorítmica, por tanto, forma parte de la teoría del diseño y análisis de algoritmos y estructuras de datos. En ocasiones la Geometría es quien abre las puertas a soluciones más eficientes en problemas que no parecen geométricos. Descubrir que los datos del problema verifican una propiedad geométrica sirve para poder aplicar alguna técnica algorítmica o alguna estructura de datos que agiliza el proceso de los mismos.

Mostraremos, mediante un ejemplo, de qué manera puede surgir la Geometría y cuál es el estilo de la Geometría Algorítmica.

Análisis de rangos en bases de datos. Un problema que se presenta a menudo en el análisis de bases de datos es el estudio de los rangos. Si los datos de la base de datos son uniparamétricos, el rango es la diferencia entre el mayor y el menor. Generalmente los datos están formados por más de un parámetro. Un primer análisis de los rangos consiste en calcular el rango en cada parámetro por separado. Este análisis, sin embargo, da una idea bastante pobre sobre la variación de los datos. Existen varias alternativas para este problema. Veamos varias de ellas planteando el problema en dimensión dos para una mayor sencillez. Cada dato compuesto por dos valores paramétricos puede asociarse a un punto del plano euclídeo, tomando dichos valores como las coordenadas cartesianas del punto. Calcular los rangos en cada coordenada por separado equivale a calcular el menor rectángulo isotético (de lados paralelos a los ejes) que contiene a los datos. En la figura 1 se muestran varias alternativas a esta solución.

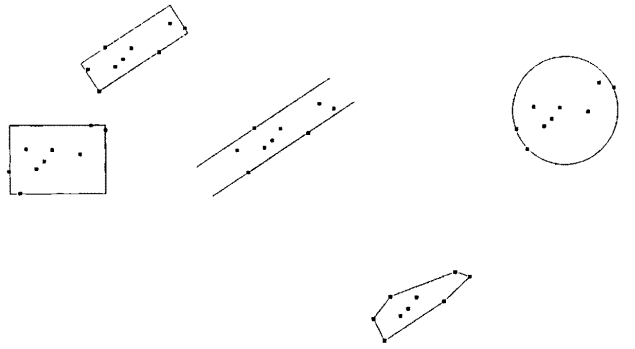


Figura 1: Diferentes alternativas para medir el rango de los datos: Rectángulo isotético, franja de anchura mínima, círculo mínimo, rectángulo libre mínimo, envolvente convexa

Cada una de las propuestas de la figura tiene asociado un problema geométrico diferente. De hecho, aún los que parecen más similares no lo son. El cálculo del menor rectángulo isotético se puede realizar en un tiempo proporcional al número de datos, mientras que el rectángulo libre no es posible calcularlo, en el peor caso, en menos de[8] O(n *log* n). El cálculo de la envolvente convexa, que es el polígono de menor perímetro que contiene a todos los puntos, se puede hacer en un tiempo de orden similar, O(n *log* n). Por otra parte, una vez conocida la envolvente convexa se pueden resolver los otros problemas en tiempo lineal O(n), por lo que todos ellos se pueden resolver en tiempo O(n *log* n). Para ello hay que resolver nuevos problemas geométricos: dado un polígono convexo, hallar el menor rectángulo que lo contiene, el menor círculo que lo contiene, o la menor franja que lo contiene.

Para estar seguros de que no existe una solución mejor para un problema, es preciso obtener una cota inferior para el tiempo de los algoritmos que lo resuelven. Este es uno de los aspectos teóricos de la algorítmica que, en ocasiones, emplea técnicas matemáticas sofisticadas. En otros casos la técnica consiste en aprovechar las cotas inferiores conocidas para otros problemas. Una cota inferior de un problema A se puede transferir a otro B cuando los algoritmos que resuelven el problema B sirven para resolver el problema A y, siempre y cuando la transformación de los datos de entrada del problema A en datos para el problema B, más la transformación de la solución del problema B en la solución del problema A, no supere a la cota inferior que queremos transferir. Un ejemplo sencillo consiste en transferir la conocida cota inferior O(n *log* n) para la ordenación de una lista de números en cota inferior para el cálculo de la envolvente convexa. Si suponemos los números de la lista positivos, transformando cada número x en el punto (x,x^2), se obtienen puntos sobre la parábola $y = x^2$, y, por tanto, en posición convexa. Esto significa que todos ellos aparecen como vértices de la envolvente convexa. Una vez calculada ésta, basta recorrer sus vértices, empezando por el más bajo, para obtener sus abscisas ordenadas de menor a mayor. El resultado es la lista de números ordenada. En consecuencia, no es posible calcular la envolvente convexa, en el peor caso, en menos tiempo, pues ésto llevaría a conseguir ordenar una lista de números en un tiempo imposible.

De los problemas planteados, el primero, calcular el menor rectángulo isotético, ya hemos dicho que puede resolverse en tiempo lineal, y esto es a su vez una cota inferior obvia, pues es preciso procesar cada punto para estar seguro que ninguno queda fuera del rectángulo solución.

[8] O(n log n) es la notación estandar de Knuth empleada para medir la complejidad asintótica de los algoritmos. Una función f (n) se dice que es de orden O(g(n)) si a partir de un cierto valor de n y para un cierto valor entero positivo k se verifica que $f(n) < k.g(n)$

La franja mínima, el rectángulo libre mínimo y la envolvente convexa se pueden calcular como se ha dicho en tiempo O(n *log* n). Los tres problemas tienen como cota inferior una función de ese mismo orden. Sin embargo, el cálculo del menor círculo contenedor de los puntos puede mejorarse empleando técnicas de poda y búsqueda más sofisticadas. Es posible con esa técnica resolver el problema en tiempo lineal.[9]

4.- LA GEOMETRÍA ALGORÍTMICA EN LOS ESTUDIOS DE INFORMÁTICA

La Geometría Algorítmica, en general bajo el título de Geometría Computacional, ha pasado a formar parte de los planes de estudio de las Ingenierías Informáticas. Esta asignatura, por una parte, complementa las asignaturas de programación y estructuras de datos, y, por otra, sirve de soporte a asignaturas específicas relacionadas con la informática gráfica y con la robótica. En ella se estudian técnicas algorítmicas así como estructuras de datos geométricos avanzadas que permiten el proceso más eficiente de datos geométricos. Como muestra de la afirmación anterior, se recogen a continuación algunas direcciones de Internet con información sobre cursos concretos. En ellas es posible encontrar los programas de las asignaturas e información sobre Geometría Algorítmica:

- Curso de Geometría Computacional en la universidad de McGill (Montreal, Canada): CS-507A Computational Geometry. Detalles en la dirección (http://cgm.cs.mcgill.ca/~godfried/teaching/cg-calendar-99.html). Es interesante visitar la página del profesor Godfried Toussaint (http://cgm.cs.mcgill.ca/~godfried/page.html), uno de los pioneros de la Geometría Algorítmica.

- Curso de Geometría Computacional en la Universidad Politécnica de Cataluña (http://www-fib.upc.es/NovaGuia/Assignatures/GEOC.html).

- Curso de Geometría Computacional en la Universidad de Sevilla (http://ma1.fie.us.es/miembros/almar/docencia/cg-info.htm)

- Curso de Geometría Computacional en la Escuela Universitaria de Informática de la Universidad Politécnica de Madrid (http://www.eui.upm.es/jesteinv/asign/geo_com.htm).

- Curso de Geometría Computacional en la Facultad de Informática de la U.P.M. (http://www.dma.fi.upm.es/docencia/segundociclo/geomcomp/home.htm).

Los libros de O'Rourke, M. De Berg et al., Preparata, Boissonat, entre otros, son textos habituales en todas ellas. (Véase la bibliografía).

[9] Pueden consultarse los detalles de éstos y otros problemas en los libros de O'Rourke o de M. De Berg et al. Citados en la bibliografía.

5.- INVESTIGACIÓN EN GEOMETRÍA ALGORÍTMICA

La Geometría Algorítmica es también un activo campo de investigación en el que participan Informáticos y Matemáticos. Buena prueba de ello son las revistas especializadas en la materia publicadas por grandes editoriales científicas (*Discrete and Computational Geometry* de Springer, *Computational Geometry, Theory and Applications* de Elsevier, *International Journal on Computational Geometry* de World Scientific), todas ellas con un nivel elevado en las listas de índices de impacto, así como el gran número de artículos publicados en otras revistas científicas relacionadas. Para hacerse una idea de la cantidad de trabajos publicados y para hacer búquedas y consultas, puede consultarse la base de datos Geombib, de la que existe un espejo en la UPC (http://www-ma2.upc.es/~geomc/geombib/geombibe.html).

Existen congresos nacionales, europeos e internacionales específicos. En España los Encuentros de Geometría Computacional (9ª edición en Girona en 2001). En otros paises existen congresos locales (Francia, Canada, EEUU,...). Existe un congreso Europeo (el XVI se celebró en Israel en 2000). Y existen varios internacionales, de los cuales el más significativo es el patrocinado por la ACM (la edición XVIII tendrá lugar en Barcelona en 2002).

El gran auge de la investigación en Geometría Algorítmica se debe al gran número de campos en los que surgen problemas geométricos para los cuáles es preciso dar soluciones algorítmicas eficientes. Esto hace que sea una de las áreas más activas de la Matemática Aplicada, que da respuesta a algunos de los problemas que plantea la Tecnología Informática y se nutre a su vez de ella con nuevos problemas.

Dos de las direcciones en Internet relacionadas con la investigación en Geometría Algorítmica son la página mantenida por Jeff Erikson (http://compgeom.cs.uiuc.edu/~jeffe/compgeom/) y la mantenida por Godfried Toussaint (http://cgm.cs.mcgill.ca/~godfried/computational.geometry.html). En ellas hay numerosos enlaces entre los que cabe destacar los enlaces a congresos, publicaciones, páginas de grupos de investigación, software disponible en la red, etc.

En España varios grupos trabajan en Geometría Algorítmica. Entre ellos el de la Universidad de Cantabria (http://matsun1.matesco.unican.es/Cag/), el de la Universidad Politécnica de Cataluña (http://www-ma2.upc.es/~geomc/) y el de la Comunidad de Madrid (http://www.dma.fi.upm.es/research/geocomp/).

6.- BIBLIOGRAFÍA

J. O'Rourke:"*Computational Geometry in C*", Cambridge U. Press,1994.

M. de Berg y otros: "*Computational Geometry: Algorithms and Applications*", Springer, 1997.

J. D. Boissonat, M. Yvinec: "*Algorithmic Geometry*". Cambridge Univ. Press, 1998.

H. Edelsbrunner: "*Algorithms in Combinatorial Geometry*". Springer, 1987.

F. Preparata, M. I. Shamos: "*Computational Geometry: An Introduction*". Springer, 1985

ESTUDIO SOBRE LA HETEROGENEIDAD DE CONOCIMIENTOS BÁSICOS EN ALUMNOS DE PRIMER CURSO DE UNIVERSIDADES POLITÉCNICAS.

Mas Estelles, Jorge, Meseguer Dueñas, Jose M..

Escuela Universitaria de Informática, Departamento de Física Aplicada Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 VALENCIA
P.O. Box 22012, E-46071 VALENCIA, SPAIN.
E-Mail:jmas@fis.upv.es, jmmesegu@fis.upv.es

RESUMEN: El trabajo analiza los resultados de un test relativo a conocimientos básicos de Matemáticas y Física realizado con alumnos de primer curso de la Escuela Universitaria de Informática (EUI) y de la Escuela Técnica Superior de Ingenieros de Telecomunicaciones (ETSIT) de la Universidad Politécnica de Valencia, durante los cursos 1996-97 y 1997-98.

Los resultados se analizan estableciendo comparaciones según el Centro (EUI y ETSIT) y según procedencia (COU y FP).

1.INTRODUCCION

Tanto en las Escuelas y Facultades destinadas a estudios de Informática como en aquellos foros en los que se debate acerca de la preparación y posibilidades laborales de los informáticos parece percibirse, de forma más o menos generalizada, una cierta prevención contra los titulados en Telecomunicaciones, que son considerados, en mayor o menor grado, como "intrusos" en un mercado laboral que los informáticos consideran como propio. En opinión de algunas personas esta "pugna" podría resolverse enfatizando las diferencias existentes entre ambas titulaciones en los planes de estudio informáticos. Otras voces opinan que esta incursión de los titulados de Telecomunicaciones en los ámbitos más propios de informáticos responde a diferencias profundas entre el alumnado de una y otra titulación, más que a factores derivados de los contenidos de las titulaciones.

En el caso de la UPV, estas presuntas diferencias en el alumnado conllevan la consideración de la Escuela Técnica Superior de Ingenieros de Telecomunicaciones (ETSIT) como un centro de "élite", frente a la Escuela Universitaria de Informática (EUI) como centro donde las exigencias académicas son menores y el acceso más fácil. Es cierto que las "notas de corte" para acceder a la EUI y Facultad de Informática (FI) son notablemente más bajas que las exigidas para acceder a la ETSIT (5.7 en la E.U.I. el curso 1996-97 frente a 7.5 en la ETSIT el curso 1997-98), pero no es menos cierto que en otras Universidades no se ha podido constatar correlación clara entre la nota de acceso de los estudiantes y su posterior rendimiento (GONZÁLEZ, 1993), aunque

posiblemente ello sea debido a factores coyunturales. Esta última observación también podría cuestionar algo que las calificaciones preuniversitarias deberían garantizar: la relación directa entre nota y conocimientos básicos que debería tener un alumno. La ausencia de conocimientos básicos suficientes en el alumnado de nuevo ingreso es un argumento que es esgrimido por muchos profesores, cada vez con mayor frecuencia, para justificar los elevados índices de fracaso académico, sobre todo en primer curso.

También se debe tener en cuenta la procedencia de los alumnos: en las Escuelas Superiores y Facultades que imparten estudios de 2° ciclo, la mayoría de los alumnos proceden del Curso de Orientación Universitaria (C.O.U), a los que se les exige haber superado los exámenes de Selectividad. En las Escuelas Universitarias, aunque en teoría no se exige, la totalidad de los alumnos procedentes de COU lo hacen con la Selectividad superada. Los alumnos procedentes de Formación Profesional (F.P.) tienen reservado un cupo del 30% de las plazas totales de nuevo acceso en las Escuelas Universitarias, no se les exige el Selectivo pero tienen condicionado el paso a estudios de segundo ciclo.

El presente trabajo no pretende cuantificar los conocimientos básicos de los alumnos que acceden a la UPV, sino establecer comparaciones objetivas entre colectivos pertenecientes a distintos centros, y distintas vías de acceso a la Universidad.

2.ESTUDIO EXPERIMENTAL

Se pasó un test, que figura al final de este trabajo, a distintos colectivos de alumnos, sobre conocimientos básicos de Matemáticas (5 preguntas) y de Física (6 preguntas). A los alumnos procedentes de C.O.U. se les pedía que indicaran cuáles habían sido las calificaciones obtenidas en C.O.U. tanto en Matemáticas como en Física. Algunos de los datos aquí expuestos han sido ya presentados en otros trabajos (MÁS, 1997).

El test se pasó a los alumnos del primer semestre del primer curso de la (E.U.I.) el primer día del curso 1996-97, en el horario de las clases de Análisis Matemático I y Algebra. El número de alumnos encuestados fue de 272 sobre un total de 816 matriculados en la asignatura de Fundamentos Físicos de la Informática (33%), que era la de mayor número de alumnos matriculados, repartidos en cuatro grupos de las distintas titulaciones que entonces se impartían en la E.U.I.(EUI, 1996):

2 grupos de la titulación Ingeniero Técnico en Informática de Sistemas (I.T.I.S.), 1 grupo de la titulación Ingeniero Técnico en Informática de Gestión (I.T.I.G.), y 1 grupo de la titulación Ingeniero en Informática (I.I.).

Aunque estos grupos estaban adscritos a las distintas titulaciones, en la práctica los alumnos estaban distribuidos en todos los grupos. Por su procedencia se clasificaron en alumnos de nuevo ingreso de C.O.U. y de Formación Profesional de 2° Grado (F.P.II), alumnos repetidores de cursos anteriores y otros grupos (mayores de 25 años, otras titulaciones...).

A los alumnos procedentes de F.P, en primer curso sólo se les permite acceder a las titulaciones de primer ciclo, I.T.I.S. e I.T.I.G.

La corrección del test fue realizada en su totalidad por dos profesores, no apareciendo, en ningún caso, discrepancias entre las dos correcciones superiores a medio punto sobre 10 (5%) en las calificaciones medias obtenidas por cada grupo de alumnos ó por el total de ellos.

También se pasó el mismo test a los alumnos del primer semestre del primer curso de la Escuela Técnica Superior de Ingenieros de Telecomunicaciones (E.T.S.I.T.). Se realizó el tercer día del curso 1997-98, en el horario de la clase de Física I.

El número de alumnos encuestados fue de 82 sobre un total de 375 matriculados en la asignatura de Física I (21.9%), pertenecientes todos ellos a un grupo de la titulación de Ingeniero de Telecomunicaciones. Su procedencia era similar a los de la EUI salvo los alumnos de FP, que no pueden acceder a titulaciones con segundo ciclo. La corrección fue realizada por uno de los profesores que corrigieron el test de la EUI, de acuerdo con los mismos criterios.

3.RESULTADOS

Las tablas que aparecen a continuación muestran, de forma resumida, los resultados obtenidos. Dada la escasa representatividad de los grupos de alumnos "Repetidores" y "Otros" no se dan sus resultados con detalle.

RESULTADOS EUI

PROCEDENCIA				
COU		FP	REPETIDORES	OTROS
163 (59.9%)		83 (30.5%)	13 (4.8%)	13 (4.8%)
CALIFICACIONES OBTENIDAS EN COU (%)				
	MATEMATICAS	FISICA		
M.H. ó Sobresaliente	12.9	11		
Notable	37.4	40.6		
Suficiente ó Bien	49.7	48.4		

RESULTADOS DEL TEST DE LOS ALUMNOS DE LA EUI (máximo 10)

PROCEDENCIA	MATEMATICAS		FISICA	
	MEDIA	DESV. TIPICA	MEDIA	DESV. TIPICA
COU	3.9	2.5	3.0	2.1
FP	0.1	0.6	0.6	1.0
TOTAL	2.7	2.5	2.1	4.7

DISTRIBUCIÓN DE LOS ALUMNOS PROCEDENTES DE COU EN LA EUI SEGÚN RESPUESTAS CORRECTAS

RESPUESTAS CORRECTAS	MATEMATICAS	FISICA
0	15 (9.2%)	20 (12.3%)
1	44 (27 %)	53 (32.5%)
2	46 (28.2%)	50 (30.7%)
3	49 (30.0%)	22 (13.5%)
4	9 (5.5%)	12 (7.4%)
5	0	6 (3.7%)
6		0

DISTRIBUCIÓN DE LOS ALUMNOS PROCEDENTES DE FP EN LA EUI SEGÚN RESPUESTAS CORRECTAS

RESPUESTAS CORRECTAS	MATEMATICAS	FISICA
0	78 (94%)	59 (71.1%)
1	4 (4.8%)	18 (21.7%)
2	1 (1.2%)	6 (7.2%)
3	0	0
4	0	0
5	0	0
6		0

DISTRIBUCIÓN GLOBAL DE LOS ALUMNOS DE LA EUI SEGÚN RESPUESTAS CORRECTAS

RESPUESTAS CORRECTAS	MATEMATICAS	FISICA
0	100 (36.8%)	93 (34.2%)
1	52 (19.1%)	74 (27.2%)
2	55 (20.2%)	63 (23.2%)
3	54 (19.9%)	23 (8.4%)
4	11 (4.0%)	12 (4.4%)
5	0	7 (2.6%)
6		0

RESULTADOS ETSIT

PROCEDENCIA				
COU		FP	REPETIDORES	OTROS
67 (81.7%)			3 (3.7%)	12 (14.6%)
CALIFICACIONES OBTENIDAS EN COU (%)				
	MATEMATICAS	FISICA		
M.H. ó Sobresaliente	52.2	52.2		
Notable	28.4	35.8		
Suficiente ó Bien	11.9	4.5		
NS/NC	7.5	7.5		

RESULTADOS DEL TEST DE LOS ALUMNOS DE LA ETSIT (máximo 10)

PROCEDENCIA	MATEMATICAS		FISICA	
	MEDIA	DESV. TIPICA	MEDIA	DESV. TIPICA
COU	6.6	2.6	5.5	2.9

DISTRIBUCIÓN DE LOS ALUMNOS PROCEDENTES DE COU EN LA ETSIT SEGÚN RESPUESTAS CORRECTAS

RESPUESTAS CORRECTAS	MATEMATICAS	FISICA
0	0 (0%)	4 (6%)
1	7 (10.4 %)	15 (22.3%)
2	16 (23.9%)	11 (16.4%)
3	7 (10.4%)	14 (20.9%)
4	25 (37.3%)	14 (20.9%)
5	12 (18 %)	6 (9%)
6		3 (4.5%)

4.DISCUSION

En primer lugar, respecto de la procedencia de los alumnos encuestados, es significativa la escasa presencia de alumnos repetidores en todos los grupos, que contrasta con los porcentajes que aparecen en los datos de matrícula. Estas cifras podrían ser achacadas a la concentración de alumnos repetidores en grupos distintos de los estudiados por razones burocráticas, ó simplemente a su inasistencia a clase durante los primeros días del curso, que es cuando se realizó el test.

En cualquier caso, el análisis por grupos de procedencia elimina las posibles distorsiones que estas diferencias pudieran producir.

Si comparamos los resultados obtenidos en el test con las calificaciones obtenidas en COU, éstas últimas son siempre más altas, tanto para los alumnos de la EUI como para los de la ETSIT.

Mientras en la EUI las notas obtenidas en COU se concentran en torno al Notable, la nota media del test es de 3.9 en Matemáticas, y de 3 en Física.

Con respecto a los alumnos de la ETSIT, un 52.2% manifiesta haber obtenido Sobresaliente ó Matrícula de Honor tanto en Matemáticas como en Física, y Notable un 28% en Matemáticas y un 35% en Física, mientras que las notas medias del test son de 6.6 en Matemáticas, y 5.5 en Física. Estas excesivas diferencias en los resultados podrían ser achacadas a multitud de factores tales como la dificultad del test propuesto, la inactividad estival de los alumnos, la incomprensión de la notación utilizada, y posiblemente otros factores adicionales. A pesar de ello el valor absoluto de las calificaciones se debe relativizar, ya que en otros tipos de pruebas se produce el mismo tipo de discrepancias: en los resultados de la prueba de Selectividad de Junio de 1996 en los tribunales de la Universidad Politécnica de Valencia se observa que, en la asignatura de Matemáticas I se examinaron 2994 alumnos, obteniendo una calificación media de 4.1 sobre 10, con una desviación típica de 2.3. En la asignatura de Física el total de alumnos examinados fue de 2334, obteniendo una nota media de 5.1 con una desviación típica de 2.2. Estas calificaciones del Selectivo son notablemente inferiores a las obtenidas en COU.

Aunque estos motivos podrían cuestionar los resultados absolutos del test, no deberían cuestionar los resultados relativos a las diferencias existentes entre los diferentes grupos de alumnos.

Los resultados obtenidos por los alumnos de C.O.U., más bajos de lo que cabría esperar, se acentúan de forma dramática con los alumnos procedentes de F.P, que no han necesitado superar la prueba de Selectividad. Las notas medias obtenidas por estos fueron de 0.14 en Matemáticas y 0.6 en Física, lo que muestra una ausencia prácticamente total de los conocimientos cuestionados. A modo de ejemplo, en el test de Matemáticas el 94% de los alumnos de F.P. no respondió correctamente a ninguna de las cuestiones planteadas, apareciendo el 30% de los test completamente en blanco. Estos alumnos tan sólo mostraron un nivel de aciertos relativamente alto en la cuestión de Física relativa a un circuito eléctrico, lo que podría indicar que poseen una cierta preparación aplicada pero poca preparación básica.

Los alumnos procedentes de COU de la ETSIT muestran una nota media en Matemáticas y en Física claramente mayor que sus homólogos de la EUI (en la primera 6.6 frente a 3.9, y en la segunda 5.5 frente a 3). La desviación típica en Matemáticas es similar (2.6 frente a 2.5) y existen mayores diferencias en Física (2.9 frente a 2.1). Estos mejores resultados deberían estar claramente correlacionados con las mejores calificaciones obtenidas en COU por los alumnos

de la ETSIT y con la mayor nota de corte exigida en dicha Escuela. Además, estas notas medias de los alumnos de la ETSIT son superiores a las medias del Selectivo de Junio de 1997 en el Tribunal de la UPV (4.89 con una desviación típica de 2.18 en Matemáticas I y 4.4. con una desviación típica de 2.12 en Física), lo que corrobora la idea de que los alumnos de esta Escuela tienen un nivel medio de conocimientos básicos generales superior a la media.

Las distintas poblaciones estudiadas fueron sometidas al test F de Snedecor (Snedecor, 1970), que permite cuantificar el nivel de significación de las diferencias encontradas entre unas y otras. En este estudio, las diferencias encontradas entre los alumnos de la EUI procedentes de COU y de FP son significativas con una probabilidad de, al menos, el 99%. Idéntica conclusión se obtiene de la comparación entre los alumnos procedentes de COU de la EUI y de la ETSIT.

No obstante, estos resultados tan concluyentes deben ser tomados en términos absolutos con una cierta cautela, ya que múltiples factores pueden haber influido en ellos, como ya se ha apuntado. De cualquier modo en términos relativos estos resultados sí que son reveladores.

5.AGRADECIMIENTOS

A nuestros/as compañeros/as de los Departamentos de Matemática Aplicada y de Estadística e Investigación Operativa de la E.U.I., a nuestros/as compañeros/as de la U.D. de la E.T.S.I.T. del DFA, y a todos cuantos alumnos se sometieron al test, y sin los cuales, obviamente, este trabajo no hubiera sido posible.

6.REFERENCIAS

ESCUELA UNIVERSITARIA DE INFORMÁTICA, (1996) "Guía de los estudios del curso 1996-97", SPUPV-98.2090.

GONZÁLEZ, R. M., (1993): "Rendimiento académico en la Universidad Politécnica de Madrid: Estudio longitudinal en primer ciclo", Vol. 1 y 2, Madrid, I.C.E. de la U.P.M.

MÁS, J (1997): "Conocimientos básicos de los alumnos de nuevo ingreso en las Escuelas Universitarias. Evolución en el tiempo", Enseñanza de las Ciencias, Número extra V Congreso Internacional sobre investigación en la didáctica de las Ciencias, 251-253. Murcia.

SNEDECOR, G.W, COCHRAN, W.G, (1970): "Métodos estadísticos aplicados a la investigación agrícola y biológica", Compañía Editorial Continental, México.

ENCUESTA CONOCIMIENTOS PREVIOS

PROCEDENCIA COU FP REPETIDOR OTROS
(Marcar la que corresponda)

ULTIMAS NOTAS COU O FP MATEMATICAS FISICA

CUESTIONES

1.- Sean $\vec{a} = 3\vec{i} + 2\vec{j} - 3\vec{k}$

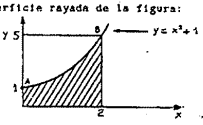
$\vec{b} = 3\vec{j} + \vec{k}$

Obtener: a) $\vec{a} \cdot \vec{b}$

b) $\vec{a} \wedge \vec{b}$

2.- Realizar la integral: $\int \frac{dx}{x}$

3.- Calcular la superficie rayada de la figura:



4.- Calcular la derivada de la función $y = x\,e^{x}$ en el punto P = (0,0)

5.- Sean los números complejos $\bar{A} = 2 + 3j$ y $\bar{B} = 1 - j$ $(j = \sqrt{-1})$.

Calcular: a) $\bar{A} \cdot \bar{B}$

b) $\bar{A} + \bar{B}$

6.- Decir si son escalares o vectoriales las siguientes magnitudes, indicando sus unidades en el S.I.:

a) Peso

b) Trabajo

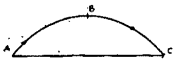
c) Intensidad de corriente eléctrica

7.- Una partícula es lanzada y describe una trayectoria parabólica. ¿En qué punto es máxima, mínima o de manera constante.....

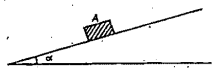
a) La aceleración a la que está sometida?

b) La velocidad?

c) La energía potencial?



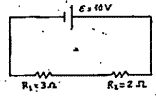
8.- Dibujar las fuerzas que actúan sobre el cuerpo A de la figura. (Datos: Coeficiente de rozamiento μ y masa m).



9.- Definir:

- Energía cinética
- Potencia

10.- En el circuito de la figura, calcular la intensidad de corriente I indicando el sentido de la misma.



11.- Calcular la densidad de una esfera homogénea de radio R y masa M.

EXPERIENCIAS DE PRÁCTICAS CON PVM EN LA ASIGNATURA DE ARQUITECTURA DE COMPUTADORES Y SISTEMAS OPERATIVOS 2

Miguel A. Mateo Pla[1], Julio Pons Terol[1]

[1]*Departamento de Informática de Sistemas y Computadoras (DISCA)*
Universidad Politécnica de Valencia (UPV)
e-mail: {mimateo,jpons}@disca.upv.es

RESUMEN: En el presente artículo se presentan las experiencias obtenidas durante la preparación y realización de una práctica con PVM [2] (Parallel Virtual Machine) para la asignatura Arquitectura de Computadores y Sistemas Operativos 2 (ACSO2), impartida en octavo semestre en la Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT) de la UPV.

Esta práctica se desarrolla en torno a un problema sencillo con necesidades computacionales altas, y con una sustancial ventaja sobre otros problemas estudiados: una salida gráfica. Este hecho permite al alumno apreciar mejor el paralelismo y las diferentes formas de resolver el algoritmo.

Se justificará la elección de PVM como entorno de programación y las herramientas utilizadas como ayuda al desarrollo de la práctica.

1.- INTRODUCCIÓN.

La asignatura ACSO2 se imparte por primera vez en el curso 1999-2000. Esta asignatura se imparte en el semestre octavo de todas las especialidades de la titulación de Ingeniero de Telecomunicación de la ETSIT.

Esta nueva asignatura surge de la implantación de los planes de estudio de 1996. En el plan de estudio anterior existía la asignatura anual ACSO de la que se han aprovechado algunos materiales. Se ha aprovechado el paso al nuevo plan de estudios para revisar y actualizar tanto el temario como los materiales y prácticas utilizados. El resultado ha sido la introducción de nuevos temas en el temario y sus prácticas correspondientes.

Uno de los temas que se ha introducido es el de las Redes de Estaciones de Trabajo (en inglés Network Of Workstations ó NOW). Este tema se trata como un aspecto actual de los sistemas operativos y de la arquitectura de computadores.

Para comprender el funcionamiento de las NOW se necesitaba encontrar un entorno de programación en el que preparar la práctica y una aplicación susceptible de paralelizarse de forma sencilla.

Con la experiencia obtenida en las prácticas de asignaturas anteriores [1], se ha observado que los alumnos se motivan más con aquellos problemas en los que pueden ver el resultado de la computación de forma gráfica. Las prácticas de asignaturas anteriores contaban con esa salida gráfica, pero se utilizaba un simulador de multicomputadores por lo que algunos alumnos tenían la impresión de que las ejecuciones no eran realistas.

Teniendo estos conceptos en mente se diseña una práctica que pretende la utilización de una NOW para la solución de un problema sencillo de forma paralela, sin utilizar simuladores y obteniendo resultados gráficos atractivos.

En este artículo presentamos los objetivos concretos perseguidos con esta práctica y las experiencias obtenidas durante su preparación, realización y posterior evaluación.

2.- OBJETIVOS

Los objetivos principales de la práctica son:

1. Introducir de forma práctica una NOW.
2. Dar a conocer al alumno una herramienta de programación de NOW.
3. Que el alumno sea capaz de escribir o modificar una aplicación paralela sencilla.
4. Conseguir que el alumno descubra los principales problemas de la programación distribuida.

Para facilitar el trabajo de los alumnos en otros laboratorios o en su casa, es deseable que la práctica se pueda probar en entornos diferentes de los disponibles actualmente en el Laboratorio de Arquitectura de Computadores del DISCA, por ello se ha escogido un entorno de programación de NOW lo más independiente posible del sistema operativo y de la arquitectura de la máquina.

3.- ENTORNO

Los dos condicionantes básicos del diseño de la práctica son: el laboratorio y el perfil del alumno. Sin definir con claridad ambos entornos será probable que la práctica no se pueda realizar o que los alumnos no puedan sacarle el aprovechamiento deseado.

Por este motivo se definen a continuación estos dos elementos.

a) El laboratorio de Arquitectura de Computadores

Todas las prácticas de ACSO2 se realizan en el laboratorio de Arquitectura de Computadores. Este laboratorio cuenta con 27 puestos de prácticas con las siguientes características:

- puestos con AMD K6-2 450 MHz.
- 3 puestos con Pentium II a 350 MHz.
- 6 puestos biprocesadores Pentium II a 350 MHz.
- 4 puestos biprocesadores Pentium 200 MMX
- 3 puestos con Pentium 166 MMX.

Todos los puestos cuentan con 64 MB de RAM, 8 GB de disco duro y monitor de 17". Todos los puestos están conectados mediante una red Ethernet a 100 Mb.

Los grupos de prácticas previstos para la realización de prácticas son 12 grupos de 2 personas. De esta forma se consigue tener grupos de prácticas reducidos y que todos alumnos tengan oportunidad de trabajar. Los puestos del laboratorio que quedan libres son los de menor calidad y aquellos que se utilizarán como nodos adicionales de las máquinas virtuales de PVM.

En cuanto a sistema operativo, todos los puestos disponen de arranque en Windows y en Linux. En las prácticas de ACSO2 se utiliza sólo LINUX.

El laboratorio dispone, además de los puestos para los alumnos, de un servidor LINUX para docencia. En este servidor existe una cuenta para cada uno de los alumnos matriculados en las asignaturas que utilizan LINUX en el laboratorio. El servidor está conectado 24 horas para que los alumnos se puedan conectar y utilizarlo cuando quieran de forma remota.

La utilización de NFS (Network File System) en todos los equipos permite que un alumno siempre acceda a la misma información, sea cual sea el puesto de trabajo que ocupe durante una sesión de prácticas. La información de cada usuario se encuentra almacenada en el servidor.

b) Alumnos

Los alumnos de ACSO2 llegan a la asignatura con poca experiencia en programación. Esto significa que no se puede exigir grandes desarrollos software, limitando las prácticas a modificaciones en programas ya escritos por los profesores, ya sea de forma completa o parcial.

En cualquier caso, a la altura del curso en la que se realiza la práctica, los alumnos han realizado dos prácticas utilizando el lenguaje de programación C, unas 10 horas de laboratorio. El trabajo realizado en estas prácticas así como el interés mostrado de los alumnos resuelven en parte este problema.

Teniendo en cuenta estos condicionantes, se plantea la práctica de forma que aquellos alumnos con menos experiencia en programación puedan sacar partido de la misma, al mismo tiempo que los que si tengan experiencia se motiven.

4.- DESARROLLO

Tomando en cuenta todas las consideraciones anteriores se decide:

1. Utilización de PVM como entorno de desarrollo de aplicaciones para NOW. Este entorno es de libre distribución y está disponible para las arquitecturas y sistemas operativos más comunes. Dispone así mismo de una amplia documentación.

2. Utilizar como problema base la implementación de un programa que dibuje las figuras de Mandelbrot. Se suministra a los alumnos una versión secuencial y otra preparada para su ejecución en PVM.

3. Separar en la aplicación la parte gráfica. De esta forma los alumnos no tendrán que enfrentarse a los problemas inherentes a la programación gráfica, complicada en cualquier sistema operativo.

Todas las aplicaciones suministradas al alumno imprimen en su salida estándar el tiempo de ejecución del algoritmo. Este valor esta claramente influenciado por la carga en la máquina, pero es más sencillo de medir y entender por los alumnos que otros parámetros más exactos pero más difíciles de obtener: utilización de CPU por parte de los procesos que componen la aplicación, tiempo de sobrecarga introducido por PVM, tiempos de comunicaciones...

a) Estructura aplicación base

A los alumnos se les proporciona una primera versión de la aplicación PVM completamente funcional. Con esta aplicación se realizarán tareas de monitorización de PVM y se enseña a los alumnos cuales son las llamadas fundamentales de la librería de una forma práctica.
La estructura de esta aplicación se puede observar en la Figura. Los elementos que componen la aplicación son:

- *Maestro*. Este proceso se encarga de inicializar el conjunto de tareas necesarias, recibir los resultados y decidir cuando finaliza la aplicación. Sólo existe un *Maestro* que se ejecuta en la misma máquina donde se encuentra el alumno.

- *Trabajador*. Proceso que calcula una parte del problema. El *Maestro* arranca *N* procesos de este tipo. Pueden ejecutarse en cualquier máquina de la NOW.

- *Dibujador*. Proceso que dibuja en pantalla las líneas ya calculadas. Cuando el *Maestro* detecta que se ha calculado completamente una línea se la manda a este proceso. Se ejecuta en la misma máquina que el *Maestro*.

La idea de esta primera versión es dividir la pantalla en tantas zonas como trabajadores, todas con el mismo tamaño. Estas zonas se componen de líneas horizontales consecutivas.

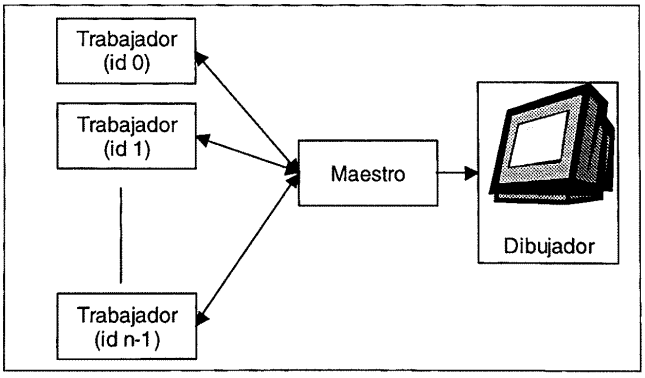


Figura 1: Estructura de la aplicación base

b) Ejercicios guiados.

El primer ejercicio consiste en ejecutar la versión secuencial del algoritmo de Mandelbrot (sin PVM). Variando la resolución y el número de iteraciones del algoritmo se obtienen diferentes tiempos de ejecución. Estos tiempos se tomarán como base para los siguientes ejercicios.

Esta primera versión carece de salida gráfica para que los alumnos puedan comprender mejor su funcionamiento. Para dar una idea del resultado existen multitud de programas que dibujan el conjunto de Mandelbrot, entre ellos *kfract* instalado en las máquinas del laboratorio.

El siguiente ejercicio consiste en la ejecución de la versión PVM del algoritmo. Para ello, se enseña al alumno como poner en marcha PVM y la forma de añadir y quitar máquinas de la misma, es decir, los mecanismos para determinar y modificar la configuración de la máquina paralela virtual.

Tras esto se prueba la salida gráfica del algoritmo para que el alumno pueda apreciar como las líneas se calculan y dibujan de forma "paralela". Esta salida gráfica no se deberá realizar a la hora de tomar tiempos para poder comparar mejor los tiempos debidos al algoritmo de Mandelbrot.

El siguiente ejercicio es comparar los tiempos de ejecución de esta versión con la versión secuencial, modificando el número de trabajadores y el número de máquinas que componen PVM con la que está trabajando.

De esta forma se demuestra al alumno dos cosas:

1. Que el algoritmo secuencial funciona mejor cuando sólo hay un procesador.

2. Que la existencia de N procesadores no significa una mejora de N en el tiempo de ejecución, pero que existe una mejora.

Así mismo, se invita al alumno a probar la consola XPVM. Con esta aplicación se puede obtener una traza de las llamadas a la máquina virtual y observar el intercambio de mensajes entre los componentes de la aplicación.

Estos mecanismos de monitorización serán útiles en el desarrollo de las soluciones propias de los alumnos, ya que permite descubrir con mayor facilidad en que parte del código se encuentran los programas al presentar errores.

Como comentario adicional se muestra como la consola XPVM consume parte de los recursos de la máquina, disminuyendo la velocidad de la ejecución efectiva de nuestra solución. Con esta reflexión deben decidir si dejar la consola en marcha o pararla antes de tomar tiempos.

c) Ejercicio propuesto

El ejercicio propuesto es la modificación de la aplicación suministrada para mejorar el tiempo total de la aplicación. Para que los alumnos entiendan por donde pueden introducir mejoras se les indican los siguientes fallos en la aplicación base:

1. Todos los procesadores calculan el mismo número de líneas.

2. No todas las líneas tienen el mismo coste computacional. En la figura suministrada las líneas más "caras" se encuentran en el medio, por lo que algunos Trabajadores tienen que hacer más trabajo.

3. No todos los ordenadores tiene las mismas prestaciones.

Estos no son los únicos aspectos mejorables pero son los que creemos que se entienden mejor por los alumnos. Con estas dos pistas, las soluciones planteadas por los alumnos se basan en una combinación de las siguientes estrategias:

1. Que la asignación de líneas no haga que aquellas zonas más caras sean asignadas a un único Trabajador.

2. Agrupar mayor cantidad de información en los mensajes que los trabajadores intercambian con el servidor.

3. Que el reparto de las líneas no sea estático. De esta forma un Trabajador le pide líneas al Maestro y este le informa de la siguiente línea o grupo de líneas a calcular.

La forma de evaluación de estos ejercicios es una competición. Se informa de una fecha tope para preparar la aplicación y de la configuración que se utilizará. Llegada la fecha se ejecutan todas las soluciones planteadas por los alumnos en una PVM, aquel grupo cuya solución obtenga mejores tiempos obtendrá la mejor nota.

Los grupos tienen asegurado el aprobado siempre que la solución funcione.

5.-PROBLEMAS DETECTADOS

El año de implantación de las prácticas es el año en el que mayor número de problemas se detectan en la misma, incluso en aquellas prácticas con gran tiempo de dedicación en su preparación.

Surgen normalmente dos tipos de problemas: aquellos relacionados con las suposiciones que se hacen de antemano sobre el entorno y aquellos problemas en los que no se había pensado.

En el caso de la presente práctica han aparecido ambos tipos de problemas. Aunque creemos que ninguno de ellos es insalvable ni ha influido en los objetivos principales de la práctica si que ha podido desanimar a los alumnos en algunos momentos.

a) Entorno

El principal problema debido al entorno ha sido imposibilidad de montar a tiempo un "cluster" independiente de máquinas Linux que se dedicara exclusivamente a la ejecución de las aplicaciones de los alumnos. Esto ha provocado que los alumnos tuvieran que utilizar como nodos de sus PVM, los computadores que otros compañeros estaban utilizando para la edición de trabajos. Este hecho provocaba una gran variabilidad de los resultados obtenidos con la versión paralela del algoritmo.

b) Problemas nuevos

El principal problema ha sido no establecer de antemano las características de la PVM donde se probarían las soluciones de los alumnos. Este hecho provocaba en los alumnos cierta ansiedad sobre la bondad de la solución en la que estaban trabajando.

Por otra parte el tiempo que los alumnos han dedicado a la práctica ha sido mucho mayor del esperado. El que los alumnos dedicasen más tiempo del esperado se debe a dos factores independientes: la poca experiencia de programación en C y el afán, a veces obsesión, de superación de la solución obtenida.

El problema de la experiencia de programación en C es difícil de superar, aunque se mantienen conversaciones con profesores de asignaturas de programación para que se solucione año a año.

El problema del tiempo dedicado por intentar mejorar la solución se debe en parte a la forma propuesta de evaluación y la inexistencia de tiempos de referencia que indicasen la bondad de la solución. Este hecho podría tener solución con una tabla de tiempos de referencia y un conjunto de máquinas dedicadas de forma exclusiva a la ejecución de las soluciones de los alumnos, problema muy relacionado con el entorno.

6.-CONCLUSIONES

Con el trabajo realizado se ha desarrollado una práctica que ha permitido al alumno conocer una herramienta de trabajo con redes de estaciones de trabajo. Así mismo el alumno ha resuelto un problema sencillos construyendo una aplicación paralela.
El alumno ha realizado variantes de los programas estudiando los parámetros que influyen en las prestaciones de una NOW.

La práctica puede adaptarse fácilmente a otras asignaturas que toquen el tema de las redes de estaciones de trabajo. Por ejemplo, en la unidad docente de Arquitectura de Computadores, se podría adaptar a la asignatura Sistemas Multiprocesadores, optativa de 9° cuatrimestre de la Facultad de Informática de Valencia.

En general se han cumplido los objetivos, por lo que junto con la motivación que se ha observado en los alumnos, nos permite calificar la experiencia de satisfactoria.

7.-BIBLIOGRAFIA

[1] "Prácticas con el Simulador de Multicomputadores". José Duato y Pedro Lopez. 1993.
[2] Página principal del proyecto PVM: http://www.epm.ornl.gov/pvm/
[3] A.Geist et. al. "PVM3 User's Guide and reference Manual" Septiembre 1994.
[4] "Parallel Programming Languages and Environments" Ira Pramanick. 2° Capítulo de "High Performance Cluster Computing" Editor Rajkumar Buyya. Editorial Prentice Hall.
[5] "MPI and PVM programming" Ira Pramanick. 3° Capítulo de "High Performance Cluster Computing" Editor Rajkumar Buyya Editorial Prentice Hall.

APRENDIZAJE A TRAVES DE LA ESCRITURA: EXPERIENCIAS.

José Miró Julià

Dept. de Matemàtiques i Informàtica
Universitat de les Illes Balears
Campus UIB
07071 Palma de Mallorca
e-mail: joe@ipc4.uib.es

RESUMEN: Durante el curso 1999-2000 he usado un nuevo método de aprendizaje consistente en hacer que los alumnos aprendan los contenidos de la asignatura escribiendo sobre ella. En este artículo se presenta una evaluación de los resultados obtenidos.

1.- INTRODUCCIÓN.

En mis años de docente he observado que puedes esperar de los alumnos que aprendan datos, ejemplos, circuitos, algoritmos, pero que raramente entienden el significado de los datos y ejemplos, o los razonamientos seguidos en el diseño de circuitos y algoritmos. El conocimiento que obtienen los alumnos es principalmente *pasivo* (reciben información, la almacenan, la olvidan). En contraposición podemos definir conocimiento *activo* a aquel basado en fundamentos y razonamientos con el cual, a partir de datos, se pueden deducir conclusiones y en definitiva, generar nuevos conocimientos. En todo saber el conocimiento activo es mucho más conveniente que el pasivo, y muy especialmente en la informática, donde lo que se ha explicado en primero puede ser obsoleto cuando los alumnos acaben la carrera cinco o seis años después. El conocimiento pasivo pasa, el conocimiento activo, los razonamientos y fundamentos, son perdurables.

En el Jenui'99 presenté una experiencia que empezaba a poner en práctica en la cual los alumnos aprenden el contenido de la asignatura de Estructura de Computadores escribiendo sobre ella [1]. El objetivo de esta innovación es el de conseguir que los alumnos tengan un conocimiento activo de la asignatura. En el artículo se explica cómo la escritura es un buen vehículo para conseguir este tipo de conocimiento ya que exige al alumno que primero haga suya la idea a exponer, a que no deje agujeros en sus líneas de razonamiento, o a que busque ejemplos adecuados a sus razonamientos. Como ventaja adicional, los alumnos aprenden a escribir. Los resultados obtenidos han sido muy buenos y la evaluación que han hecho los alumnos del curso también ha sido muy positiva. En este artículo discutiré los resultados obtenidos tras este primer año de la experiencia.

Es importante notar que la escritura no es la forma en la que se *evalúan* los conocimientos del curso sino la forma cómo los alumnos *aprenden* estos conocimientos. No se trata de enseñar mediante el `método tradicional' y después pedir un informe, sino que en clase se aprende a escribir sobre un tema y *así* se aprenden los conocimientos del tema.

El curso se ha estructurado en cinco ciclos. En cada uno de ellos se introduce a los alumnos las ideas básicas de un tema de la asignatura y de un tema de escritura. Los temas desarrollados pueden verse en la Figura 1. El ciclo se desarrolla en unas cinco semanas (20 horas de clase) y al final del mismo los alumnos, en equipos de dos personas, han de entregar un documento del tema de arquitectura del ciclo. Este documento se valorará en función de sus contenidos –sobre todo de la calidad de sus razonamientos– y de la calidad de la escritura, según lo expuesto en el ciclo. Los cinco documentos representan dos terceras partes de la nota final del curso, la tercera parte restante se obtiene de un examen de problemas. Algunos de los documentos entregados por los alumnos durante el curso pueden descargarse de http://dmi.uib.es/people/joc/teach/EstrC.html.

Dentro de cada ciclo se dedican dos horas a exponer brevemente las ideas principales del tema de arquitectura, otras dos a exponer el tema de escritura, entre cuatro y seis horas se dedican a problemas y el resto se dedica a escribir el documento. Esto está en consonancia con los objetivos propuestos: la mayor parte del tiempo de clase se dedica a aprender el tema a base de escribir sobre él.

Ciclo	Tema de Arquitectura	Tema de Escritura
1	Aritmética	Edición
2	El procesador	Claridad de frases
3	La segmentación	Estructura de párrafos
4	La unidad de memoria	Estructura de secciones
5	Entrada/Salida	Tablas y gráficos

Figura 1: Contenido de los ciclos

Durante el año he ido estudiando los resultados obtenidos, he hablado mucho con los alumnos y al final he hecho una encuesta valorativa de la experiencia. Dados los resultados y las respuestas de los alumnos presento en este artículo mi valoración del método. Es una valoración subjetiva, pero he intentado basarla en hechos observables y no en intuiciones para hacerla lo más fiable posible.

2.- VENTAJAS DEL METODO

Usando la escritura como vía para el aprendizaje los alumnos aprenden a escribir. Pero además de esta ventaja obvia que presenta el método, existen otras. Describo a continuación las que considero más importantes. Aunque las ventajas están interrelacionadas, para explicarlas las he dividido en tres grupos: mejor conocimiento, mejor comunicación y mayor autonomía.

El conocimiento obtenido por los alumnos es más profundo que con la enseñanza tradicional. Para poder escribir sobre un tema los alumnos han tenido que hacer suyos los conocimientos que han expuesto. Gracias a esto, han aprendido a relacionar conceptos entre diferentes temas y entre diferentes asignaturas: a menudo usaban conocimientos obtenidos en otras asignaturas para mejorar sus escritos en ésta. También están más seguros de sus conocimientos. Saben qué es lo que saben y qué es lo que no. Durante el curso se dieron cuenta de lo que yo les dije el primer día: sólo puedes escribir bien de lo que sabes. El papel no engaña.

Los alumnos han aprendido a comunicarse. Esto es más que simplemente aprender a escribir. Aparte de saber poner las palabras unas detrás de otras, han aprendido cuál es la estructura de una exposición efectiva, dónde debe ir la información importante de cada frase, de cada párrafo, de cada sección. Han aprendido a identificar al lector con quien quieren comunicarse y a escribir pensando en él.

Esto a su vez les ha ayudado a leer mejor. Esta mejora no es algo que se haya enseñado durante el curso, sino que está unido al aprender a escribir. Y han tenido que aplicar esta mejora de su capacidad de lectura, ya que la explicación del tema hecha en clase es breve y sólo de las líneas básicas. El resto de la información que necesitan para escribir el informe la han de obtener de textos que han de leer y entender.

Los alumnos se han vuelto más autónomos. Al cabo del año han tenido repetidamente que aprender y explicar un tema. A partir de una pequeña información inicial han tenido que escoger y delinear el tema que quieren tratar, determinar cuáles son los campos de los que no saben lo suficiente y han de aprender más, buscar la información pertinente, digerirla y unirla al conocimiento que ya tienen. Todo esto lo han hecho con cada vez menor ayuda del profesor. Al final del curso saben leer, aprender, comprender y exponer un tema ellos solos.

3.- INCONVENIENTES

Como todo en esta vida, este método también presenta inconvenientes. Algunos los había previsto y se exponen en el artículo anterior [1], otros se han presentado imprevistos. En todo caso es importante tener presentes los inconvenientes identificados para tratar de disminuir los efectos.

La mayor parte de los trabajos entregados se pueden considerar *light*. Son de divulgación, cualitativos, sin entrar en detalles en los temas complejos. Un motivo de esto es que el escritor ha de saber bien y tener mucha seguridad en lo que explica, conque el tema expuesto ha de estar algo por debajo de su nivel de conocimientos. Esto explica algo, pero no todo. Creo que los alumnos han evitado dificultades, han buscado temas sencillos y han circunvalado las partes complejas. Es decir, dentro de lo que han podido han ido a lo fácil. Este es un problema difícil de resolver ya que es importante dejar libertad de elección de tema y enfoque a los alumnos para poder fomentar muchas de las ventajas comentadas en la sección anterior.

El aprendizaje a través de la escritura va muy bien para entender conceptos, pero la asignatura exige también habilidades en la resolución de problemas. No he encontrado una buena manera de compaginar ambas cosas. Para cumplir con el objetivo de ser capaz de resolver problemas, los alumnos disponían de una lista de problemas resueltos y un tercio de la nota del curso provenía de un examen sobre los problemas de la lista. A los alumnos no les gustaron ni los problemas, ni el examen. Y con razón. Tal como me escribió un alumno en la encuesta final de curso "Preparar el examen se ha convertido en una tarea prácticamente burocrática que desentona totalmente con el espíritu y esencia en que se ha desenvuelto el resto del curso".

Otra idea fue la de requerir el diseño y la resolución de problemas originales en los trabajos. Tras dos ciclos vi que los alumnos no eran capaces de crear problemas y encajarlos dentro de sus trabajos. Decidí que era una tarea demasiado difícil y, quizá prematuramente, eliminé el requisito. En cursos venideros será prioritario buscar otras maneras de abordar los problemas.

4.- CONSEJOS PARA EL QUE QUIERA HACER LA PRUEBA

Es la primera vez que se ha llevado a cabo una experiencia de este tipo. Yo tenía unas ideas sobre qué se podía hacer, pero ante la falta de experiencia he estado abierto durante todo el curso a las sugerencias de los alumnos y he estado preparado para hacer todos los cambios necesarios en cuanto

hiciera falta. Si alguien quiere poner en marcha una experiencia de este tipo, estos consejos le pueden resultar útiles.

La evolución es lenta. En cualquier asignatura los alumnos conocen mejor el Tema 1 después de haber explicado el Tema 2, aún en el caso que éste tenga poco que ver con aquel. Esto es marcadamente cierto con este modelo de enseñanza. He repetido innumerables veces "Esto no está bien, pero no os preocupéis es al final de curso que tiene que estar bien. Tenéis todo el año para aprenderlo". Se lo decía a los alumnos, pero también me lo repetía a mí mismo. Esta paciencia se ha visto ampliamente recompensada. Saber explicar un conocimiento es un proceso global y aunque te puedes ir fijando en aspectos parciales, no hay claras etapas a cumplir.

Es bueno que trabajen en equipo. Puse a los alumnos a trabajar por parejas para reducir la carga de trabajo, tanto la suya, como sobre todo la mía. Ha sido una buena idea, pero no por la reducción, sino porque los alumnos han tenido con quien compartir y comparar sus ideas, con quien discutir cómo hacer las cosas, a quien explicar y ser explicado. Dudo que se hubiera obtenido tan buen resultado si los trabajos se hubieran desarrollado individualmente. También he promovido el trabajo entre equipos –por ejemplo, haciendo intercambios de índices comentados– pero he visto que las ventajas de comunicación se daba esencialmente entre miembros del equipo.

Uso de ejercicios de escritura. A principio de curso, las clases de escritura consistían esencialmente en una pequeña charla y después hacerles escribir, y a veces sacar alumnos a la pizarra para ver lo que habían hecho. El resultado no fue demasiado bueno. Con el tiempo fui evolucionando hacia una lista de ejercicios concretos que los alumnos debían hacer en tiempo de clase. Los ejercicios propuestos iban en consonancia con el tema de escritura del ciclo y están diseñados para que se hagan sobre los informes a entregar. Así se aprovecha mejor el tiempo de clase y se disminuye la carga de trabajo del alumno. Una lista de algunos de los ejercicios propuestos puede verse en la Figura 2.

> Dada una transparencia con muchos tipos de letra clasifícalas por orden de `belleza'.
>
> Resume tu trabajo en 10 líneas. Pásalo a otro grupo. Este grupo escribe dos preguntas que le gustaría ver contestadas. Se repite el proceso dos o tres veces y se devuelve la hoja al grupo original. En vista de las preguntas, reescribe el resumen.
>
> A partir de una sección del borrador del trabajo se escribe en una hoja la parte introductoria de cada párrafo. Reescribe las frases transcritas como un único párrafo. ¿Es este párrafo un buen resumen del texto?¿Habría que añadir o eliminar alguna frase? A partir de las frases que has añadido o eliminado añade o elimina los párrafos que faltan o sobran. ¿Mejora el texto?
>
> A partir de un esquema `de títulos' pasa a un esquema `de ideas'. Por ejemplo, pasa de "La cache" a "La cache es el corazón de la jerarquía de memoria. De su buen diseño depende el buen rendimiento del ordenador".
>
> Simplifica (elimina algo) de cada diagrama que has hecho. Decide si la complejidad añadida del diagrama original vale la pena.

Figura 2. Ejercicios de escritura

Reuniones de evaluación. Una de las ideas más útiles del curso ha sido la de reunirme con los grupos para evaluar los trabajos. En el ciclo 1 entregué a cada grupo un informe de una o dos páginas en la que comentaba y evaluaba su documento. Pero me era imposible escribir en el informe todo lo que me hubiera gustado decirles, conque a partir del segundo ciclo decidí reunirme con cada grupo para leer con ellos el trabajo y poder discutirlo en detalle. Los alumnos aprendieron muchísimo en estas reuniones. No sólo pude indicar con todo detalle los puntos fuertes y flojos de sus escritos, sino que podíamos estudiar alternativas, responder a dudas, etc. Las primeras reuniones fueron muy largas

(entre una y dos horas, por grupo), las últimas, mucho más reducidas (entre 15 y 30 minutos por grupo).

Seguir los consejos de los alumnos. A lo largo de todo el curso hice mucho caso a las sugerencias que me fueron haciendo los alumnos. Lo hice más que nada porque aunque sabía bien qué es lo que *quería* conseguir, no tenía nada claro qué es lo que se *podía* conseguir. A posteriori, ha sido muy positivo hacerles caso: se sintieron involucrados en el proyecto, y aprendieron mucho más de las experiencias. Una ventaja añadida es que si alguna idea salía mal –todo era experimental, al fin y al cabo–, no perdían la fe en el proyecto.

Fue muy importante que el riesgo que representaba esta actitud se hiciera explícito. A los alumnos se les dijo claramente que esto era una experiencia nueva, qué había un riesgo y por lo tanto podía salir mal. Les aseguré que no iban a salir perjudicados si el proyecto no tenía los resultados esperados. Esto les animó a experimentar y a ser más creativos, lo cual es otro beneficio de este curso.

5.- CONCLUSIONES

He presentado un nuevo método de aprendizaje que he usado durante el curso 1999-2000 en la asignatura Estructura de Computadores de tercer curso de la Ingeniería Técnica de Informática de Sistemas. El método consiste en hacer que los alumnos aprendan los contenidos de la asignatura escribiendo sobre ella. En las clases, en vez de dedicar el tiempo a explicar el contenido de un tema, se dedica a estudiar cómo puede exponerse el mismo. Al final del año los alumnos han escrito cinco informes en los que explican los fundamentos de la asignatura. El resultado obtenido en conjunto puede considerarse muy bueno, tanto desde el punto de vista del profesor, como de la evaluación hecha por los alumnos.

Con este método se consigue un aprendizaje con buenas raíces, que permite al alumno establecer conexiones entre un tema y otro y entre esta asignatura y otras. Aumenta la independencia del alumno que ha de aprender buscar información, comprenderla y exponerla. Como no podía ser menos, este método incrementa la capacidad de comunicación –tanto de lectura como de escritura– de los alumnos.

Para poder ser efectivos hay que tener en cuenta que el método refuerza el conocimiento conceptual, pero que deja un tanto de lado otros, como la capacidad de la resolución de problemas. Otra dificultad que hay que vencer es buscar formas de hacer que los alumnos, manteniendo su iniciativa de elección de qué escribir y cómo escribir, no abusen de esta libertad para evitar aprenderse las partes más difíciles de la asignatura.

Aprender a través de la escritura requiere paciencia y trabajo tanto del profesor como de los alumnos, pero se consigue un tipo de aprendizaje que es muy difícil de conseguir con el método tradicional.

REFERENCIAS

[1] J. Miró Julià. *Aprendizaje a Través de la Escritura.* Ponencias de las V Jornadas sobre la Enseñanza Universitaria de la Informática, Jenui'99. La Almunia de Doña Godina, Octubre 1999.

FORMULACIÓN DE LOS OBJETIVOS DE UNA ASIGNATURA EN TRES NIVELES JERÁRQUICOS*

Juan J. Navarro, Miguel Valero-García,
Fermín Sanchez y Jordi Tubella

Departamento de Arquitectura de Computadores
Universidad Politécnica de Cataluña, Barcelona.
e-mail: {juanjo, miguel, fermin, jordit}@ac.upc.es

Resumen: Una adecuada formulación de los objetivos didácticos de una asignatura es una poderosa herramienta en el proceso de enseñanza-aprendizaje. En este artículo:

- Explicamos qué son los objetivos didácticos de una asignatura poniendo ejemplos de la enseñanza de la informática, y en particular, de asignaturas básicas de arquitectura de computadores (en los textos sobre pedagogía no es frecuente encontrar ejemplos sobre estos temas).
- Proponemos una formulación multinivel de los objetivos didácticos de una asignatura, que está muy relacionada con la estructura de la organización de nuestros centros de enseñanza.
- Mostramos algunas posibles ventajas de la herramienta.
- Proponemos una estrategia para comenzar a redactar los objetivos de una asignatura de forma fácil.

1.- INTRODUCCIÓN

Tener definidos los objetivos didácticos de las asignaturas es de gran ayuda para todas las personas involucradas en el proceso de enseñanza-aprendizaje (responsables académicos del centro y del departamento, profesores y alumnos). A pesar de que se ha escrito mucho sobre el tema en ámbitos pedagógicos, en nuestra opinión, la formulación de objetivos es una herramienta poco usada en la enseñanza de las ingenierías y en particular en las ingenierías de informática. Ahora que estamos inmersos, a nivel nacional, en un proceso de revisión de los planes de estudio, es un buen momento para introducir el uso de los objetivos didácticos en nuestras asignaturas.

Definición

Un objetivo didáctico expresa con claridad lo que esperamos que el alumno haya aprendido al acabar el curso. Informa sobre el resultado o el cambio esperado en el alumno como consecuencia del proceso de enseñanza-aprendizaje (conoce lo que no conocía, entiende lo que no entendía, hace lo que no sabía hacer...)[1], [2].

La lista de objetivos de una asignatura viene precedida por una sentencia del estilo:

El alumno debe ser capaz de:

Cada objetivo es un enunciado que contiene:

- Un verbo que expresa el resultado, la acción, el comportamiento mediante el cual el alumno manifiesta lo aprendido.
- Uno o varios complementos que indican el tema sobre el que se espera el resultado y las circunstancias (modo, causa, finalidad, etc.) en las que debe darse el resultado.

Ejemplos de objetivos didácticos bien formulados son:

Enunciar la ley de Amdahl.
Definir el concepto de la memoria cache.
Aplicar los postulados del álgebra de Boole para pasar de una expresión lógica cualquiera a una expresión en suma de productos que tenga el mínimo número de operadores.
Determinar los cambios producidos en el estado del computador (registros y memoria) al ejecutar una secuencia de cómo máximo 5 instrucciones del lenguaje ensamblador del Intelx86, con la única ayuda del manual de instrucciones.

2.- NIVELES DE CONCRECIÓN

Dependiendo del verbo y los complementos utilizados en la formulación de un objetivo, éste puede ser más o menos **General / Específico**. Podemos referirnos al nivel de concreción de un objetivo según lo concreto de la acción, del tema o de las circunstancias.

a) De la acción

Un objetivo es específico con respecto a la acción si está formulado con un verbo válido para hacer una pregunta o proponer un ejercicio. Por ello, un objetivo específico sólo admite una interpretación y es directamente evaluable (es decir, se parece a una pregunta de examen pero sin datos concretos)[1]. Los objetivos que se han usado como ejemplo en la sección anterior son objetivos específicos: se puede pedir al alumno que enuncie la ley de Amdahl o que defina lo que es una memoria cache.

Un objetivo general define un marco de referencia, indica la dirección del aprendizaje, pero admite varias interpretaciones. Por ello, no expresa con precisión lo que el alumno tiene que saber hacer para demostrar que ha cumplido el objetivo[1]. Que el alumno comprenda el concepto de memoria cache es un ejemplo de objetivo general. No se puede formular una pregunta de examen con el verbo comprender (*comprenda Ud. el concepto de memoria cache*). Verbos típicos correspondientes a objetivos generales son: conocer, entender, comprender, saber, dominar, ser consciente, etc.

b) Del tema

Según el complemento directo que acompaña al verbo, el tema del que trata el objetivo puede ser más o menos general / específico, es decir, estar más o menos detallado, acotado. El tema circuitos digitales es más general que el tema bloques combinacionales y éste más que el tema multiplexor.

c) De las circunstancias

El objetivo también puede ser más o menos concreto en función de la existencia o no de otros complementos que aclaran las circunstancias en las que se va a dar la acción, la conducta. A continuación ofrecemos dos ejemplos en orden creciente de concreción respecto a las circunstancias:

> *Obtener la tabla de verdad de un circuito combinacional a partir de una expresión lógica de su funcionamiento*
> *Determinar, en menos de cinco minutos, los cambios producidos en el estado del computador al ejecutar una secuencia de cómo máximo 5 instrucciones de lenguaje ensamblador del Intelx86, usando únicamente el manual de instrucciones.*

3.- FORMULACIÓN JERÁRQUICA MULTINIVEL

Proponemos una formulación jerárquica multinivel de los objetivos didácticos de una asignatura en la que se relacionan el nivel de concreción de los objetivos con los niveles jerárquicos de la organización docente en la universidad.

Los entes involucrados en la organización de la docencia de una asignatura son:

- El centro docente (facultad, escuela) responsable del plan de estudios al que pertenece la asignatura.
- El departamento asignado para impartirla.
- El profesor (o grupo de profesores) que la imparte.
- El alumno que la cursa.

Entendemos los objetivos de una asignatura, a un cierto nivel de formulación, como un contrato o acuerdo entre dos de estos entes vecinos. Por ejemplo, cuando el centro encarga una asignatura a un departamento, ambos deberían acordar qué deberán saber los alumnos al final del curso. Este acuerdo o contrato es la lista de objetivos de la asignatura a nivel centro-departamento.

Con esta visión, proponemos una formulación de los objetivos de una asignatura en tres niveles (tres contratos):

> Nivel 1. Centro-departamento
> Nivel 2. Departamento-profesor
> Nivel 3. Profesor-alumno

Conforme aumenta el nivel, aumenta también:

- el nivel de concreción de la acción, tema y circunstancias,
- el número de objetivos y
- la facilidad para hacer modificaciones en la formulación de los objetivos de un año para otro.

La formulación es jerárquica porque a cada objetivo de un nivel determinado le corresponden uno o varios del siguiente nivel que lo concretan.

Veamos ahora algunos aspectos particulares de cada uno de los tres niveles.

a) Nivel centro-departamento

Los objetivos a este nivel se derivan de los objetivos formativos del plan de estudios al que pertenece la asignatura. Son un contrato entre el centro y el departamento y cualquier modificación de su formulación debe ser aprobada en el órgano pertinente del centro. Para que sean operativos deben ser pocos y generales. Un ejemplo:

> *Comprender la estructura interna, a nivel de análisis y síntesis de circuitos digitales, de un procesador RISC sencillo.*

b) Nivel departamento-profesor

Los objetivos a este nivel representan un contrato entre el departamento y los profesores que imparten la asignatura. Un ejemplo de objetivo a este nivel, que podría ser uno de los objetivos que concretan el ejemplo anterior, es:

> *Sintetizar sistemas secuenciales basados en el modelo de Mealy*

Cualquier modificación de estos objetivos debe ser aprobada por el órgano correspondiente del departamento, puesto que puede afectar a otras asignaturas del propio departamento y porque el departamento debe velar para que se cumpla el contrato que tiene firmado con el centro. En el ejemplo anterior el departamento podría plantearse, posiblemente promovido por un profesor de la asignatura, pasar del modelo de Mealy al de Moore. Como esta modificación no afecta al objetivo más general correspondiente de nivel anterior, podría hacerse sin involucrar al centro.

c) Nivel profesor-alumno

En este nivel, los objetivos se formulan con el máximo nivel de concreción y representan un contrato entre el profesor y sus alumnos. Este contrato acarrea un compromiso para el profesor por lo que respecta a la evaluación ("si aprendéis todo esto entonces aprobaréis la asignatura") y también en cuanto a los métodos docentes a utilizar, ya que deberá prever en su plan docente actividades apropiadas para que los alumnos puedan alcanzar los objetivos formulados. Uno de los posibles objetivos a este nivel, que se corresponde con el ejemplo anterior, podría ser:

> *Diseñar sistemas secuenciales basados en el modelo de Mealy de cómo máximo 2 señales de entrada y 4 estados, usando el mínimo número de puertas NAND y biestables tipo JK.*

Después de un tiempo de impartir el curso, los profesores de la asignatura podrían decidir que los biestables tipo D resultan más fáciles de entender que los JK, y modificar entonces este objetivo específico sin que este cambio tenga que ser aprobado por el departamento, entendiendo que no afecta al objetivo de nivel anterior correspondiente.

Los objetivos, a todos los niveles, pueden negociarse entre las partes, incluso en el caso profesor-alumno. Esta estrategia aumenta la motivación del estudiante y su compromiso en relación al proyecto de aprendizaje que representa para él la asignatura.

4.- VENTAJAS DE LA FORMULACIÓN DE OBJETIVOS

La formulación jerárquica de los objetivos de una asignatura es de gran importancia para todos los implicados en el desarrollo de la asignatura: alumno, profesor y responsables académicos tanto del departamento como del centro. En nuestra opinión, esta herramienta puede conseguir motivación, compromiso y satisfacción de todas las partes involucradas y contribuir a mejorar la calidad de la docencia.

Si los objetivos didácticos están bien formulados, entonces

el alumno:

- tiene claro, desde el comienzo, qué es lo que se espera de él, lo cual disminuye la tensión de cara al examen porque hay menos terreno para las sorpresas y le permite enfocar los esfuerzos en la dirección correcta desde el primer momento,
- tiene todos los elementos de juicio para autoevaluar su proceso de aprendizaje, revisando periódicamente la lista de objetivos y preguntándose, por ejemplo, si sabe ya diseñar circuitos secuenciales,
- está preparado, desde el primer momento, para diferenciar lo importante de lo secundario (con respecto a lo que se le explica en clase o en los libros),
- está más motivado porque le resulta fácil ver el resultado final como asequible;

el profesor:

- tiene autonomía suficiente para adaptar los objetivos de la asignatura a nivel profesor-alumno a su criterio (mientras respete los objetivos de nivel departamento-profesor,
- tiene más facilidad para seleccionar los métodos más adecuados para cada uno de los objetivos perseguidos, lo cual facilita una recomendable diversificación de estrategias,
- tiene claro cómo deben ser los exámenes,
- tiene mucho más fácil la coordinación con el resto de profesores de la asignatura y con los profesores de otras,
- puede evaluar su tarea docente por el porcentaje de alumnos aprobados, siempre que los exámenes los pongan otros profesores usando adecuadamente los objetivos a nivel profesor-alumno;

el responsable académico del departamento y del centro:

- puede identificar posibles lagunas o solapamientos en el plan de estudios,
- tiene todos los elementos de juicio para tomar decisiones sobre la conveniencia o no de añadir, quitar o modificar objetivos, prerrequisitos, etc., de algunas asignaturas, de forma que las decisiones sean coherentes con los objetivos del plan de estudios.

5.- CÓMO EMPEZAR

El proceso lógico de formulación de objetivos sería redactar primero los de nivel centro-departamento, refinar después cada uno de ellos para obtener los de nivel departamento-profesor y finalmente obtener los de nivel profesor-alumno. No obstante, este proceso que va de lo más general a lo más específico resulta difícil cuando no se tiene agilidad en la formulación de objetivos (ésa al menos ha sido nuestra experiencia).

Proponemos ahora una estrategia que resulta mucho más fácil para empezar, porque realiza el camino en sentido inverso, partiendo de elementos muy concretos y conocidos. Se trata de tomar los exámenes de la asignatura correspondientes a los últimos años y clasificar las preguntas por tipos (normalmente no hay demasiados tipos diferentes de preguntas). Luego, para cada tipo de pregunta resulta relativamente fácil redactar un objetivo específico que se corresponda con tal tipo de pregunta.

Es posible que este procedimiento produzca alguna sorpresa, especialmente si se tiene en cuenta la frecuencia y el peso de cada tipo de pregunta en los exámenes. Por ejemplo, puede ocurrir que un objetivo tenga una presencia en los exámenes mucho mayor de lo que parece razonable, mientras que otros que habría que considerar como importantes apenas se preguntan nunca. Por tanto, este procedimiento necesitará normalmente una revisión, que puede representar eliminar algunos objetivos específicos y añadir otros. Se obtiene así una primera versión de los objetivos de nivel profesor-alumno, a partir de la cual pueden obtenerse los de niveles más generales.

6.- CONCLUSIONES

Con este artículo hemos pretendido acercar el tema de la formulación de objetivos de una asignatura a la enseñanza de la informática, poniendo ejemplos de objetivos didácticos del área de Arquitectura y Tecnología de Computadores.

Proponemos una formulación jerárquica en tres niveles que puede ayudar a clarificar las relaciones entre los implicados en la asignatura: el centro, el departamento, los profesores y los alumnos.

También proponemos una estrategia que puede ser útil para empezar a formular los objetivos de una asignatura y que comienza analizando los exámenes de los últimos años.

El tema de los objetivos va más allá de lo expuesto en este breve artículo. Todos los ejemplos que hemos utilizado pertenecen al ámbito del los conocimientos, pero existen también los ámbitos de las destrezas o habilidades y de las actitudes. Además, es muy importante la clasificación de los objetivos en función del nivel de competencia [3], por la incidencia que puede tener en los métodos docentes utilizados.

7.- REFERENCIAS

[1] Morales Vallejo, P. Los Objetivos Didácticos. Cuadernos Monográficos del ICE. Universidad de Deusto. Bilbao, 1995.

[2] Ramsden, P., Learning to teach in higher education. London, Routledge, 1992

[3] Bloom, B.S., Hastings, J.T. y Madaus, G.F., Taxonomía de los objetivos de la educacion. Tomo I (conocimientos) y Tomo II (dominio afectivo), Ed. Marfil, Alcoy 1973.

APLICACIÓN DE MÉTODOS HIPERMEDIA EN LA IMPARTICIÓN DE UNA ASIGNATURA DE FUNDAMENTOS DE INFORMÁTICA.

Nieves Pavón Pulido[1], Jesús María Álvarez Llorente[1]

[1] *Profesor Asociado del Departamento de Ingeniería Electrónica, Sistemas Informáticos y Automática.*
e-mail: **npavon@uhu.es**

RESUMEN: En el presente artículo se presenta una nueva experiencia en la docencia de una asignatura de Fundamentos de Informática en 1° de I. T. Industrial. Mostramos el desarrollo de un ambicioso y extenso temario adaptado a las necesidades de cada especialidad y la aplicación de técnicas hipermedia en el laboratorio, de modo que se proporciona una mayor motivación a los alumnos y se facilita la comprensión de unos conceptos de gran complejidad en el ámbito de la programación de sistemas utilizando el lenguaje C.

1.- INTRODUCCIÓN.

En el marco del nuevo Plan de Estudios de Ingeniería Técnica Industrial en la Universidad de Huelva podemos encontrar una alta carga docente de asignaturas relacionadas, directa o indirectamente, con la Informática.

El plan de estudios contempla 4 especialidades: Mecánica, Química, Electricidad y Electrónica Industrial. Esta última, es la que más carga tiene de asignaturas relacionadas con Electrónica, Control e Informática en general, a lo largo de toda la carrera. Sin embargo, todas las especialidades comparten una asignatura básica de iniciación: Fundamentos de Informática.

Después de realizar un exhaustivo análisis de los planes de organización docente de asignaturas similares en otras universidades españolas, llegamos a la conclusión de que en la mayoría sólo se ofrecen conceptos básicos de ofimática. Estos conceptos no resultan útiles en la formación del alumnado de I. T. Industrial de la Universidad de Huelva de acuerdo al planteamiento global establecido para el resto de asignaturas de la carrera concernientes al Departamento de Ingeniería Electrónica, Sistemas Informáticos y Automática (en adelante D.I.E.S.I.A.), principalmente, en la especialidad de Electrónica Industrial, ya que, frente a estos conceptos típicos, se busca introducir a los alumnos en el mundo de la programación de sistemas en C. Esta preparación es muy necesaria, pues supone la base donde se apoyan otras asignaturas de cursos siguientes tales como Informática Industrial, Automatización Industrial, Electrónica Digital, etc.

El montaje de una asignatura con un temario que plantea unos objetivos muy ambiciosos, ya que los conceptos de programación a impartir son bastante extensos y de gran nivel, implica concebir un material para la misma que cumpla los siguientes requisitos:

- Incremento de la motivación del alumnado.
- Dotación de una documentación detallada de todo lo impartido, tanto en la parte teórica como en el laboratorio.
- Presentación de unas tutorías continuadas mediante el uso de técnicas hipermedia.

Veamos, pues, en los siguientes apartados una descripción de los objetivos pretendidos en la asignatura, del material aportado y de los resultados obtenidos tras aplicar las técnicas de innovación docente mencionadas.

2.- OBJETIVOS DE LA ASIGNATURA Y P.O.D.

En el seno del D.I.E.S.I.A. se ha planteado un temario para Fundamentos de Informática que cumple perfectamente los objetivos propuestos:

- Dotar a los alumnos de una base robusta y de alto nivel en programación estructurada y modular para su aplicación en el diseño de programas de acceso, gestión y control de dispositivos.

- Proponer y estudiar técnicas de diseño descendente para llevar a cabo la resolución de problemas de forma eficiente.

- Presentar el lenguaje C como un lenguaje estructurado, adecuado y potente para realizar programas y aplicaciones orientados principalmente al control y la gestión de sistemas y dispositivos.

Se observa, por tanto, que el temario no sólo abarca conceptos típicamente propios de desarrollo de aplicaciones de gestión, sino también amplios conocimientos sobre programación de sistemas.

Para llevar a cabo la implementación de la parte práctica de la asignatura se eligió el compilador de *Borland Turbo C 2.0* para *MS-DOS*, por su simplicidad de manejo, flexibilidad para realizar programas que pueden acceder al hardware directamente y su facilidad de distribución, ya que se trata de un software liberado por la empresa, que puede ser descargado de la página Web de la misma mediante un sencillo protocolo de afiliación a la "*Comunidad Borland*", totalmente gratuito y sin compromiso.

Los puntos básicos que componen el temario teórico de la asignatura se enumeran, a continuación:

1 Conceptos básicos de algorítmica. Proceso de compilación.

2 Elementos básicos de programación: Palabras reservadas. Tipos, variables y constantes. Punteros. Estructura de la memoria. Coherencia de tipos. Operaciones básicas aritmético-lógicas y de entrada-salida.

3 Control de flujo del programa y subprogramas y modularidad.

4 Estructuras de datos complejas y asignación de memoria dinámica. Vectores y tablas. Estructuras. Punteros a estructuras. Campos de bits.

5 Modelo de tres capas: hardware, BIOS, DOS. Arquitectura Intel 8086.

6 Programación del sistema en C. Gestión eficiente de la memoria RAM. Acceso directo a memoria de vídeo. Lectura–escritura de puertos.

7 Operaciones sobre ficheros e interrupciones.

Dado que la asignatura consta de 6 créditos, con una distribución homogénea de créditos teóricos y prácticos (2 horas semanales de teoría + 2 horas semanales de laboratorio), cabe pensar que el temario es demasiado extenso y complicado para ser impartido a alumnos que tienen su primer contacto con la Informática. El reto ha estado, pues, en encontrar una metodología docente que permita transmitir todos estos conceptos de un modo sencillo y motivador.

3.- METODOLOGÍA

Motivación, transparencia y sencillez han sido los 3 puntos sobre los que ha girado la docencia de la asignatura.

La transparencia y sencillez se consigue mediante el desarrollo de clases magistrales esquemáticas que profundizan en los aspectos más relevantes del temario. En nuestro caso, puntos normalmente estudiados al final del temario, como el uso de punteros y administración de memoria dinámica, se han impartido en las primeras semanas del cuatrimestre con el fin de que los alumnos trabajasen y se familiarizasen desde el principio con conceptos tradicionalmente "difíciles".

Hemos considerado que el alumno sólo necesita memorizar una serie de ideas básicas para inferir el resto de nociones mediante el estudio práctico de casos concretos, y esto se ha tenido en cuenta a la hora de desarrollar el material utilizado en las clases magistrales y el resto de herramientas hipermedia diseñadas.

La parte teórica ha sido revisada al final del cuatrimestre mediante un test que ha examinado los conceptos teórico-prácticos adquiridos por el alumno. La demostración de que el alumno domina dichos conceptos verifica que está capacitado para resolver problemas de cualquier tipo mediante el uso de técnicas de diseño estructurado y descendente (refinamientos sucesivos).

La parte práctica ha consistido en el desarrollo de una aplicación grande en el lenguaje de programación C con el compilador anteriormente señalado. La revisión de esta parte práctica ha exigido una defensa oral de la misma. Con la realización de la práctica el alumno se ha enfrentado a la construcción de un sistema real que soluciona un problema que requiere un análisis estructurado adecuado. Además, se ha exigido una documentación precisa y ajustada al método de diseño e implementación seguido.

La propuesta de práctica ha sido diferente para cada especialidad, ya que se han buscado problemas relacionados con cada una en particular, con el objetivo de aumentar la motivación y disposición de los estudiantes hacia la parte práctica de Fundamentos de Informática. Por ejemplo, a los alumnos de Electrónica Industrial se les ha propuesto el diseño de una aplicación de simulación de un IRC («chat»)

a través del puerto serie, en cambio, a los alumnos de Química se les ha propuesto una práctica relacionada con la resolución automática de problemas de formulación.

En cuestiones de calificación global de la asignatura, para todas las especialidades se ha seguido un común denominador: la asignatura se considera aprobada cuando el alumno supera ambas partes por separado con una calificación mínima de 5 puntos. La nota obtenida se calcula como la media de la nota de prácticas y la nota de teoría, ya que ambas partes son igualmente importantes.

4.- MATERIAL DE APOYO DESARROLLADO.

Como en muchas otras asignaturas, se ha desarrollado un conjunto de apuntes y hojas de ejercicios que facilitan el estudio y seguimiento de las clases magistrales. Sin embargo, los apuntes y herramientas tradicionales no son suficientemente válidas cuando hablamos de una asignatura con unos contenidos tan prácticos. Por ejemplo, es posible dejar apuntes escritos en fotocopiadora sobre cualquier tema pero, ¿qué ocurre si queremos dejar el resultado de la ejecución de un programa, la simulación de la construcción de un cable, o la muestra de cómo debe funcionar una práctica concreta?

En este sentido, los métodos tradicionales se nos quedan cortos, de ahí, la idea de introducir material hipermedia en la docencia de la asignatura. Y no sólo como un complemento, sino como el eje principal de estructuración de todas las clases prácticas en el laboratorio real.

Este material consiste en una página WEB, que nos sirve como portal de un *Laboratorio Virtual* de programación que el alumno puede visitar en cualquier momento desde cualquier sitio. La existencia de este *Laboratorio Virtual* no ha eximido a los alumnos de tener que asistir a las clases prácticas, sino todo lo contrario, dicho laboratorio tiene todo su sentido en el propio laboratorio real.

Las presentaciones prácticas se encuentran descritas y simuladas en el *Laboratorio Virtual*, y el alumno puede acceder a cualquier material (incluidos los apuntes de las sesiones teóricas), en cualquier momento, seguir el hilo de la clase a través de su propio puesto en la sala y llevarse a casa el resumen de todo lo que se ha explicado para consultarlo nuevamente: anotaciones, simulaciones, ejecutables, código fuente, etc.

Además, ofrece la posibilidad de llevar a cabo *Tutorías Virtuales* con los profesores encargados de la asignatura, entregar ejercicios y prácticas a través de Internet y completar y corregir automáticamente hojas de ejercicios mediante el uso de formularios interactivos.

a.) Esquema general del laboratorio virtual.

Nuestro *Laboratorio Virtual* está construido mediante una página WEB que consta de las siguientes partes fundamentales:

- *Descripción*, *Objetivos* y *Normativa* ofrecen una visión general sobre cómo se estructura la asignatura y cuáles son las normas básicas de asistencia, comportamiento y evaluación de la misma.

- *Bibliografía* ofrece información sobre libros y revistas relacionados con la programación estructurada y modular, la programación de sistemas a bajo y alto nivel, y el lenguaje C.

- *Temario* ofrece enlaces a páginas que contienen todos los temas impartidos en formato HTML. Dichos temas también pueden ser descargados en formato DOC desde el enlace *Descarga.*

- *Prácticas* ofrece la posibilidad de descargar todos los códigos fuentes y ejecutables realizados en las clases prácticas así como los propuestos en las clases teóricas. El alumno puede ver el resultado de la ejecución de cualquier ejemplo desde Internet. También puede repasar los apuntes, transparencias y presentaciones de cualquier sesión práctica (ver Figura 1)

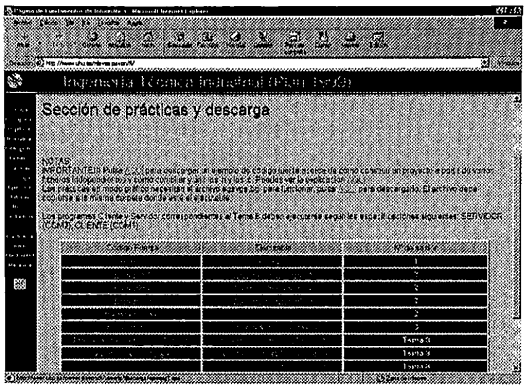


Figura 1: Sección de *Prácticas* del Laboratorio Virtual.

- *Ejercicios* ofrece un conjunto de hojas de ejercicios que pueden ser resueltos a través de Internet. Tenemos dos modalidades:

  - Ejercicios que pueden ser enviados a través de Internet para que el profesor pueda corregirlos a distancia. Permiten al profesor seguir el rendimiento de cada alumno en particular de forma continuada.

  - Ejercicios tipo test, que se valoran a través de la página WEB en tiempo real mediante la simple pulsación de un botón, utilizando un sencillo formulario interactivo (ver Figura 2). El alumno puede conocer los fallos y aciertos de forma inmediata. Además el profesor puede publicar nuevas hojas de ejercicios desde su propio despacho o, si fuera necesario, desde cualquier punto de acceso a Internet.

- *Tutorías Virtuales* ofrece al alumno la posibilidad de formular preguntas a través de un formulario que es enviado automáticamente por correo electrónico al profesor. Las consultas pueden ser atendidas de forma pública, es decir, mediante la actualización de la propia página en el espacio reservado para las respuestas, o bien de forma privada, mediante un mensaje a la dirección de correo electrónico proporcionada por el alumno en el texto del mensaje (ver Figura 3).

- Cada especialidad tiene sus propios enlaces donde se publican enunciados particulares, horarios, correcciones de exámenes y otros enlaces de interés.

- Además, a través de la página, es posible consultar estadísticas sobre calificaciones, encontrar novedades y noticias relacionadas con el desarrollo de la asignatura, realizar búsquedas de material, etc.

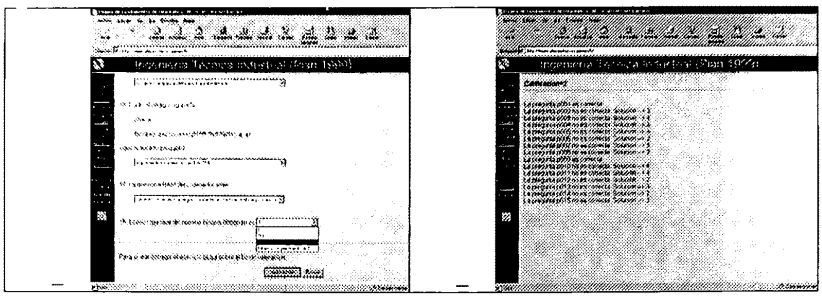


Figura 2: Ejemplo de realización y corrección de un ejercicio de forma automática.

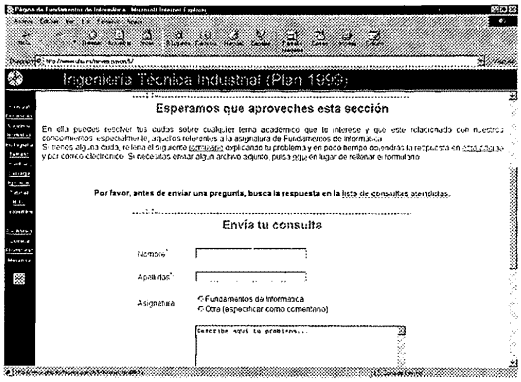


Figura 3: Sección de Tutorías Virtuales del Laboratorio Virtual.

5.- EVALUACIÓN DE LA ASIGNATURA POR PARTE DEL ALUMNADO. CONCLUSIONES.

Se ha realizado una evaluación de la asignatura al final del cuatrimestre, por parte del alumnado. El test de evaluación ha consistido en un conjunto de preguntas sobre la calidad de las clases magistrales, la idoneidad del temario propuesto y la utilidad de los materiales hipermedia.
En conjunto:

- Casi el 90% del alumnado coincide en que la asignatura ha sido difícil, pero, ante todo, útil y entretenida.

- Aproximadamente un 50% admite que el nivel de exigencia debe mantenerse frente a un 40% que cree que debe relajarse y un 10% que debe aumentarse.

- El 100% del alumnado considera que la aplicación de técnicas de innovación docente, en concreto técnicas hipermedia, aumentan la motivación frente a la asignatura y facilitan su estudio y dedicación.

- Las zonas más visitadas de la página han sido el *Tablón de apuntes*, la sección de *Prácticas* y de *Descarga*, la sección de *Ejercicios* y, en menor medida las *Tutorías Virtuales*.

- Casi todos los alumnos coinciden en que el uso de la página aumentará en función de la disposición para acceder a Internet (cuando se produzca un mayor abaratamiento de costes de conexión desde casa), ya que la mayoría de ellos ha accedido a la página exclusivamente desde los laboratorios que la Universidad de Huelva pone a su disposición.

6.- LÍNEAS FUTURAS.

Las líneas futuras que se pretenden seguir van encaminadas en la mejora y ampliación del *Laboratorio Virtual* mediante:

- Adición de nuevas secciones para mejorar las Tutorías Virtuales a través de sistemas de grupos de noticias e IRC (aprendizaje colaborativo).

- Creación de un marco de compilación y ejecución de pequeños ejercicios a través de la propia página WEB, ya que muchas veces los alumnos no pueden disponer del compilador adecuado en salas de ordenadores de libre acceso. La idea consiste en disponer de una sección donde, para un enunciado propuesto (por ejemplo, realizar una función en C que calcule el factorial de un número), el alumno pueda escribir su solución (el código fuente de dicha función) a través de un formulario dinámico, que será compilada y probada de forma remota, devolviendo el resultado a través de la propia página WEB.

7.- BIBLIOGRAFÍA.

[Mic 96] Tischer, Michael. Jennrich, Bruno. *"PC Interno 5. Programación de Sistemas"*. Marcombo, 1996.

[Joy 96] Joyanes Aguilar, Luis. "Fundamentos de programación: Algoritmos y estructuras de datos". McGraw-Hill, 1996.

[CONIED 99] Actas del CON.I.ED. *"Informática y Educación en el siglo XXI"*. M. Ortega y Manuel Bravo Editores, 1999.

[Her 92] Schildt, Herbert. *"Turbo C/C++. Manual de referencia"*. McGraw-Hill, 1992.

CALIOPE[10]: UNA ARQUITECTURA PARA APRENDIZAJE AUTÓNOMO COLABORATIVO EN ENTORNOS NO PRESENCIALES.

Miguel Rebollo, Vicente J. Julián, Carlos Carrascosa, Vicente J. Botti

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
e-mail: {mrebollo,vinglada,carrasco,vbotti}@dsic.upv.es

RESUMEN: La aplicación de las nuevas tecnologías a la pedagogía ha abierto un nuevo horizonte en los procesos de aprendizaje. En el presente artículo se presenta una arquitectura basada en el paradigma de sistemas multi-agente como una herramienta de apoyo a la creación de aulas virtuales. La arquitectura integra los elementos necesarios para la interacción de profesores y alumnos en un entorno asíncrono que facilite el aprendizaje activo colaborativo a través de experiencias no presenciales.

1.- INTRODUCCIÓN.

La aplicación de las Tecnologías de la Información y las Comunicaciones (TIC) a la pedagogía hace posible la evolución hacia nuevas formas de aprendizaje que rompen con la concepción de tiempo y espacio de la enseñanza tradicional presencial, posibilitando la creación de **aulas virtuales**, en las que el proceso de aprendizaje tiene lugar entre alumnos no sincronizados y geográficamente dispersos [Hazemi98].

La gran complejidad de las actividades de aprendizaje, junto con el alto grado de flexibilidad necesario para posibilitar el **aprendizaje autónomo** *de facto* (Boud, 1988), justifica la aplicación de nuevas tecnologías para gestionar adecuadamente todo el proceso y dar el soporte necesario tanto a profesores como alumnos. Lo que se propone en este artículo es aplicar las técnicas más recientes en el campo de la Inteligencia Artificial (IA), como son los sistemas multi-agente [Wooldridge95].

El resto del artículo se estructura de la siguiente manera: en la sección 2 se describen los sistemas multi-agente y se justifica su aplicabilidad al problema objeto de estudio. La sección 3 esboza la problemática asociada al aprendizaje autónomo colaborativo en entornos no presenciales. La arquitectura que se propone para afrontar este tipo de problemas queda recogida en la sección 4. La sección 5 justifica el empleo de XML como lenguaje de intercambio de información entre los distintos componentes del sistema. Por último, en la sección 6 se presentan las conclusiones.

[10] Co-operative and Active LearnIng for remOte exPEriences

2.- SISTEMAS MULTI-AGENTE.

Actualmente, podemos encontrar un gran número de propuestas en la literatura acerca de lo que es y lo que no es un **agente**, sin que ninguna de ellas haya sido plenamente consensuada por la comunidad científica. Una definición bastante aceptada es la propuesta en [Wooldridge95], según la cual, un agente viene definido por su flexibilidad, entendiendo por flexible que un agente sea:

- *Reactivo*, responda al entorno en que se encuentra.
- *Proactivo*, que sea capaz de intentar cumplir sus propios objetivos.
- *Social*, sea capaz de comunicarse con otros agentes mediante algún tipo de lenguaje.

En los trabajos más recientes relacionados con el tema se tiende a etiquetar a un agente en función del papel que desempeña. Así, por ejemplo, un agente con acceso a diversas fuentes de información y que es capaz de agrupar y manipular la información obtenida de dichas fuentes para ponerla a disposición del usuario se conoce como **agente de información**. Por otra parte, un agente que provee a su usuario de la asistencia necesaria para facilitar su interacción con una determinada aplicación, se conoce como **agente interfaz**. Estas definiciones son complementarias, pudiendo encontrar agentes de información que sirvan de interfaz a sus usuarios.

En la mayoría de las ocasiones, los agentes no son desarrollados de forma independiente sino como entidades que constituyen un sistema. A este sistema se le denomina **multi-agente** [Huhns98]. Las interacciones más habituales como son informar o consultar a otros agentes permiten a los agentes hablar entre ellos, tener en cuenta lo que realiza cada uno de ellos y razonar acerca del papel jugado por los agentes.

Para el problema que nos ocupa las características que podemos encontrar dentro de un sistema multi-agente son sumamente interesantes:

- El sistema tiene un marcado carácter distribuido.
- La autonomía que incorpora cada agente puede permitir un desarrollo independiente del proceso de aprendizaje.
- En el sistema debe existir una comunicación a través de Internet, la cual favorece la colaboración y/o coordinación entre dichas entidades.

3.- DESCRIPCIÓN DEL PROBLEMA

Cuando los adultos aprenden por sí mismos, siguen una secuencia de etapas, utilizan una serie de recursos, muestran un estilo determinado y valoran su aprendizaje de forma que éste resulta único para cada alumno. Para facilitar el aprendizaje, nuestra docencia debe seguir el flujo natural del proceso de aprendizaje de cada persona, en lugar de imponer nuestra propia secuencia de enseñanza. La **autonomía** en la enseñanza es una aproximación que requiere que los estudiantes asuman ciertas responsabilidades acerca de su propio aprendizaje, tomando la iniciativa en alguna de las tareas [Boud88].

El promover esta autonomía e independencia en el aprendizaje posibilita que no sea imprescindible la presencia física del profesor durante el proceso. Este hecho proporciona una base firme para defender la adecuación de dedicar cierta cantidad de tiempo al **aprendizaje no presencial**, entendiendo como tal el que realiza el alumno fuera del tiempo y el espacio dedicado a las clases y/o tutorías [Hazemi98].

Las aproximaciones a este tipo de aprendizaje no implican el aislamiento de los alumnos los unos del los otros, si no que suelen convertirse en estudiantes interdependientes, que trabajan y se ayudan entre sí (Cornwall, 1979). La aproximación en la que nos basaremos para soportar la colaboración en el aprendizaje es el **aprendizaje cooperativo** asistido por ordenador (CSCL —Computer Supported Co-operative Learning—) [Ovejero90]. Se trata de una aproximación emergente en las recientes investigaciones sobre el aprendizaje, basado en la idea de que el aprendizaje tiene lugar cuando los estudiantes tienen que explicar, desarrollar o justificar sus ideas para lograr un objetivo compartido. [Barros99].

Tal y como se puede observar en la Figura 1, en la enseñanza tradicional existe un hueco importante que queda sólo parcialmente cubierto por algunas actividades puntuales (tutorías, prácticas grupales, entornos colaborativos no presenciales). La originalidad de CALIOPE estriba en la inclusión de herramientas que apoyen el aprendizaje autónomo, cubriendo parte del hueco que dejan las aproximaciones clásicas para la enseñanza.

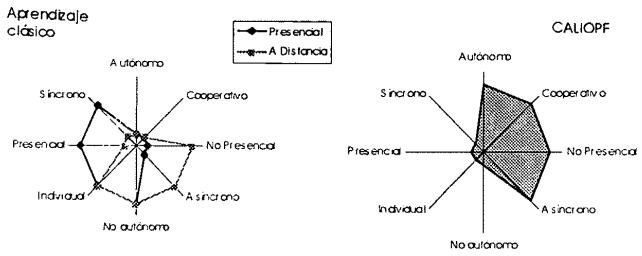


Figura 1. Estilos de aprendizaje cubiertos por CALIOPE

El objetivo que nos fijamos es desarrollar una herramienta informática, basada en la tecnología de sistemas multi-agente, que integre los tres tipos de aprendizaje: autónomo, no presencial y colaborativo. CALIOPE integra agentes que facilitan las tareas del profesor y del alumno en las tres fases del proceso de aprendizaje: planificación, actuación y evaluación.

4.- ARQUITECTURA DEL SISTEMA

El sistema está basado en el estandar propuesto por FIPA —*Foundation for Intelligent Physical Agents*— para el desarrollo de sistemas multi-agente [FIPA97]. Los motivos principales para escoger este estándar es que considera todos los aspectos necesarios para desarrollar sistemas multi-agente y los integra en un único modelo. El sistema consta de:

- un **asistente personal para los alumnos** (LPA —*Learner's Personal Assistant*—), el cual, a grandes rasgos, actúa como guía en el proceso de aprendizaje del alumno a lo largo del curso.

- un **asistente personal para el profesor** (TPA —*Teacher's Personal Assistant*—), realiza un seguimiento del curso en sus distintas fases (planificación, actuación y evaluación).

- una **plataforma** (AP) que dé soporte a toda la estructura de agentes, basada en el estándar FIPA.

- una **herramienta para el diseño de cursos** (CDT).

Existen una serie de agentes auxiliares que proporcionan soporte a los agentes anteriores. En una AP independiente, un agente wrapper proporciona todos los accesos a las bases de datos que mantienen la información sobre el curso. Un agente de ontologías (OA) mantiene los lenguajes para el intercambio de información, de manera que cualquier agente nuevo que se añada al sistema sea capaz de comunicarse o los cambios en un lenguaje no requieran la reprogramación de los agentes.

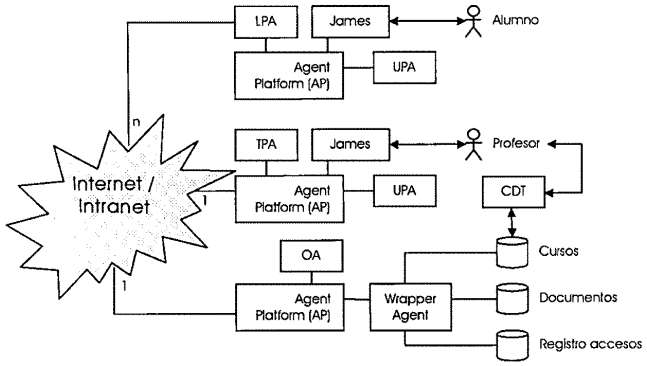


Figura 2. Arquitectura de CALIOPE

Tanto el LPA como el TPA colaboran con un conjunto de agentes que dan soporte a la interacción con usuarios humanos. En todas las AP debe existir un **agente de interfaz**, denominado *James* [Julián99], que actúa como un gestor de diálogos (UDMA) y convierte las demandas de los usuarios al lenguaje común de los agentes para su comunicación a través de la AP. Otro componente importante es un **agente de perfiles de usuarios** (UPA), que permite almacenar los modelos de usuario y aprender sus hábitos a partir del comportamiento observado. Este componente nos va a permitir asistir al alumno a lo largo de su proceso de aprendizaje, sugiriendo líneas de actuación en función del trabajo desarrollado hasta el momento, las directrices de su contrato de aprendizaje, etc...

a) Asistente personal para el alumno.

Este agente estará en continuo contacto con el alumno, siendo el encargado de guiar las tareas de aprendizaje de forma individualizada según las características del propio alumno. Entre las tareas asignadas, el agente debe ser capaz de realizar tareas de búsqueda de información relevante para el alumno frente a peticiones del mismo o de forma totalmente proactiva. El agente también debe facilitar las herramientas necesarias para la coordinación de trabajos (documentos comunes) con otros miembros del grupo al que pertenezca el alumno. Por último, analizará y observará el comportamiento

del alumno para sugerir líneas de actuación e información adicional que trate de complementar el proceso de aprendizaje del alumno.

b) Asistente personal para el profesor:

Este asistente permite al profesor realizar un seguimiento del progreso del curso que imparte. Para ello, el asistente le permite al profesor consultar información relativa al progreso del curso, mostrando tanto la evolución de cada grupo de trabajo, como la información relativa al trabajo individual de cada alumno. Mediante este asistente, el profesor puede adaptar dinámicamente el curso de acuerdo al progreso de los alumnos (eliminando alguno de los temas, cambiando la duración estimada de los mismos, o incluso añadiendo algún tema adicional siempre que éste se encuentre recogido previamente en la base de datos de cursos). Además, el asistente ofrece una serie de estadísticas para que el profesor pueda evaluar el aprovechamiento del curso (tanto grupal como individual).

c) Plataforma de agentes.

Una plataforma de agentes (AP) proporciona una infraestructura básica para los agentes. Cada agente debe registrarse en una plataforma para poder interactuar con el resto de agentes que se encuentran en esa o en otras plataformas.

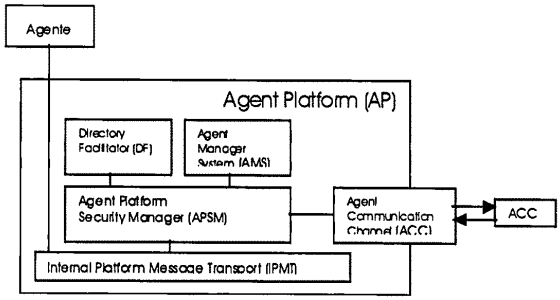


Figura 3. Componentes de una plataforma de agentes

Proporciona los siguientes servicios básicos:

- **Directory Facilitator** (DF). Se trata de un servicio de páginas amarillas. Los agentes registran sus servicios en el DF y lo consultan para saber qué servicios proporcionan otros agentes.
- **Agent Manager System** (AMS). Servicio de páginas blancas (direcciones físicas). Supervisa el acceso y uso de la AP.
- **Agent Platform Security Manager** (APSM). Es el responsable de mantener las políticas de seguridad para la plataforma.
- **Agent Communication Channel** (ACC) Encamina los mensajes de los agentes de una plataforma fuera de ella.
- **Internal Platform Message Transport** (IPMT) Se encarga del intercambio de mensajes dentro de la plataforma.

El entorno de aprendizaje debe permitir la comunicación entre todos los agentes que componen el sistema y que previamente se han registrado como usuarios autorizados. El tipo de mensaje y el protocolo empleado para inter-cambiar información depende del tipo de acto comunicativo que se realiza.

Además de los protocolos que propone FIPA, es necesario un protocolo específico que cubra las necesidades de la co-edición de documentos entre varias personas (independientemente de su naturaleza —resúmenes, trabajos de investigación informes, artículos, software, diseños, etc...—). Este protocolo se emplea principalmente para permitir el trabajo en grupo entre varios alumnos y para confeccionar los contratos de aprendizaje entre el alumno y el profesor. Se trata de una variación del protocolo `FIPA-iterated-contract-net`.

5.- INTERCAMBIO DE INFORMACIÓN CON XML

XML —*eXtensive Markup Language*— es un lenguaje de marcado estructurado en el que la información se etiqueta de acuerdo a una estructura lógica que permite separar el contenido, la presentación, el comportamiento y la estructura del documento. La elección de XML como lenguaje para el intercambio de información entre los agentes se debe, entre otras razones, a que permite: (a) integración de los datos de las fuentes mas dispares, (b) gestión y manipulación de los datos desde el propio cliente web y (c) mayor autonomía de los clientes web para desarrollar tareas que actualmente se ejecutan en el servidor.

XML es el lenguaje en el que se almacena toda la información referente a un curso. Se definen varios DTD —*Document Type Definition*—en función del tipo de documento (contrato de aprendizaje, temario, artículos, materiales multimedia, listados, trabajos de grupo, etc...). Los agentes que forman el sistema deben conocer éstos DTD para poder comunicarse. Por ello, un agente de ontologías (OA) se encarga de gestionar los DTD actuales y de comunicar a los agentes interesados cualquier cambio que se produzca en ellos para que puedan comprender los mensajes.

6.- CONCLUSIONES

En este artículo se ha presentado una arquitectura basada en el paradigma de sistemas multi-agente (CALIOPE) para la gestión integral de cursos que favorezcan el aprendizaje autónomo colaborativo no presencial. CALIOPE está formada por un conjunto de agentes autónomos que guían a los distintos individuos involucrados en un proceso de enseñanza-aprendizaje, permitiendo adaptar el ritmo y las necesidades del aprendizaje a cada alumno. Su principal novedad frente a otras herramientas es que ofrece un apoyo importante para poder utilizar técnicas que fomenten el aprendizaje autónomo.

REFERENCIAS

Barros, B.; Verdejo, M.F. (1999). *"DEGREE: Un sistema para la realización y evaluación de experiencias de aprendizaje colaborativo en enseñanza a distancia"*. Actas de la VII Conferencia Española para la Inteligencia Artificial (CAEPIA '99), Vol: 1, pp.: 68 – 77. Murcia.

Botti, V.; Carrascosa, C.; Julian, V.; Soler, J. (1999). *"Modelling Agents in Hard Real-Time Environments"*. Proceedings of the MAAMAW'99. Lecture Notes In Computer Science, vol. 1647. Springer - Verlag (pag. 63-76), Valencia. ISBN 3-540-66281-2.

Boud, D. (Ed.) (1988). *"Developing Student Autonomy in Learning"*. Kogan Page (London) & Nicholas Publishing Company (New York).

Cornwall, M.G. (1979). Students as Teacher: Peer Teaching in Higher Education

FIPA (1997) *"Agent Management"*. Foundation for Physical Intelligent Agents.

Hazemi, R.; Hailes, S.; Wilbur, S. (Eds.) (1998) *"The Digital University. Reinventing the Academy"*. Springer-Verlang.

Huhns, , Michael N.; Singh, Munindar (1998). *"Readings in Agents"*. Chapter 1, pages 1-24. ISBN 1-55860-495-2

Julian, V.; Carrascosa, C.; Soler, J. (1999) *"Una Arquitectura de Sistema Multi-Agente para la Recuperación y Presentación de Información"*, IV Congreso ISKO-España, EOCONSID'99, p.291-296

Ovejero, A. (1990). *"El aprendizaje cooperativo: Una alternativa eficaz a la enseñanza tradicional"*. Promociones y Publicaciones Universitarias. Barcelona

Wooldridge, M.; Jennings N. R., editors (1995). *"Intelligent Agents - Theories, Architectures, and Languages"*. Volume 890 of Lecture Notes in Artificial Intelligence, Springer-Verlag, ISBN 3-540-58855-8.

Wooldridge, M; Jennings, N. R.; Kinny, D (1998). *"A Methodology for Agent-Oriented Analysis and Design"*. In O. Etzioni, J. P. Muller, and J. Bradshaw, editors: Agents '99: Proceedings of the Third International Conference on Autonomous Agents Seattle, WA.

REINGENIERÍA DOCENTE BASADA EN MODELOS DE CONOCIMIENTO ORIENTADOS AL CONCEPTO .

Ferran Virgós Bel[1,2], Jesús Marín[1], Joan Segura[1]

[1] *Escola Universitària d'Enginyeria Tècnica Industrial de Barcelona*
Universitat Politècnica de Catalunya, UPC
e-mail: {Ferran.Virgos|Jesus.Marin-Sanchez|Joan.Segura}@upc.es

[2] *Universitat d'Andorra (UdA)*
e-mail: fvirgos@andorra.ad

RESUMEN: La globalización de la economía y la maduración de los mercados llega tarde al mundo universitario, pero parece que ha llegado, al fin. Los mercados cautivos cada vez lo son menos y los equipos rectorales han de buscar estrategias competitivas. Apostar por la calidad es una vía, pero no es de aplicación inmediata, no es la única y, además, ha de basarse en soportes tácticos. Entonces, se gira la vista hacia las Tecnologías y Sistemas de Información y se levanta la veda de espacios virtuales y campus extendidos con planteamiento mixto. Bien está!, ya era hora!. Pero vale la pena hacerlo bien. En este artículo, se hace notar que este planteamiento corresponde simplemente a aprovechar la cadena de valor virtual (CVV), en este caso, de la actividad educativa. Para aprovechar al máximo las sinergias, la gestión de esta cadena sólo puede hacerse de una manera integrada con la cadena de valor real (CVR) y, además, así, se facilita el trabajo en equipo. Aparece la necesidad de un modelo conceptual previo de representación del conocimiento didáctico, que permita diferentes vistas e interacciones en forma dinámica. La arquitectura de cualquier sistema deberá basarse en este modelo previo.

1.- INTRODUCCIÓN.

La globalización de la economía, junto a la internacionalización y maduración de los mercados ha llevado a los diferentes sectores industriales a un gran aumento de la competencia que ha forzado a los equipos directivos a buscar estrategias competitivas, ya sean de liderazgo en coste, diferenciación de producto (entre ellos, la calidad) o la especialización [Porter, 1985].

La aplicación de las tecnologías de la información y comunicaciones (TIC), ha sido uno de los recursos básicos utilizados para mejorar la posición competitiva de la empresa. En este sentido, el concepto de

cadena de valor (CV) de Porter [1989] es relevante. El modelo permite describir y analizar las actividades que añaden valor en una organización. El objetivo es encontrar un marco conceptual que permita detectar las "oportunidades" de las TIC.

Históricamente la Universidad no se ha sentido tan "presionada" por el mercado, en el sentido de disponer de unos clientes locales "cautivos", junto a una cierta indiferencia ante el tema de la eficiencia o la eficacia. Pero más recientemente la globalización de la economía está trayendo la competitividad, aparecen las Universidades privadas y también a la Universidad pública se le exigen resultados a través de contratos-programa.

2.- LA CAPACIDAD INTEGRADORA DE LAS TIC.

Siempre me gusta explicar una anécdota que me sucedió en una ocasión, cuando asistiendo a un seminario, el monitor había repartido un libro de anillas (que sin duda le había costado trabajo preparar) con material didáctico relativo al tema del curso, pero utilizaba en su presentación unas transparencias "distintas" (que sin duda, también le había costado trabajo preparar).

En un descanso aproveché para acercarme a él a fin de preguntarle si era posible disponer de una copia de las transparencias. Me miró con cara entre extrañado y agredido, y me respondió simplemente " tiene lo mismo en el libro". Naturalmente no contesté y me fui cabizbajo pensando que algo estábamos haciendo mal los profesores: él había hecho el trabajo dos veces y a mí me había dado algo que apenas me servía un poco más que cualquier libro que podría comprar en una librería. Aquel día comprendí que nos faltaba disponer de un modelo conceptual único de conocimiento a efectos didácticos que permitiera diferentes entradas en la "creación" (*inputs* del profesor) y diferentes "vistas al acceder" (diferentes *outputs* en la visión alumno) a ese conocimiento. Más adelante volveremos con el tema pero es fácil adelantar que las TIC deberán tener un notable proitagonismo en la solución integradora ([Virgós et al,. 1997], [Virgós et al., 2000]).

3.- LA CADENA DE VALOR VIRTUAL Y SUS POSIBILIDADES EN LA UNIVERSIDAD.

La cadena de valor (CV) de Porter es una cadena de valor del mundo real (CVR) y sirve de modelo para analizar las posibilidades de las TIC mediante el diseño de un sistema de información pero no considera que éste tenga valor en si mismo. Es sólo un sistema de soporte. Aplicando el concepto a la Universidad, podríamos decir que cuando un profesor realiza unos apuntes o unas transparencias para clase con un ordenador, está usando las TIC como ayuda a su cadena de valor real (su docencia!).

Rayport & Sviokla [1995] nos recuerdan que el Sistema de información de la CVR constituye un sistema paralelo (la cadena de valor virtual, CVV) que tiene (o puede tener) valor en si mismo. El concepto de cadena de valor virtual (CVV) aparece cuando nos apercibimos de que la información de que disponemos en el sistema de Información tiene valor intrínseco.

Las actividades clásicas de los Sistemas de Información : recoger, tratar (organizar, seleccionar y procesar), almacenar y distribuir información, constituyen las bases del concepto de cadena de valor virtual.

Para crear valor en esta cadena, debemos detectar la disponibilidad de elementos que, independientemente o en su conjunto, puedan tener valor para los clientes en el mercado electrónico.

En el artículo citado, Rayport & Sviokla proponen una evolución en tres fases para incorporar los procesos de información que añaden valor: visibilidad, capacidad de replicación o dualidad (*replicate*, es el término usado en el original) y establecer nuevas "relaciones" con los clientes.

Si aceptamos que un elemento esencial de la cadena de valor virtual de la universidad es el propio conocimiento que ésta transmite, deberemos concluir que el concepto de campus extendido o la formación virtual no son más que nuevas relaciones con el cliente de esta CVV.
Con ello, tendremos centrado este trabajo.

4.- NECESIDAD DE UN MODELO CONCEPTUAL.

a) Una revisión de los paradigmas habituales en entornos virtuales.

La aparición de campus extendido y modelo mixto presencial / no presencial es un intento de una auténtica "reingeniería"([Chelappa et al., 1997], [Tsichritzius, 1999]). Sin embargo, la capacidad integradora de la tecnología es puramente formal. Esto es, se utiliza la tecnología en proyectos específicos y, en el mejor de los casos, sirve para integrar material didáctico heterogéneo. Sin embargo, no es habitual que exista un modelo conceptual previo de la estructura del conocimiento (didáctico).

A nuestro entender dicha situación recuerda lo ocurrido en el campo de la ingeniería del software o las bases de datos, en donde una formalización previa del problema evita la dependencia física, la duplicación de tareas, la redundancia de información (que puede generar inconsistencias) o la detección de errores conceptuales en una fase de desarrollo demasiado tardía como para poder rectificar. En el fondo, libros, transparencias y apuntes de los profesores son de algún modo como los ficheros de antaño, con los problemas de siempre: redundancias (inconsistencias), o una dependencia física demasiado acentuada que impiden la obtención de vistas específicas según la necesidad docente.

b) Requerimientos iniciales.

Una vez aparecida la necesidad de diseñar una arquitectura tecnológica basada en un modelo conceptual diseñado previamente, estableceremos unos objetivos principales:

- Simplicidad: Un modelo simple permite una implantación / implementación menos costosa, esencialmente mediante el uso de herramientas ya existentes. No pretendemos introducirnos en el mundo de la I.A.

- Capacidad integradora desde el punto de vista tecnológico.

- Capacidad incremental (ciclo de vida espiral) desde el punto de vista de recursos humanos (trabajo en equipo dinámico en el tiempo).

- "Pluri-morfismo": Partiendo de un modelo orientado al concepto deberíamos ser capaces de obtener cierta independencia respecto al hecho de encontrarnos ante una enseñanza presencial o no, de desear orientar la docencia hacia el autoaprendizaje, o usar o no algún soporte tecnológico (un libro, también es material didáctico).

- Reutilización.

5.- IDENTIFICACIÓN DE COMPONENTES.

En una segunda fase, identificamos aquellos elementos indispensables que formarán parte del conocimiento que pretendemos modelizar (y que nos hemos permitido la licencia de rebautizar para evitar connotaciones):

- *Sesión básica de estudio.* Es la unidad mínima temporal de estudio que correspondería normalmente al tiempo de una clase convencional (50 minutos) o el tiempo requerido por el estudiante para el (auto)aprendizaje de un concepto atómico.

- *Concepto atómico.* Corresponde a un concepto didáctico obtenido en un último nivel de descomposición de tal forma que sea viable exponerlo en una sesión básica, y que sea posible "verificar" su asimilación según un eje de actividad o ejercicio (que hemos llamado *ejercicio conceptual*). Un concepto atómico estará compuesto por una o varias *unidades visuales.*

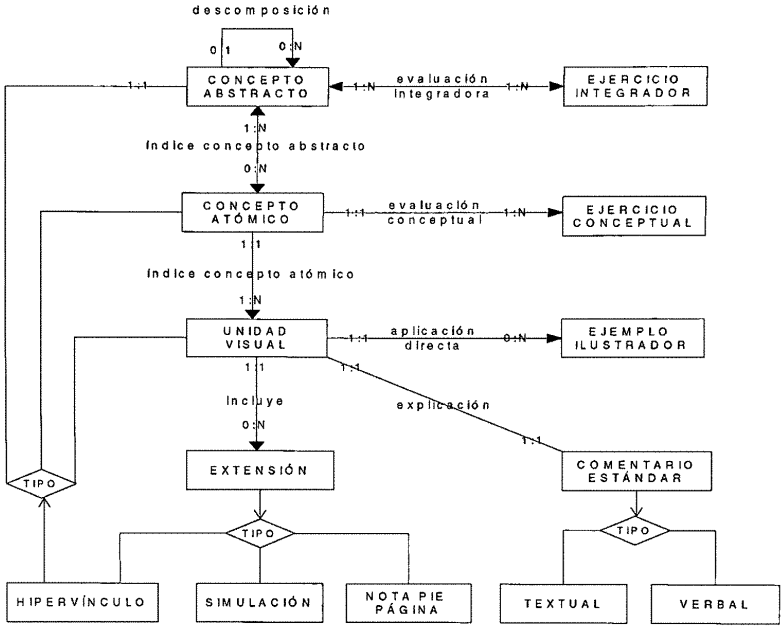
- *Unidad visual.* Parte del concepto clásico de pizarra, o transparencia. Puede estar dotada de un *comentario estándar*, de *extensiones* o aclaraciones, y de *ejemplos ilustradores.*

- *Extensión.* Abarca elementos como aclaraciones a alumnos, notas pie de página de un libro, hipervínculos de repaso, de relación o de definición,...

- *Concepto abstracto.* Es un concepto que no puede ser expuesto en una sesión básica de estudio. Estaría compuesta por conceptos atómicos y/o por conceptos abstractos de nivel inferior según una definición recursiva. Los *ejercicios integradores* facilitarían la verificación de la asimilación.

6.- EL MODELO CONCEPTUAL.

En una tercera fase, elaboramos un modelo conceptual usando un formalismo de modelización propio de ingeniería del software, que además nos proporciona como ventaja adicional, la facilidad para ser informáticamente implementado.



7.- CONCLUSIONES.

Orientando la arquitectura tecnológica en función de este modelo, creemos que pueden garantizarse los requerimientos ya expuestos en el epígrafe 4b, singularmente:

- Capacidad integradora.
- Capacidad incremental.
- "Pluri-morfismo" a través de diferentes "vistas".
- Fácil reutilización en diferentes cursos.

Pero, además, creemos que facilitaría:

- La construcción de material didáctico para formación bajo demanda.
- Por extensión, también facilitaría la integración de material procedente de diferentes fuentes, ya sean universidades, profesores individuales, etc, siendo la base de incorporación de recursos docentes genéricos.

Naturalmente debería dotarse al sistema de un nivel superior que facilitara un *front-end* con un motor de "modulo" para adaptarlo al nivel y otro de "alumno" para considerar su individualidad del alumno, personalizando su visión e integrando la evaluación.

8.- BIBLIOGRAFÍA.

[Candiotti & Clarke, 1998]
A. Candiotti, N. Clarke. Combining universal access with faculty development and academic facilities. Communications of the ACM. January 1998. Vol 41. N. 1.

[Chelappa et al., 1997]
R. Chellappa, A. Barua , A. B. Whinston. An electronic infrastructure for a virtual university. Communications of the ACM. Sept 1997. Vol 40. No. 9.

[Igbaria, 1999]
M. Igbaria. The driving forces in the virtual society. Communications of the ACM. Dec. 1999. Vol 42. No. 12.

[Llamas et al., 1997]
M. Llamas, M. J. Fernández, A. Gil, R. Rodríguez, A. Suarez. Cómo convertir el web en un entorno educativo. Novática, may/jun 1997 No. 127.

[Porter, 1985]
M.E. Porter. Competitive advantage. The Free Press. 1985

[Rayport & Sviokla, 1995]
J. F. Rayport, J. J. Sviokla. Exploiting the virtual value chain. Harvard Deusto Business Review. Nov-dec 1995.

[Rockart, 1975]
J. F. Rockart. Computers & the Learning process. Sloan School. CISR WP 15. 1975.

[Roxanne & Wellman, 1997]
S. Roxanne Hiltz, B. Wellman. Asynchronous learning networks as a virtual classroom. Communications of the ACM. Sept 1997. Vol 40. No. 9.

[Taylor, 1980]
R.Taylor. The computers in the school: Tutor, tool, tutee. Teaches College Press. 1980

[Telleen, 1998]
S. Telleen. Intranets: the new knowledge base. Oracle magazine. September-october, 1998.

[Tsichritzis, 1999]
D. Tsichritzis. Reengineering the University. Communications of the ACM, june 1999/Vol 42. No. 6.

[Vaquero, 1998]
A. Vaquero. Las TIC para la enseñanza, la formación y el aprendizaje.
Novática, mar./abr. 1998 no. 132.

[Virgós, 1989]
F. Virgós. Los ordenadores en el entorno educativo. II premios EPSON de divulgación informática, en el libro "Informática y Escuela". Marcombo. Barcelona, 1989

[Virgós et al., 1997]
F. Virgós, J. Segura, F. Juan. Las TI en las Escuelas de Ingeniería Técnica : ¿qué hacer para afrontar el futuro?. V Congreso Universitario de Innovación metodológica en las enseñanzas técnicas. Actas del Congreso. Barcelona, 1997.

[Virgós et al., 2000]
F. Virgós, M. García, A. Pérez, F. Pla, J. Segura. Reengineering the University through managing knowledge on virtual value chain. CIDUI 2000. Barcelona, junio 2000.

[Wulf 97]
W. A.- Wulf. Look in the spaces for tomorrow's innovations. Communications of the ACM. February 1997. Vol 40, num 2.

ÁREA DE MATEMÁTICAS

Responsable: Dr. Josep Miró
(Universitat de les Illes Balears)
E-mail: joe@ipc4.uib.es

ANOTACIONES SOBRE LA IMPORTANCIA DE LA ESTADÍSTICA EN LAS INGENIERIAS INFORMÁTICAS

Angel Igelmo Ganzo
Universitat de les Illes Balears
e-mail:angel.igelmo@uib.es

RESUMEN: Esta Ponencia justifica la docencia de la asignatura de *Estadística* en los *Estudios de Ingeniería Informática*. La asignatura, *Estadística*, es troncal a nivel de Ingeniería Técnica pero desigualmente distribuidos, pues en Informática de Sistemas la carga docente es de 4,5 créditos mientras que en Informática de Gestión es de 9 créditos. Aunque es una asignatura consolidada según los vigentes planes de estudios es necesario insistir en la importancia de la misma justificando la docencia en las Ingenierias Informáticas recurriendo al propio valor de la disciplina como herramienta tecnológica. Por tanto en los apartados siguientes hacemos una referencia a la metodología estadística y a la metodología científica y técnica, haciendo ver que ***la Estadística es el Método Científico de las Ciencias de la Naturaleza y por tanto de la Ingeniería***

1.- EL MÉTODO ESTADÍSTICO VERSUS EL MÉTODO CIENTÍFICO

1.1.- El Método estadístico

Sin más preámbulo asumimos lo escrito por Box, Hunter and Hunter (1) en la primera página del primer capítulo de su libro Estadística para Investigadores, cuando nos dicen que:

> *La investigación científica es un proceso de aprendizaje dirigido. ..... El aprendizaje avanza con la interacción bidireccional entre hipótesis (conjeturas, modelo, teoría) y datos (hechos, fenómenos). Una hipótesis inicial conduce, por un proceso de deducción, a ciertas consecuencias que pueden ser contrastadas con datos. Cuando las consecuencias y los datos no concuerdan la discrepancia puede conducir, por un proceso inductivo a la modificación de la hipótesis. Se inicia entonces un segundo ciclo de iteración. Se deducen las consecuencias de la hipótesis modificada y se comparan de nuevo con los datos que a su vez pueden llevar a ciertas modificaciones y ganancia de conocimiento. Esta proceso de aprendizaje puede también describirse como un bucle de realimentación en el que la discrepancia entre los datos y las consecuencias de la hipótesis H_1 conduce a la hipótesis modificada H_2, H_2 conduce a H_3 y así sucesivamente.*

En base a lo anteriormente expuesto podemos definir las siguiente etapas en la aplicación de la metodología estadística, de acuerdo con Fernández-Abascal (2) y otros (10), (11):

i.- Planteamiento del problema. Se trata de definir las variables a observar, las cuales concretarán el problema, su ámbito, su alcance, etc. En definitiva es imprescindible saber lo que se va a estudiar. Un planteamiento inadecuado del problema nos puede llevar a hermosos errores.

ii.- Obtención de datos. La toma de datos es una fase importante del proceso pues una mala toma de datos puede invalidar todo el proceso, llegando a conclusiones erróneas. En las ciencias experimentales e ingenierías será una labor a desarrollar en laboratorio o in situ. Es evidente que en esta fase juega un papel clave el concepto de muestreo.

iii.- Análisis descriptivo de datos. Se trata de recoger, ordenar o clasificar y analizar los datos, función propia de la Estadística Descriptiva y del más moderno Análisis Exploratorio de Datos, que denominamos Análisis Descriptivo de Datos.

iv.- Modelo teórico. A la vista de las conclusiones obtenidas en el apartado anterior, se pasa a proponer un modelo teórico, recurriendo a técnicas de contraste de hipótesis, estimación de parámetros, pruebas no paramétricas, etc. Esta etapa es la que conocemos con el nombre de Estadística Inferencial (proceso inductivo).

v.- Explotación de modelo. Si no encontramos un modelo que se ajuste a los datos obtenidos, será conveniente volver a la fase *i* o *ii*, revisando tanto el planteamiento del problema como la obtención de datos. Si disponemos finalmente de un modelo que se ajuste a nuestra observación empírica procederemos a sacar conclusiones de todo orden a partir de nuestra modelización. Esta fase corresponde a un proceso matemático deductivo

Pero esta secuencia de etapas coincide sustancialmente con el método científico como describiremos más adelante. Por esta motivo algunos han definido que la Estadística es la ***"Metodología de la Ciencia Experimental"***.

Abundando en esta forma de ver la Estadística, su metodología, también es interesante recurrir a L. Sachs (8) cuando nos dice:

> *Para nosotros es importante lo siguiente: basándonos en las condiciones específicas del problema hacemos suposiciones acerca del modelo subyacente y acerca del correspondiente modelo estadístico. Una vez comprobada la compatibilidad de las observaciones con el modelo estadístico se pasa a la descripción estadística de la población: ..... En ambos casos se llega a conclusiones de tipo probabilístico, es decir, en términos de probabilidad. Así pues, la misión de la estadística consiste en encontrar los modelos estadísticos adecuados a cada problema y en aprovecharlos para extraer toda la información esencial que puedan encerrar los datos; es decir, la estadística proporciona modelos para la reducción de la información.*

En todo este procesos o esquema hay, en mi opinión, una palabra clave: **DATOS**, de manera que parodiando a Arquímedes, podemos decir: ***"Dadme datos y moveré al mundo".***

Mencionemos lo que dice D. Peña (4) sobre la metodología Estadística:

> *La Estadística actúa como disciplina puente entre los modelos matemáticos y los fenómenos reales. Un modelo matemático es una abstracción simplificada de una realidad más compleja y siempre existirá cierta discrepancia entre lo observado y*

lo previsto por el modelo. La Estadística proporciona una metodología para evaluar y juzgar estas discrepancias entre la realidad y la teoría.

Esta claro por tanto que el estudio de la Estadística es fundamental para todos aquellos que deseen trabajar en ciencia experimental, pura o aplicada (ingeniería), que requiera del análisis de datos y del diseño de experimentos, habiéndose dicho, incluso, que **la Estadística es la "tecnología" del método experimental**.

Estamos de acuerdo en que la Estadística es como una tecnología, aunque en todo momento usa un importante aparato matemático, por lo que algunos están tentados en presentar la estadística como una cosa matemática. Pero eso se denomina Estadística Matemática y no debe ser nuestro objetivo. Precisamente para que el alumno capte la capacidad tecnológica de la Estadística, debemos seguir un método didáctico en nuestras explicaciones diarias con los alumnos, que procure seguir la idea manifestada por Sixto Rios (7) en su Prologo de Métodos Estadísticos:

Creemos, por otra parte, que el papel de la Estadística Matemática, en relación con otros cursos de Matemáticas, puede ser importante para modificar la tendencia a cultivar en la enseñanza de la Matemática solamente las facetas algorítmicas y lógico-deductivas. La manera tradicional de enseñar las Matemáticas como un edificio acabado, partiendo de unos axiomas y demostrando una serie de teoremas para terminar cada lección con algunos ejercicios que van de lo ingenioso a lo trivial, pero que están tan lejos de la realidad y de las aplicaciones como la teoría escuetamente expuesta, es totalmente inadecuado y deformante para el que pretende trabajar como matemático aplicado, físico o ingeniero. El tipo tradicional de ejercicios de fin de capítulo, que suelen comenzar: "probar que...", "resolver la ecuación...", está muy distante de las preguntas que el matemático se encontrará en la realidad que serán del tipo "¿que sucede si...?", "¿Cuál es la relación que existe entre...?", "¿Cómo podemos encontrar...? Si queremos preparar a los alumnos para ser capaces de matematizar los problemas del mundo real, en que es frecuente que el objetivo central aparezca oscurecido por una masa de información incompleta y en que precisamente lo principal es destacar dicho objetivo, debemos cambiar radicalmente la manera tradicional de enseñar la Matemática. Del triple proceso de conceptualización, razonamiento lógico-deductivo y desconceptualización, que se presenta en toda aplicación de la Matemática al estudio de los fenómenos reales, se olvidan frecuentemente en la enseñanza la primera y última fases, que son justamente las más importantes,...

Por este motivo debemos hacer especial hincapié en los aspectos primero y tercero, siendo el aspecto matemático un asunto instrumental, pero sin que ello implique pérdida de rigor.

1.2.- La Estadística como metodología científica

El objeto de las *ciencias experimentales puras o aplicadas* no son características o sucesos aislados y únicos que afectan a un determinado individuo o elemento, sino experiencias repetibles. Los métodos estadísticos son necesarios en aquellas situaciones en que unos resultados no se pueden reproducir ilimitadamente y con toda exactitud, quedándonos en una situación de incertidumbre.

Otra forma de definir por tanto la Estadística es con relación a la toma de decisiones en un contexto de incertidumbre: "*La Estadística es un compendio de métodos que nos permite tomar decisiones óptimas en casos de incertidumbre*". De aquí que la metodología estadística sea una herramienta imprescindible en la metodología científica y técnica.

Sin embargo es más habitual definir la estadística a partir de lo que hace, es decir como la *ciencia que recoge, ordena y analiza los datos de una muestra extraída de una población, para inducir conclusiones sobre el comportamiento de la población, valiéndose del cálculo de probabilidades*. Es decir, de las observaciones hechas sobre una parte de la población (muestra) se sacan conclusiones sobre la totalidad de la población; esto es, se procede inductivamente. Es fundamental que la muestra haya sido extraída al azar. Las muestras extraídas al azar, muestras aleatorias, son importantes, puesto que sólo de ellas se pueden extraer conclusiones acerca de la totalidad de la población. Hay diversos motivos por lo que no es posible analizar la totalidad de la población: por ejemplo, el análisis puede ser de tipo destructivo, o ser muy costoso o llevar mucho tiempo.

La definición dada de estadística permite distinguir dos etapas. La parte que recoge, ordena o clasifica y analiza los datos de la muestra, es lo que llamamos *Estadística Descriptiva*. Mientras que la parte que induce conclusiones sobre la población a partir del análisis de la muestra es lo que llamamos *Estadística Inferencial* (o Inferencia Estadística). Esta permite obtener regularidades generales cuya validez rebasa el conjunto de observaciones de la muestra.

En todas las ciencias empíricas la estadística permite la interpretación de datos empíricos y la verificación de teorías científicas o técnicas al confrontar resultados empíricos con resultados deducidos de modelos probabilísticos teóricos , considerando que todas las conclusiones posibles son de tipo probabilístico.

Abundando en estas ideas P.C. Mahalanobis dice que ***"las leyes científicas no son formuladas por el principio de autoridad o justificadas por la fe o la filosofía medieval; la estadística es el único tribunal de apelación del conocimiento nuevo".***

La ciencia trata con el conocimiento de los fenómenos naturales y su perfeccionamiento. Tal conocimiento se expresa en formas de leyes, axiomas o teorías, las cuales permiten la predicción de futuros acontecimientos. Buscamos hipótesis de trabajo, que estén apoyados por los hechos observados, los cuales en el transcurso del tiempo pueden ser reemplazados por mejores hipótesis, con más pruebas que las evidencien a partir de un grupo de datos más amplio. Según Popper (5) el método de investigación científico implica el siguiente ciclo sin fin:

$$P_1 \rightarrow CT \rightarrow EE \rightarrow P_2$$

donde P_1 y P_2 son la teoría inicial y su modificada respectivamente, CT representa el proceso de comprobación de la teoría y EE la eliminación de errores. La teoría inicial P_1 nace del fallo del conocimiento humano recibido anteriormente.

Este esquema permite aludir a la actitud de Popper ante la ciencia que lejos de considerarla como una ocupación rutinaria y prosaica, coloca a los científicos a la altura de los poetas y los músicos, por cuanto a imaginación creadora se refiere.

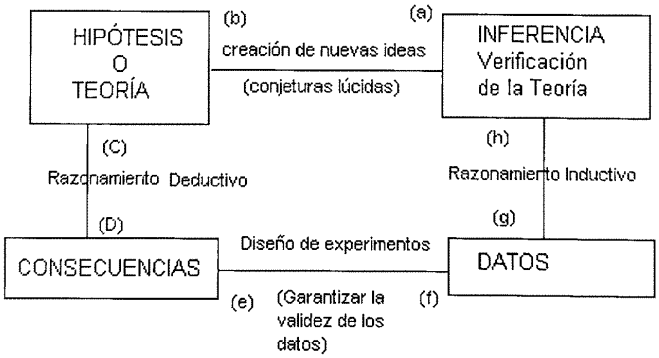
Pero incluso la metodología estadística transciende a la ciencia, llegando a la filosofía. Así, Vargas Llosa (9) refiriéndose a Popper manifiesta:

> *Creo que no hay filosofía contemporánea o ningún pensamiento contemporáneo que haya hecho de la libertad un hecho tan absolutamente central como es el caso*

de la filosofía de Popper. La idea de la verdad en Popper a mi me recuerda mucho un mito con el que se abre uno de los grandes libros de la antropología moderna, La Rama Dorada de Sir James Frazer. Recuerdo cómo se inaugura ese libro, describiendo en no se que cultura del Pacífico a ese reyezuelo que daba vueltas muy alarmado entorno al árbol, símbolo del poder en su cultura, esperando aquel que vendría a sucederlo, aquel que, como había hecho él con su antecesor, un día aparecería, lo mataría y lo sucedería y heredaría su poder. Leyendo a Popper, la idea de la verdad en Popper, yo he pensado siempre en ese reyezuelo. Porque, según Popper, la verdad tiene, como en el caso de este rey primitivo, ese carácter provisional: es una verdad que reina mientras no viene otra a liquidarla, a acabar con ella y a reemplazarla.

Esa verdad, que es una creación humana más que un descubrimiento, que es una hipótesis o teoría que pretende responder una pregunta y que entra luego a vivir una vida precaria, constantemente sometida a esa prueba del ensayo y del error, y que un día es incapaz de resistir esa presión y sucumbe a la prueba y es derrotada y reemplazada por otra, es una verdad que hace de la crítica, es decir, el derecho a someter a todas las verdades establecidas al fuego graneado, constante, continuo de la refutación, la condición básica del conocimiento. Ninguna verdad puede ser reemplazada sin la prueba del ensayo y del error, y no puede haber prueba del ensayo y del error sin la crítica, es decir, sin la libertad, sin el derecho a refutar todas las verdades establecidas sin excepción.

El método científico, siguiendo a Rao (6), se puede representar gráficamente como sigue:



donde se pone de manifiesto los dos procesos lógicos: razonamiento deductivo y razonamiento inductivo. En este diagrama se ponen de manifiesto dos etapas: por un lado los pasos (a) → (b) y (c) → (d), donde están involucrados cada área de investigación, y por otro lado (e) → (f) y (g) → (h) que están sujetos a la metodología estadística.

A través de la recolección de datos relevantes y válidos, la estadística permite al científico poner en juego todas sus potencialidades creativas y así descubrir nuevos fenómenos. Los métodos estadísticos han sido de gran valor especialmente en aquellas ciencias y técnicas donde el rango de variación de las observaciones es a menudo grande y el número de observaciones suele ser limitado. Resumiendo: "***Si no existiese la Estadística, habría que inventarla***".

2.- LA INFLUENCIA ACTUAL DE LA ESTADÍSTICA

Desde principios de siglo un sinfín de nuevos métodos y teorías y un inmenso campo de aplicación dificultan una visión general de lo acontecido y hace arriesgado resaltar alguna innovación decisiva. Sin embargo repasemos algunos acontecimientos.

Hasta la segunda guerra mundial se extiende la aplicación de la estadística en áreas tan diversas como la Ingeniería (Control de Calidad por Shewart, métodos de predicción y control de procesos y codificación de señales por Wiener y Shanon), la Física (teoría cinética de los gases), la Antropología, la Psicología o la Medicina. La búsqueda de respuesta a los nuevos interrogantes planteados por estas aplicaciones impulsan, a su vez, el desarrollo de nuevos métodos estadísticos. Los problemas agronómicos llevan a Fisher a crear la teoría de diseños experimentales y un problema de discriminación en antropología (estudio de cráneos) le lleva a inventar el análisis discriminante. El análisis factorial surge ligado a problemas en Psicología mientras que las ciencias sociales impulsan el desarrollo de métodos para analizar muchas variables simultáneamente (análisis multivariante). Los problemas en Ingeniería conducen a un estudio sistemático de la teoría de modelos dinámicos (procesos estocásticos), y a la teoría de predicción y de extracción de señales de Wiener y Komogorov en la década de los cuarenta. Las necesidades de control de procesos sugiere a Pearson la creación de la teoría general de contraste de hipótesis, junto con Neyman. El trabajo del Statistical Research Group en Princeton en aplicaciones industriales de la estadística durante la segunda guerra mundial condujo a Wald a inventar los contrastes secuenciales para el control de recepción y fue básico en el nacimiento y desarrollo de la teoría estadística de decisión. Durante este periodo las aplicaciones de la estadística a la economía conducen a una disciplina con contenido propio: la Econometría.

A partir de la segunda guerra mundial se puede decir que empieza la época moderna de la Estadística, apareciendo aspectos diferenciadores respecto de la época anterior, de los que cabe resaltar (ver (10), (11)):

- El espectacular desarrollo de las matemáticas, que proporciona nuevos recursos formales a la Estadística.

- Los nuevos campos de aplicación, principalmente la Biología, Economía, Industria, etc. que abren nuevas perspectivas a la Estadística.

- La generalización masiva del uso de los ordenadores y de la informática que introduce la Estadística en problemas prácticos hasta ahora inabordables por imposibilidad de cálculo, a la vez que impulsa teorías apropiadas para el análisis de grandes colecciones de datos.

Esta breve revisión de la evolución de la Estadística muestra como los grandes periodos de avances teóricos se han producido generalmente ligados a la resolución de importantes problemas prácticos. Paradójicamente, la mayoría de las contribuciones importantes que hemos revisado son debidas a investigadores que no pueden calificarse exclusivamente como estadísticos, sino en el sentido mas amplio del término como científicos.

3.- BIBLIOGRAFÍA

1. Box, G.E.P., Hunter, W.G., Hunter, J.S. (1989). *Estadística para investigadores. Introducción al diseño de experimentos, análisis de datos y construcción de modelos*. Editorial Reverter.
2. Fernández-Abascal, H., Guijarro, M.M., Rojo, J.L., Sanz, J.A. (1994) *Cálculo de Probabilidades y Estadística*. Editorial Ariel.
3. Igelmo, A. (1998) *Análisis Descriptivo de Datos*. Servicio de Publicaciones de la Universidad de las Islas Baleares.
4. Peña Sánchez de Rivera, D. (1994). *Estadística, Modelos y Métodos. Tomo I: Fundamentos*. Alianza Universidad. 2ª Edición.
5. Popper, K.R. (1973). *La lógica de la Investigación Científica*. Editorial Tecnos. 3ª reimpresión.
6. Rao, C.R. (1994). *Estadística y verdad*. PPU.
7. Rios, S. (1973). *Métodos Estadísticos*. Ediciones del Castillo.
8. Sachs, L. (1978). *Estadística Aplicada*. Editorial Labor.
9. Schwartz, P., Rodriguez Braun, C. y Méndez Ibisate, F. (1993). *Encuentro con Karl Popper*. Alianza Editorial
10. A. Meca, M. Almiñana, F. Borrás, J. Cánovas, N. Llorca, E. Molina, D. Morales, A. Sánchez, J.V. Segura. (1998). *Sobre el futuro de la Estadística y la Investigación Operativa: Primera Parte*. Boletín de la SEIO. Volumen 14, nº 3, septiembre de 1998, pag. 2-9.
11. A. Meca, M. Almiñana, F. Borrás, J. Cánovas, N. Llorca, E. Molina, D. Morales, A. Sánchez, J.V. Segura. (1998). *Sobre el futuro de la Estadística y la Investigación Operativa: Segunda Parte*. Boletín de la SEIO. Volumen 14, nº 3, septiembre de 1998, pag. 2-9.

GEOMETRÍA FRACTAL

Carmen Escribano[1], M. Asunción Sastre[1].

[1]*Depto. Matemática Aplicada, Facultad de Informática, U.P.M.*
e-mail: **cescribano@fi.upm.es**
e-mail: **masastre@fi.upm.es**

RESUMEN: En este artículo se presenta la Geometría Fractal como uno de los contenidos matemáticos de interés para los estudiantes de informática. Se describen el contenido y el desarrollo de Geometría Fractal, asignatura optativa de cuarto curso (cuatrimestral de 6 créditos), impartida este año en la Facultad de Informática de la U.P.M. En el último apartado haremos una valoración de esta experiencia y del interés manifestado por los alumnos.

1.- INTRODUCCIÓN.

La Geometría Fractal es un lenguaje utilizado para analizar y describir tanto fenómenos como formas complejas presentes en la naturaleza. Mientras que los elementos del lenguaje tradicional de la Geometría Euclidea son formas básicas visibles, tales como rectas, esferas, círculos,..., los del nuevo lenguaje utilizan algoritmos, que no son observables de forma directa. Estos algoritmos generan formas y estructuras con la ayuda del ordenador y nos proveen de una potente herramienta descriptiva, mediante la cual podemos describir con una reducida cantidad de información la forma de una nube de forma tan fácil y precisa como un arquitecto describe una casa a través del lenguaje de la geometría tradicional.

En 1977 B.B. Mandelbrot publicó el libro "The Fractal Geometry of Nature" describiendo numerosas aplicaciones de este tipo de estructuras para la investigación en ciencias aplicadas, siendo una herramienta fundamental en la modelización de un gran número de fenómenos naturales. El término fractal, procedente del latín "fracmus" (fragmentado, irregular), fue introducido por Mandelbrot para designar estos conjuntos que no tenían ningún nombre concreto y desde entonces se conoce esta rama de las matemáticas como geometría fractal o teoría geométrica de la medida.

El objetivo del curso es la introducción a la Geometría Fractal y sus aplicaciones a la informática, entre las que destacan la compresión fractal de imágenes, simulación y generación de imágenes.

Comenzaremos estudiando los conjuntos fractales clásicos con diferentes algoritmos para generarlos y los conceptos matemáticos básicos para diferenciar su "tamaño". A continuación daremos las herramientas matemáticas necesarias utilizadas en la compresión fractal de imágenes. Acabaremos el curso con diferentes algoritmos de codificación y decodificación fractal de imágenes, comparando con otros métodos más clásicos de compresión de imágenes.

2.- CONTENIDOS DEL CURSO.

En este apartado expondremos y comentaremos el contenido de la asignatura.

a) Fractales Clásicos.

La aparición de conjuntos muy irregulares como el conjunto de Cantor (alrededor de 1890) y de otros que le siguieron (triángulo de Sierpinski, curva de Koch, curva de Peano,....) causaron tal confusión entre los matemáticos de la época que llegaron a denominarse conjuntos patológicos. El origen de la geometría fractal está en el estudio de estos conjuntos, que llamaremos fractales clásicos, y sus propiedades. Estos conjuntos serán el objeto de estudio de este primer tema. También veremos diferentes algoritmos para generar los fractales clásicos y algunas modificaciones de estos.

Para acabar el tema introduciremos, de forma intuitiva y practica, los conceptos de medida y dimensión de Hausdorff y dimensión Box. Calcularemos estos valores para los conjuntos ya estudiados.

b) Sistemas de Funciones Iteradas.

La mayor parte de los conceptos matemáticos en que se fundamenta la compresión fractal son básicos y ya han sido vistos por los alumnos en los primeros cursos. En este tema recordamos estos conceptos y explicamos de forma muy intuitiva los que el alumno no conoce, como por ejemplo la distancia de Hausdorff. Se aplican las ideas generales que ya conoce el alumno al espacio de los conjuntos compactos, en particular el teorema del punto fijo a los sistemas de funciones iteradas, y se definen los conjuntos autosemejantes. Una variación interesante de los S.F.I es la denominada S.F.I. recurrentes, en la que se pueden poner a trabajar en paralelo diferentes S.F.I. para construir un nuevo fractal mezcla de los originales.

Con estas sencillas herramientas, gracias a la propiedad de continuidad de los S.F.I. respecto a los parámetros, se pueden construir variaciones, deformaciones y movimientos de los conjuntos fractales estudiados en el primer tema.

Acabaremos el tema con el teorema del Collge, que es la idea en que descansa la compresión fractal.

c) Compresión fractal de imágenes.

Para empezar el tema veremos cómo guardar la información de las semejanzas que utilizamos para construir el triángulo de Sierpinski y compararemos lo que ocupan estos datos y lo que ocupa un fichero BMP del mismo triángulo. Veremos las ideas en que se basan otros métodos de compresión y los algoritmos más sencillos de compresión fractal, tanto para imágenes binarias como monocromas. Para comprimir imágenes en color, se utiliza la compresión por bandas, normalmente RGB o YUV, comprimiendo cada banda como una imagen monocroma, volviendo a juntarlas en el proceso de descompresión.

La principal ventaja de la compresión fractal es su radio de compresión, mientras que su principal inconveniente es la lentitud a la hora de comprimir, ya que la descompresión es bastante rápida. Esto hace que sea muy útil para guardar mucha información en poco espacio, así como para transmitirla por la red. Un ejemplo de esto es la enciclopedia Encarta, cuyas

imágenes están comprimidas con este tipo de compresión. Acabaremos el curso estudiando mejoras de los algoritmos estudiados que aumenten la velocidad de compresión, sin perder calidad.

3.- DESARROLLO Y DIDÁCTICA DEL CURSO.

La asignatura es cuatrimestral con cuatro horas a la semana. Tres horas en el aula y una de realización de prácticas. Por ser el primer año que se impartía esta asignatura y debido a que algunos alumnos que la escogieron tenían conocimiento de cómo se aplicaba la Geometría fractal en diversas materias en las que tenían interés, se les permitió realizar una practica libre, en la que podían profundizar en esos temas y exponerlo en clase.

Durante los dos primeros meses aproximadamente, mientras los alumnos iban realizando las prácticas y tenían materia suficiente para trabajar, se utilizaban las tres horas semanales en créditos teóricos. Estas tres horas se impartían en dos días, uno una hora y otro dos. La última hora se aprovechaba para mostrar con el ordenador proyectos fin de carrera realizados en la Facultad, relacionados con los temas que se explicaban en el momento. También se mostraban programas de libre distribución que sirvieran para aclarar ideas y motivar a los alumnos en sus prácticas, tales como fractin, el juego de la vida, chaos, fractus, etc.

En la última parte del curso, se reservaba la última hora del día de dos para que los alumnos expusieran sus prácticas. Estas solían presentarse con ordenador, ya que en la mayor parte de los casos se implementaban algoritmos, aunque también se admitía la exposición de un tema. Hay que mencionar que todas las aulas de la Facultad están equipadas con cañon, lo que permite explicar con ayuda del ordenador.

4.- CONCLUSIÓN, OBJETIVOS Y VALORACIÓN FINAL DEL CURSO.

El interés y el trabajo realizado por los alumnos, en la mayoría de los casos, fue extraordinario. Gracias a las prácticas realizadas, se vieron temas que en un principio no se incluían en la asignatura. Cabe destacar entre estos temas, el de música fractal, L-System, series fractales, fractales en la ciencia, farctales aleatorios y generación de paisajes, mapas, planetas y nubes. Hay que destacar que debido al interés de los alumnos, el tema de compresión de imágenes, que aún estando en el programa, se estudió con mayor profundidad de la que se pensaba en un principio. En este sentido varios de los trabajos realizados por los alumnos consistieron en el análisis e implementación de estos algoritmos con diferentes mejoras.

La creatividad mostrada por varios de los alumnos en la realización de las practicas, así como la calidad y trabajo de estas hacen valorar muy positivamente esta experiencia.

En general este tipo de asignaturas que se imparten en los cursos superiores, enfocadas adecuadamente a los intereses de los estudiantes de informática, llevan al alumno a interesarse por temas matemáticos para realizar su proyecto fin de carrera e incluso investigar en estos campos con la idea de desarrollar su tesis doctoral.

5.- BIBLIOGRAFÍA.

1. Barnsley M.F., *Fractals Everywhere.* Academic press, 1993
2. Barnsley M.F., Hurd L.P., *Fractal Image Compression.* AK Peters, Ltd., 1993
3. Bunde A. Y Havlin S. (Eds.) , *Fractals in Science.* Springer-Verlag, Berlin, 1994.
4. Falconer K.J., *Fractal Geometry.* Wiley, New York, 1990.
5. Fisher Y., *Fractal Image Compression.* Springer-Verlag, 1994.
6. Flake G-W., *The computational Beauty of Nature.* Bradford Book, Massachusetts, 1999.
7. Guzman M., Martín M.A., Morán M., Reyes M. *Estructuras fractales y aplicaciones.* Labor, Barcelona, 1993.
8. Mandelbrot B., *Fractals Geometry of Nature.* W.H. freeman and Co., New york, 1982.
9. Peitgen O.H., Jürgens H. y Saupe D., *Chaos and Fractals. New Frontiers of Science.* Springer-Verlag, Berlin 1992.
10. H.-O.Peitgen y P.H.Richter, *The beauty of fractals.* Springer-Verlag, Berlin, 1986.
11. Stewart, *I. The Magical Maze: Seeing the World Through Mathematical Eyes.* John Wiley & Sons, Inc. 1998.

HERRAMIENTA PARA LA ENSEÑANZA DE LA DEDUCCIÓN NATURAL.

Faraón Llorens y Sergio Mira

VGIA (Grupo de investigación Gráficos, Visión e Inteligencia Artificial)
Dpto. Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
e-mail: faraon@dccia.ua.es

RESUMEN: En el presente artículo se presenta una herramienta didáctica para la enseñanza de la lógica, en concreto para el aprendizaje de la técnica de inferencia de la *Deducción Natural*. **ADN** (Asistente para la Deducción Natural) es un programa, ejecutado vía web, que supervisa y comprueba que las fórmulas lógicas introducidas están sintácticamente bien construidas (*fbf*) y que la secuencia obtenida (pasos de la deducción) es correcta. También dispone de otras herramientas de soporte: árbol sintáctico, aconsejador, ayuda en línea, información detallada de errores, ...

1.- INTRODUCCIÓN

La lógica nos proporciona métodos de cálculo que nos permiten inferir, por simple manipulación sintáctica, nuevas fórmulas a partir de las conocidas. Uno de estos métodos es la Deducción Natural, cuyo mecanismo está muy cercano al razonamiento intuitivo del ser humano. Así, de forma sencilla, a partir de las fórmulas dadas y mediante la aplicación de reglas, obtenemos nuevas fórmulas. Vamos a presentar una herramienta didáctica de apoyo para el aprendizaje de la técnica de Deducción Natural. Dicha herramienta ha sido desarrollada en el departamento de Ciencia de la Computación e Inteligencia Artificial y se utiliza en las clases prácticas de la asignatura obligatoria *Lógica de Primer Orden*, impartida en el primer curso de la Ingeniería en Informática de la Universidad de Alicante [Llorens1996, Llorens98]. Esta herramienta pretende ser un instrumento didáctico que ayude a los estudiantes a escribir fórmulas lógicas bien formadas y a realizar deducciones correctamente. No se trata de un sistema que construya demostraciones de forma automática; simplemente es un asistente que guiará al estudiante. En la actualidad existe un variado número de herramientas que pueden servir de ayuda en el aprendizaje de la lógica [Goldson1993, Barwise1992, Barwise1994, TTL2000].

2.- ADN (*Asistente para Deducción Natural*)

El **A**sistente para **D**educción **N**atural es una herramienta didáctica que asiste al alumno que se inicia en la técnica de Deducción Natural a elaborar sus propias deducciones. Comprueba si la fórmula es sintácticamente correcta (*fbf*) y si se ha obtenido de forma adecuada (*reglas*

básicas). Además, posee las siguientes herramientas de apoyo: visor de árboles sintácticos de las fórmulas, informe detallado de los errores, visor de reglas básicas, aconsejador y ayuda en línea. ADN puede ser ejecutado con cualquier navegador, ya que se trata de un applet escrito en Java. Se encuentra disponible en internet en el sitio web:

http://www.dccia.ua.es/logica/ADN

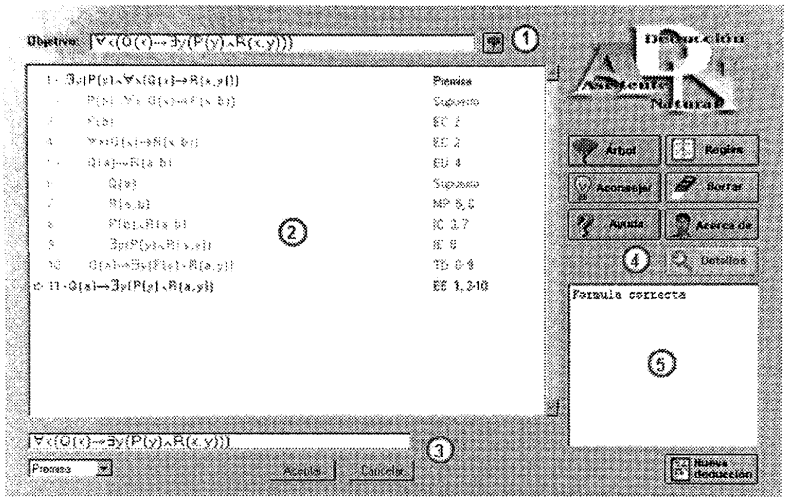


Ilustración 1: Ventana principal del ADN

En la Ilustración 1 se muestra la pantalla principal del ADN y las partes en las que se divide la misma:

1. Editor de la fórmula objetivo: nos permite editar la fórmula que será el objetivo de la deducción.

2. Cuerpo de la deducción (*pizarra*): en esta zona se irán visualizando los pasos de la deducción. Se pueden observar tres partes (dispuestas en columnas):

   - Numeración de las líneas, para poder hacer referencia a ellas.

   - Fórmulas lógicas que vamos obteniendo. Las sangrías indican que entramos en un nuevo supuesto (hacia la derecha) o que lo cancelamos (vuelta a la izquierda), configurando lo que llamaremos *subpruebas*.

   - Justificación de la fórmula obtenida mediante la aplicación de una regla básica a una o más fórmulas anteriores.

3. Editor de fórmulas: se utiliza para insertar nuevas fórmulas dentro de la deducción y decir de qué fórmulas han derivado las mismas.

4. Área de opciones del programa: aquí tenemos los botones que acceden a las diferentes funciones del asistente: Árbol, Aconsejar, Ayuda, Reglas, ...

5. Ventana de información: en este área se muestra información al usuario.

3.- EL LENGUAJE DE LA LÓGICA

La lógica pretende formalizar las expresiones del conocimiento humano. Y dicho conocimiento lo adquirimos y transmitimos por medio del lenguaje. Pero el lenguaje natural que utilizamos es ambiguo y engorroso, y por ello, para trabajar formalmente con el conocimiento, necesitamos un lenguaje artificial. En el lenguaje que utilizamos en la Lógica de Primer Orden descomponemos las frases en objetos y relaciones/propiedades, dando lugar a los conceptos de términos y predicados. ADN utiliza el alfabeto clásico de la Lógica de Primer Orden y la definición usual de fórmula bien formada (*fbf*).

Veamos un ejemplo. Sea la sentencia "hay una mujer que gusta a todos los hombres", que utilizando el lenguaje de la lógica de primer orden se correspondería con la siguiente fórmula bien formada (*fbf*):

$$\exists y \{ \text{mujer}(y) \wedge \forall x [\text{hombre}(x) \rightarrow \text{leGustaA}(x,y)] \}$$

Si utilizamos para los predicados las letras P (ser mujer), Q (ser hombre) y R (le gusta) quedaría:

$$\exists y \{ P(y) \wedge \forall x [Q(x) \rightarrow R(x,y)] \}$$

Representar la fórmula en forma de árbol etiquetado nos puede ayudar a ver claramente la estructura sintáctica de la misma. El árbol sintáctico de la fórmula bien formada $P(a) \rightarrow \exists y\ P(y)$, proporcionado por el programa ADN, lo podemos ver en la Ilustración 2. El árbol sintáctico de una fórmula lógica nos permite distinguir claramente cuál es el operador principal y las prioridades entre ellos. Esto nos ayudará en la posterior manipulación sintáctica cuando tengamos que aplicar las reglas de la deducción.

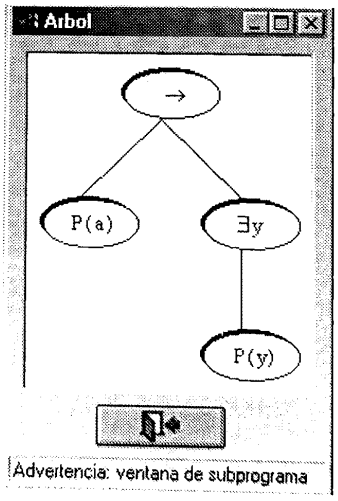


Ilustración 2 : Árbol sintáctico

4.- PASOS DE LA DEDUCCIÓN NATURAL

Uno de los aspectos fundamentales de la lógica es que, además de un lenguaje de representación y formalización de conocimiento, nos proporciona unas técnicas de razonamiento o inferencia, que nos permiten obtener nuevo conocimiento a partir del que ya poseemos. Existen distintos métodos para ello. Nosotros vamos a trabajar con la Deducción Natural [Garrido1995, Reeves1990, Socher-Ambrosius1997], sistema formal que a partir de unas premisas y con el único apoyo de unas reglas básicas, obtiene determinadas conclusiones. Así, si asumimos las premisas y cada paso elemental que damos lo justificamos con una regla básica, iremos obteniendo nuevas fórmulas lógicas que podemos asumir como conclusiones derivadas de las premisas. Todas las fórmulas lógicas que vamos introduciendo son visualizadas en la parte central de la ventana de ADN, que llamaremos *pizarra*. Una deducción la podemos considerar como un algoritmo que partiendo de unos valores de entrada (premisas) obtiene unas determinadas salidas (conclusiones) utilizando un conjunto dado de instrucciones (reglas) [Llorens1999].

a) Reglas básicas

Para determinar las *reglas básicas* de la Deducción Natural nos basaremos en el cálculo de Gentzen [Gentzen1934] que propone dos reglas (una de introducción y otra de eliminación) para cada símbolo lógico (conectivas y cuantificadores). Si la regla básica introduce en su conclusión una conectiva o cuantificador que no aparece en sus premisas será una regla de introducción; si elimina de su conclusión una conectiva o cuantificador que aparece en sus premisas será una regla de eliminación. Se intuye que si disponemos de procedimientos para añadir o quitar los distintos símbolos lógicos, podremos transformar por pura manipulación sintáctica las premisas en la conclusión. Así, aplicando un punto de vista de ingeniería, se trataría de desmontar las fórmulas lógicas que tenemos como premisas hasta obtener sus componentes básicas (fórmulas atómicas) y volver a montarlas en la configuración adecuada (fórmula lógica que queremos obtener como conclusión). Las reglas serían los instrumentos que nos permitirían montar y desmontar dichas fórmulas lógicas. En la ventana de reglas básicas visualizamos las reglas de introducción y eliminación correspondiente al operador lógico que esté seleccionado en la lista superior (Ilustración 3).

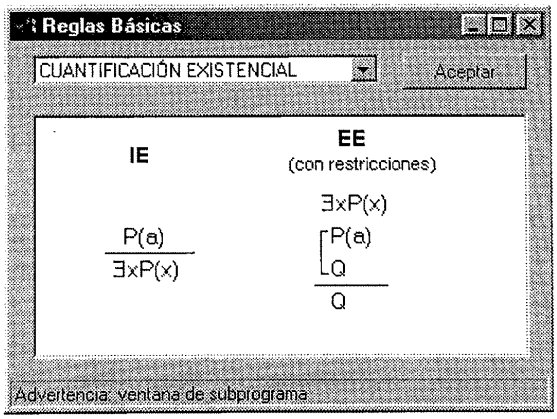


Ilustración 3: Ventana de Reglas Básicas

Tendremos una deducción correcta cuando consigamos una secuencia finita de fórmulas, donde cada una de las fórmulas ha sido obtenida mediante la aplicación de alguna regla de inferencia. En ADN, cada línea de nuestra deducción (y por tanto la fórmula lógica escrita en ella) estará "justificada" por la aplicación de una regla básica a alguna o algunas líneas anteriores. Dicha justificación aparece en la columna de la derecha. Por ejemplo, la justificación de la línea 11 de la Ilustración 1, *EE 1,2-10* significa que esa fórmula la hemos obtenido porque tenemos una fórmula cuantificada existencialmente en la línea 1, suponemos en la línea 2 que un individuo genérico del dominio (*b*) cumple dicha fórmula y llegamos en la línea 10 a una conclusión que no depende de la elección, por lo que podemos concluir la fórmula de dicha línea.

b) Subdeducciones

Otro de los aspectos cruciales de la deducción natural es el de las *subpruebas* (subdeducciones o subderivaciones). En cualquier paso de nuestra deducción podemos introducir un *supuesto provisional*, que debe ser cancelado en alguna línea posterior. Desde el supuesto hasta la cancelación tendremos una subdeducción. Así, en nuestras deducciones tendremos fórmulas a distintos niveles, que visualizaremos gráficamente mediante una sangría a la derecha. Los supuestos provisionales son una herramienta muy potente ya que nos permiten suponer lo que nosotros queramos. Pero debemos pagar un alto precio por ello: para poder finalizar una demostración deberemos haber cancelado todos los supuestos que hayamos hecho. Por tanto, la cancelación de supuestos provisionales se convierte en una pieza clave de las deducciones naturales. Cuando un supuesto es cancelado, sangramos a la izquierda y sombreamos en gris las fórmulas interiores de la subdeducción. Dichas fórmulas interiores serán inaccesibles a partir de este momento. La utilización de subdeducciones nos permite "modularizar" nuestras deducciones, planteándonos subobjetivos más sencillos que el objetivo final, y que en su conjunto nos lleven a la conclusión que buscamos.

5.- ASISTENTE

El ADN dispone de una herramienta muy útil a la hora de realizar una deducción. Está herramienta es el *Aconsejador*. El aconsejador analiza las fórmulas de la deducción e intenta guiarnos hacia el objetivo o decirnos qué reglas podemos aplicar. No nos llevará siempre a la solución, ya que la deducción natural en lógica de primer orden no es un problema decidible, simplemente es un apoyo. En cualquier momento podemos poner en marcha el aconsejador, y se puede dejar visible durante toda la deducción, con lo que se irá actualizando conforme añadamos o eliminemos fórmulas. En la Ilustración 4 se puede ver la ventana del aconsejador.

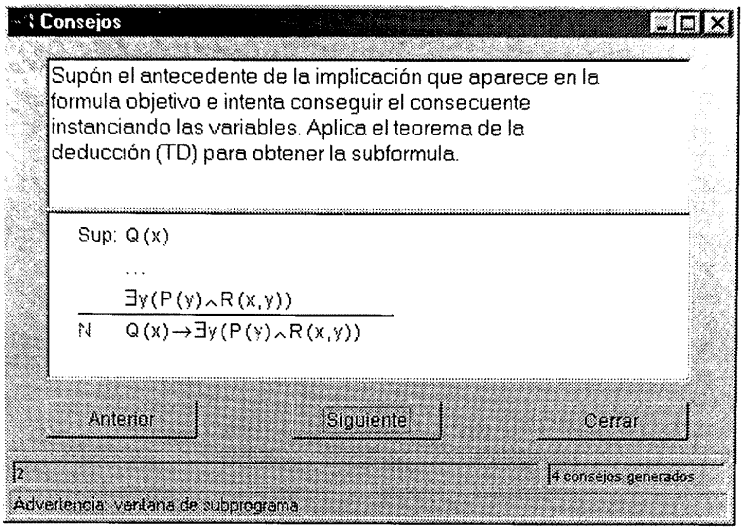


Ilustración 4: Ventana del *Aconsejador*

En la parte superior de la ventana se muestra una sugerencia de lo que podemos hacer y justo debajo da una pista gráfica con las fórmulas que tenemos en la deducción. Podemos navegar entre todos los consejos generados mediante los botones de Anterior y Siguiente. En la barra de estado del aconsejador podemos ver cuantos consejos se han generado (a la derecha) y cuál estamos visualizando actualmente (a la izquierda). El ADN muestra consejos generados de arriba-abajo (basados en las premisas y fórmulas previas) así como consejos generados de abajo-arriba (basados en la conclusión).

6.- EJEMPLO

En la Ilustración 5 podemos ver un ejemplo de deducción natural realizado mediante el *ADN*:
$\forall x\ P(x) \vee \forall x\ Q(x) \Rightarrow \forall x\ (P(x) \vee Q(x))$

BIBLIOGRAFÍA

[Barwise1992] J. Barwise and J. Etchemendy. *The Language of First-Order Logic (Tarski's World).* CSLI Lecture Notes n. 34, Stanford, 1992.

[Barwise1994] J. Barwise and J. Etchemendy. *Hyperproof.* CSLI Lecture Notes number 42, Stanford, 1994.

[Garrido1995] Manuel Garrido, *Lógica simbólica.* Ed. Tecnos, 3ªedic., 1995

[Gentzen1934] Gerhard Gentzen, "Untersuchungen über das logische Schliessen" (Investigaciones sobre la deducción lógica), Mathematische Zeitschrift, vol. 39, 1934.

[Goldson1993] D. Goldson, S. Reeves and R. Bornat. *A Review of Several Programs for the Teaching of Logic.* The Computer Journal, vol. 36, nº 4, 1993.

[Llorens1996] F. Llorens y Mª J. Castel. *Lógica de Primer Orden en las Ingenierías Informáticas*. II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías, I.C.E. U.P.M., 1996.

[Llorens1998] F. Llorens, F. Escolano, M. Pujol y O. Colomina. *Formalización del Razonamiento*. JENUI'98. IV Jornades sobre l'ensenyament universitari de la Informàtica, Andorra, 1998

[Llorens1999] F. Llorens, R. Satorre, F. Escolano y P. Arques. *Deducción Natural versus Computación*. JENUI'99. V Jornadas sobre la Enseñanza Universitaria de la Informática, Zaragoza, 1999

[Reeves1990] S. Reeves y M. Clarke, *Logic for Computer Science*. Ed. Addison-Wesley, 1990

[Socher-Ambrosius1997] R. Socher-Ambrosius y P. Johann, *Deduction Systems*. Ed. Springer-Verlag, 1997

[TTL2000] *Fist International Congress on TOOLS FOR TEACHING LOGIC*. Universidad de Salamanca, Junio 2000

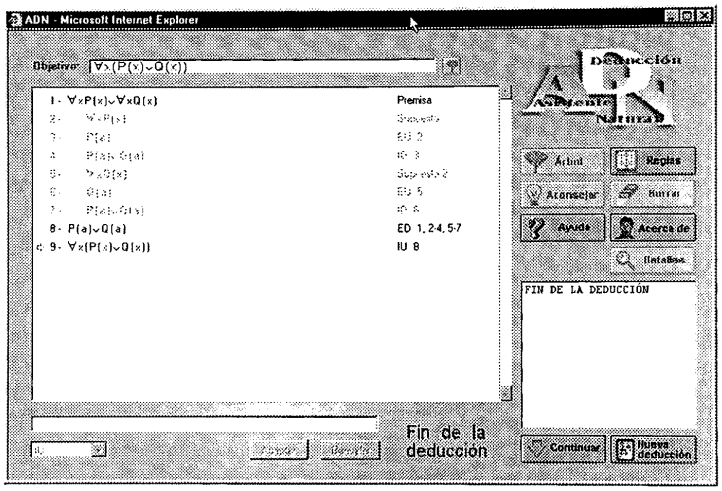


Ilustración 5: Ejemplo

UNA PROPUESTA PARA LAS PRÁCTICAS DE LA MATERIA "TEORÍA DE AUTÓMATAS Y LENGUAJES FORMALES".

José A. Gámez, Juan A. Guerrero, José M. Puerta

Departamento de Informática
Universidad de Castilla-La Mancha
Escuela Politécnica Superior
Albacete, 02071
e-mail: {jgamez,guerrero,jpuerta}@info-ab.uclm.es

RESUMEN: En este trabajo presentamos una propuesta para la realización de las prácticas relativas a la materia "Teoría de Autómatas y Lenguajes Formales", basadas en el uso de distintos programas que permiten no sólo el diseño y prueba de los distintos modelos de autómata y gramática, sino también la interacción con el alumno para la resolución paso a paso de distintos problemas (minimización de autómatas finitos, conversiones entre distintos modelos de autómata y/o gramática, etc...).

1.- INTRODUCCIÓN.

En los planes de estudios de Ingeniería Informática e Ingeniería Técnica en Informática de Sistemas, la materia troncal "Teoría de Autómatas y Lenguajes Formales" (TALF en adelante) tiene asignados 9 créditos. En los últimos años, siguiendo las directrices relativas a la proporcionalidad entre créditos teóricos y prácticos, se ha pasado de impartir estos 9 créditos de forma teórica en su totalidad, a tener que impartir un tercio de ellos de forma práctica.

Si bien las recomendaciones realizadas por las asociaciones ACM e IEEE para la docencia en prácticas de TALF, consisten en el uso de simuladores que permitan al alumno afianzar los conocimientos adquiridos en las clases teóricas mediante el diseño y prueba en un computador de los distintos modelos de autómata y gramática estudiados, la carencia de estos simuladores, la restringida capacidad de los mismos, o simplemente el desconocimiento de su existencia, motivó (y continua motivando en algunos casos) que los créditos prácticos se emplearan en la resolución de problemas en pizarra y en la realización de programas que implementen algunos de los algoritmos vistos en clase.

Desde nuestro punto de vista y teniendo en cuenta que en nuestro plan de estudios TALF se imparte en segundo curso, coincidiendo en el tiempo con asignaturas tales como Metodología de la Programación, Algoritmica y Estructuras de Datos, consideramos que no es TALF el lugar adecuado para la realización de prácticas de programación. Por otra parte, la resolución de problemas en pizarra es imprescindible en algunos temas, aunque nuestra experiencia nos indica que en aquellos en los que es posible sustituirlos por prácticas de laboratorio, esta sustitución es ventajosa, ya que el alumno se motiva más y tiene una mayor participación respecto a las clases

de problemas en pizarra, en las que (la mayoría) suele limitarse a copiar la resolución realizada por el profesor o por otro compañero.

Una vez que hemos planteado nuestra opinión, nos fijamos dos objetivos para el resto del trabajo:

1. Realizar una propuesta de planificación temporizada de las prácticas para TALF. Para ello hemos considerado un temario clásico de la asignatura (ver sección 5) que puede ser cubierto, por ejemplo, usando los libros de texto de Hopcroft y Ullman, Isasi y col., y Lewis y Papadimitriou. La temporización se ha realizado considerando que la asignatura es cuatrimestral con una duración de 14 semanas (4 horas de teoría + 2 horas de prácticas).

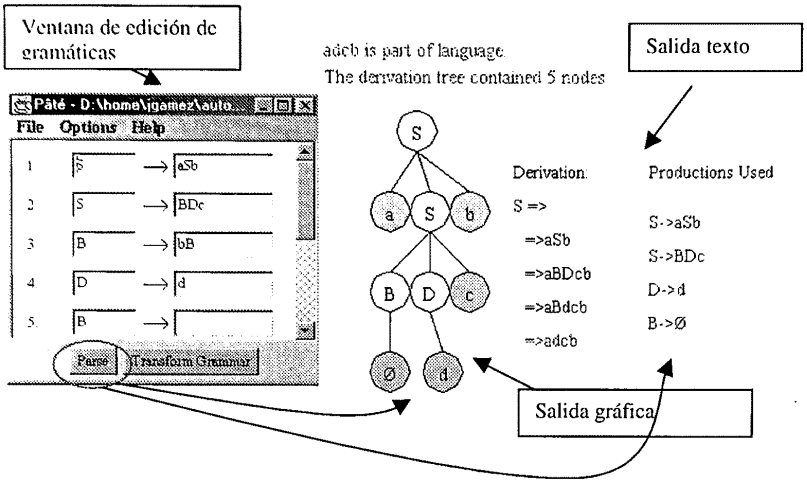
2. Asociar a cada una de las prácticas la herramienta con la cual llevarla a cabo. Estas herramientas han sido desarrolladas por el grupo de Susan Rodger en la Universidad de Duke (Rodger, 2000). Aunque estas herramientas pueden ser obtenidas en Internet, dada la grandeza (física) de la red, es difícil encontrarlas a menos que se tenga conocimiento de ellas. Nuestro objetivo, por tanto, es dar a conocer estas (a nuestro juicio) excelentes herramientas.

A continuación introducimos brevemente las herramientas utilizadas en las prácticas, proponemos una planificación temporizada (y sincronizada con la teoría) de las mismas, y finalmente presentamos algunas conclusiones.

2.- PÂTÉ.

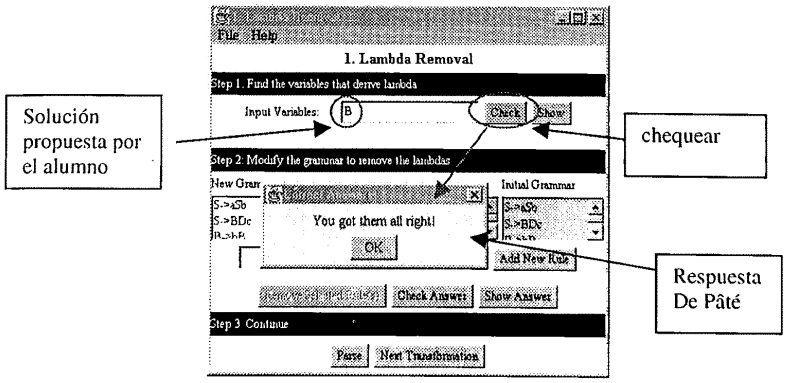
Pâté (**http://www.cs.duke.edu/~rodger/tools/pateweb**) es una herramienta visual e interactiva para el análisis y transformación de gramáticas desarrollada en la Universidad de Duke (Hung y Rodger, 2000). Pâté esta implementado en JAVA por lo que no necesita instalación y puede ejecutarse sobre cualquier plataforma en la que esté instalado el JDK 1.2.

La clase de gramáticas a analizar no se limita como ocurre en otras herramientas a las libres de contexto, si no que podemos escribir también gramáticas de tipo 1 o 0. En la siguiente figura podemos ver la ventana de edición de gramáticas y como al pulsar sobre el botón `parse` podemos proceder al análisis de una cadena. Inicialmente obtenemos un mensaje indicando si la cadena pertenece o no al lenguaje generado por la gramática. En caso positivo podemos ver (completamente o paso a paso) el árbol de análisis o la secuencia de derivaciones que genera la cadena.



En relación con la transformación de gramáticas, Pâté permite "limpiar" la gramática inicial mediante la eliminación de producciones nulas, producciones unitarias y producciones inútiles, así como la transformación de la gramática ya limpia a Forma Normal de Chomsky.

Evidentemente, introducir la gramática y pulsar un botón para obtener la gramática transformada no tiene mucho interés didáctico, a menos que el alumno compare el resultado con el obtenido por él. Sin embargo, Pâté permite la interacción por parte del alumno con el proceso de transformación, permitiendo al alumno introducir la solución parcial y chequear si su respuesta es válida. La siguiente figura muestra un ejemplo en el proceso de eliminación de producciones nulas, para lo que el alumno debe (1) identificar el conjunto de símbolos anulables, y (2) dar el conjunto de reglas de producción resultantes.



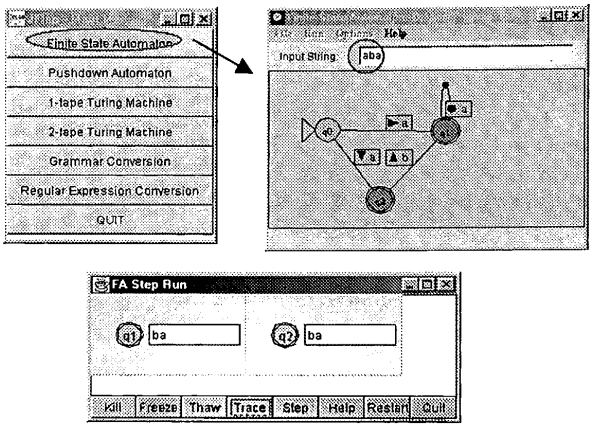
3.- JFLAP.

JFLAP (Java Formal Language & Automata Package) es un paquete de herramientas gráficas para asistir la docencia en TALF. Al igual que Pâté ha sido diseñado e implementado en JAVA

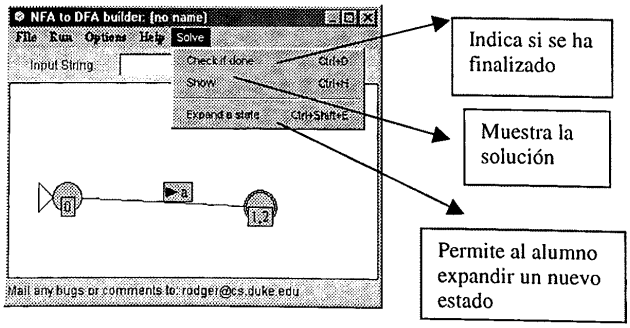
en la Universidad de Duke (Bilska y col., 1997). La última versión (1999) puede ser descargada de **http://www.cs.duke.edu/~rodger/tools/jflap.**

Las capacidades de JFLAP pueden dividirse en dos grupos:

1. Diseño y prueba (simulación) de diversos modelos de autómata: autómatas finitos (deterministas, no deterministas, y con transiciones nulas), autómatas con pila, máquinas de Turing (con una y dos cintas), expresiones regulares y gramáticas (regulares y libres del contexto). La siguiente figura muestra el menú principal de JFLAP, la ventana de edición de autómatas y un ejemplo tomado de la traza del proceso de reconocimiento de la cadena "aba" tras leer la primera "a".



2. Conversiones entre los distintos modelos. Así, podemos pasar de un tipo de autómata finito a otro, transformar un autómata en gramática regular o en expresión regular (y viceversa), minimizar autómatas finitos deterministas y, transformar autómatas con pila en G.L.C. (y viceversa). Al igual que ocurría en Pâté la ventaja (de cara a la docencia) es la posibilidad de que el alumno vaya construyendo su solución y verificándola con la ayuda de JFLAP. La siguiente figura forma parte de la transformación del AFND anterior en un AFD.



4.- MTURING.

MTURING es un programa desarrollado por Luis Jiménez Linares (Jiménez, 2000) en la Escuela Superior de Informática de la Universidad de Castilla-La Mancha. El programa dispone de una interfaz modo texto en la que se puede visualizar el estado y evolución de la cinta, así como la posición de la cabeza lectora/escritora, el estado actual y la transición que se está ejecutando en cada momento. MTURING permite definir (mediante la especificación de un fichero texto conteniendo sus transiciones) autómatas finitos y máquinas de Turing. En nuestro caso no usaremos MTURING con este fin, ya que JFLAP incluye estas capacidades, sino que usaremos MTURING exclusivamente para la última práctica de nuestra propuesta: *computación de funciones mediante esquemas de máquinas de Turing*. MTURING permite definir esquemas siguiendo la notación propuesta en (Lewis y Papadimitriou, 1998), es decir, permite definir un esquema como un AFD en el que los estados son máquinas de Turing (previamente definidas y etiquetadas).

Para hacer factible el desarrollo de la práctica, se facilitará al alumno un fichero con la definición de las máquinas básicas, es decir, máquinas de impresión de símbolos (P_*, P_1), máquinas de desplazamiento (R, L, R_*, L_*) y máquinas de copia y desplazamiento de cadenas (K, T), solicitándosele al alumno que implemente esquemas para la computación de funciones.

5. PLANIFICACIÓN DE LAS PRÁCTICAS.

Las prácticas de laboratorio se desarrollan de forma paralela a las de clase, pero con un desfase de al menos una semana entre la teoría explicada y la práctica correspondiente, con el objetivo de que el alumno tenga tiempo de asimilar la materia y de trabajar sobre las hojas de ejercicios que luego se usarán en las sesiones prácticas. En cada práctica se proponen dos tipos de tareas: comprobación y simulación de ejercicios resueltos previamente (diseño de autómatas, gramáticas, transformaciones, etc.), y resolución de ejercicios en los que, o bien se requiere gran cantidad de cálculos, o bien es interesante ver el proceso de resolución (minimización de AFD's, análisis y síntesis de lenguajes mediante el teorema de Kleene, etc.). En el siguiente cuadro se resume la temporización propuesta:

Semana	Teoría	Práctica
1	0. *Presentación* (1h) 1. *Introducción* (1h) 2. *Lenguajes Formales* (2h)	Preparación y organización de los grupos de prácticas
2	3. *Gramáticas Formales* (3h) Definición, conceptos, jerarquía de Chomsky.	
3	4. *Autómatas Finitos* (4h). AFD's, minimización de AFD's, AFND's.	Pr. 1: Construcción (y prueba) de gramáticas. [PÂTÉ]
4	4. *Autómatas Finitos* (3h). AFND-ε. Equivalencia AFD↔AFND↔AFND-ε. Equivalencia AF↔Gram. Regular. 5. *Otros modelos de AF* (1h) Máquinas secuenciales.	Pr. 2: Diseño y prueba de distintos AFDs. [JFLAP]
5	5. *Otros modelos de AF* (3h). Equivalencia Moore↔Mealy. Aut. Probabilísticos. 6. *Expresiones Regulares* (1h). Definición, ejemplos y propiedades.	Pr. 3: Minimización de AFDs (comprobación). Diseño de AFNDs. [JFLAP]
6	6. *Expresiones Regulares* (4h) Equivalencia: AF ↔ ER (Teor. De Kleene), AF→ER (ecuaciones), ER→AF (derivadas).	Pr. 4: Diseño de AFND-ε. Transformaciones AFD↔AFND↔AFND-ε (comprobación). [JFLAP].
7	7. *Propiedades de los Leng. Reg.* (4h). Lema de bombeo, propiedades de cierre, algoritmos de decisión.	Pr. 5. Gram. regulares y AFs. AFs y expresiones regulares (Kleene). (Comprobación) [JFLAP]
8	8. *Gramáticas Libres del Contexto* (4h) Transformación (limpieza) de GLCs. Formas normales (Chomsky y Greibach)	Pr. 6: AF→ER, ecuaciones características. ER→AF, derivadas. (comprobación) [JFLAP]
9	8. *Gramáticas Libres del Contexto* (1h) 9. *Autómatas con Pila* (3h) Motivación, conceptos y ejemplos.	Pr. 7: Transformación de Gramáticas libres de contexto. (comprobación) [PÂTÉ]
10	9. *Autómatas con Pila* (2h). Equivalencias $L_F \leftrightarrow L_V$, GLC↔AP 10. *Propiedades de los L.L.C.* (2h) Propiedades de iteración (lema de bombeo).	Pr. 8: Diseño y prueba de distintos autómatas con pila. [JFLAP]
11	10. *Propiedades de los L.L.C.* (2h) Propiedades de cierre y algoritmos de decisión. 11. *Máquinas de Turing* (2h) MT como reconocedor de lenguajes, conceptos y ejemplos. Variaciones al modelo de MT.	Pr. 9: Equivalencias $L_F \leftrightarrow L_V \leftrightarrow L_{VF}$, GLC↔AP (Comprobación). [JFLAP]
12	11. *Máquinas de Turing* (4h) MT y lenguajes de tipo 0. Aut. Linealmente acotados. MT como computador de funciones.	Pr 10: Diseño y prueba de MTs para reconocer diversos lenguajes. [JFLAP]
13	11. *Máquinas de Turing* (4h) Esquemas de MT. MT Universal, problema de la parada.	Pr. 11: Diseño y prueba de MTs para computar distintas funciones [JFLAP]
14	12. *Funciones Recursivas* (4h) Recursión primitiva, μ-recursión, equivalencia con MT.	Pr. 12: Computación de funciones usando esquemas de MT. [MTURING]

6. CONCLUSIONES.

Hemos pretendido, sobre todo, manifestar que la docencia de TALF puede apoyarse en un sistema de prácticas sin tareas de programación adicional. Para ello, hemos propuesto el uso de un conjunto de herramientas que permite tanto la simulación como la resolución-visualización

de problemas: PÀTÈ, JFLAP y MTURING. Además, hemos diseñado un temario de prácticas que permite usar el entorno, de forma regular, a lo largo de todo el curso, sin descartar ninguno de los temas esenciales.

7. REFERENCIAS.

(Bilska y col., 1997) A. Bilska, K. Leider, M. Procopiuc, O. Procopiuc, S. Rodger, J. Salemme y E. Tsang. *A Collection of Tools for making Automata Theory and Formal Languages Come alive*. Twenty-eight SIGCSE Technical Symposium on Computer Science Education, pp. 15-19. 1997.

(Hopcroft y Ullman, 1979) J.E. Hopcroft y J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

(Hung y Rodger, 2000) T. Hung y S. Rodger. *Increasing Visualization and Interaction in the Automata Theory Course*. Thirty-first SIGCSE Technical Symposium on Computer Science Education, pp. 6-10. 2000.

(Isasi y col., 1997) P. Isasi, P. Martínez y D. Borrajo. *Lenguajes, Gramáticas y Autómatas. Un enfoque práctico*. Addison-Wesley, 1997.

(Jiménez, 2000) L. Jiménez. Página en Internet. **http://oreto.inf-cr.uclm.es/personas/ljimenez/index.html**

(Lewis y Papadimitriou, 1998) H. Lewis y C. Papadimitriou. *Elements of the Theory of Computation (2ª ed.)*. Prentice-Hall, 1998.

(Rodger, 2000) S. Rodger. Página en Internet. **http://www.cs.duke.edu/~rodger**

UTILIZACIÓN DEL MATLAB PARA LA REALIZACIÓN DE ANIMACIONES QUE SIMULEN PROCESOS ITERATIVOS PARA LA RESOLUCIÓN DE ECUACIONES NO LINEALES

Fernández Gutierrez, M.J., García Gonzalo, M.E.

Departamento de Matemáticas. Universidad de Oviedo
e-mail: mjfg@pinon.ccu.uniovi.es

RESUMEN: En el presente artículo se desarrollan simulaciones de procesos iterativos de punto fijo, mediante animaciones en tiempo real realizadas con MATLAB. Así se muestran las posibilidades didácticas de dicho programa en este tema ya que permite la visualización de gran variedad de casos y ejemplos, contemplando la convergencia o divergencia, velocidad y orden de convergencia, así como casos prácticos aplicados a la vida real.

1.- INTRODUCCIÓN.

Se desarrolla la presentación de un tema de análisis numérico utilizando los medios informáticos PowerPoint y Matlab. Básicamente consta de una gran variedad de ejemplos del tema de resolución de ecuaciones no lineales, comparando casos parecidos, mostrando diferencias y señalando analogías. En la bibliografía aparece algún ejemplo gráfico, pero limitándose a los casos más básicos. El Power Point se ha utilizado como un apoyo para escribir las fórmulas y desarrollo matemático. El eje principal de la exposición son las gráficas y sobre todo las animaciones, ambas realizadas con Matlab, que muestran como evolucionan las distintas sucesiones hacia la solución del problema. Se considera que en este tema en particular las animaciones son importantes debido a que así se puede visualizar como a partir de una situación inicial parecida la evolución puede ser totalmente distinta. Además las animaciones se autoexplican mientras que si se presenta el mismo ejemplo con una gráfica implica un desglose y a veces una profusa explicación de la misma.

Las animaciones dan un acceso sencillo e intuitivo a secciones del tema que habitualmente no se tratan, contribuyendo así no sólo a aclarar y fijar los conceptos habituales sino también a su ampliación y comparación con los nuevos.

Otra ventaja es que dada la sencillez y brevedad de los programas pueden ser realizados y modificados por prácticamente todos los alumnos (en nuestro caso segundo curso de ingeniería

técnica informática) y repetidos tantas veces como sea necesario incluso en un ordenador con pocos requerimientos tanto de procesador como de memoria (se desarrollaron en un ordenador con procesador 486 con 8M de RAM), ya que los programas ocupan muy poco espacio y generan la animación en el instante en que se corre el programa.

2.- EL MÉTODO DE ITERACIÓN DE PUNTO FIJO.

a) Antecedentes matemáticos

La resolución de una ecuación $F(x) = 0$ es uno de los problemas matemáticos más antiguos y comunes. Fórmulas exactas, entendiendo por exactas aquellas en las que la solución puede expresarse por medio de radicales, sólo hay en un número reducido de casos, como el de las ecuaciones polinómicas hasta grado cuatro inclusive. En el resto de los casos, en general, la solución ha de obtenerse de forma aproximada. Aunque existen métodos para la resolución numérica de determinados tipos de ecuaciones, como por ejemplo las polinómicas, tiene gran interés la construcción de métodos numéricos que permitan la aproximación de raíces en el caso general. Este es el caso de los métodos conocidos como de iteración de punto fijo que consisten en la transformación de la ecuación inicial en otra del tipo $f(x) = x$, cuyas raíces se denominan puntos fijos, y la utilización para la resolución del problema de la sucesión $x_{n+1} = f(x_n)$. Si la función f es continua y la sucesión tiene límite, éste es un punto fijo de f lo que justifica la utilización del método.

Pero no todas las funciones con un punto fijo generan una sucesión que converge a dicho punto. En general interesa que $f(x)$ varíe poco al variar x: el caso ideal sería f constante; entonces, para x_0 arbitrario x_1 nos daría exactamente el valor buscado.

Se obtendrán condiciones para la convergencia global del proceso de iteración de punto fijo, que en síntesis se trata de la contractividad de la función en un intervalo que se aplica sobre sí mismo utilizando la función de iteración. Con estas condiciones se consigue, no sólo garantizar la convergencia de la sucesión al punto fijo del intervalo sea cual sea el punto inicial tomado dentro de este, sino la unicidad del punto fijo en él. Además, la constante de contractividad asociada a la función y al intervalo sirve para establecer una cota del error.

Sin embargo la condición de contractividad es difícil de verificar en la práctica, y se sustituye por la condición, más exigente pero más sencilla de utilizar, de que el valor absoluto de la derivada de la función de iteración sea menor que uno en el intervalo.

Localmente hablamos de puntos fijos atractivos y repulsores. Esta cualidad viene determinada por el valor de la derivada de la función de iteración f en el punto fijo $\bar{x}$. Si la derivada de f es continua en un entorno de $\bar{x}$ y $|f'(\bar{x})| < 1$, entonces, si partimos de un punto suficientemente cercano a $\bar{x}$ la sucesión converge y el punto se llama atractivo. Si $|f'(\bar{x})| > 1$ sucede lo contrario y el punto es repulsor. Si $|f'(\bar{x})| = 1$ es un caso dudoso.

Una vez determinada la convergencia de un método iterativo es interesante conocer la velocidad de convergencia del mismo, para lo cual se define el concepto de orden de convergencia y se

estudia su obtención para los métodos de iteración de punto fijo. Se demuestra que el orden de iteración depende del orden de la primera derivada no nula de f en $\bar{x}$. Con esto se retoma la idea de que interesa una función que varíe lo menos posible: si $f'(\bar{x})=0$, la función tiene tangente horizontal en el punto y la función varía muy poco en su entorno; si además, la función tiene curvatura cero en el punto ($f''(\bar{x})=0$) y por lo tanto es plana en él, esta variación es aún menor; y en general, cuanto mayor sea el orden de la primera derivada no nula en $\bar{x}$, menos variará la función en un entorno de $\bar{x}$.

Las condiciones para la convergencia local exigían que $|f'(\bar{x})|<1$. Si $f'(x)=0$ y f' continua, la convergencia local está garantizada sea cual sea el orden de la primera derivada que no se anule. Cuanto mayor sea el orden de convergencia menor el número de iteraciones. También influye, aunque en menor medida, el coeficiente asintótico de convergencia que sólo tiene interés si estamos comparando sucesiones del mismo orden de convergencia.

El orden de convergencia comienza a ser interesante cuando es como mínimo de orden dos, así que una vez establecidas las condiciones que ha de reunir la función de iteración f para tener convergencia de orden dos, se construirá una función de iteración con este orden. De las diferentes posibilidades se toma la función de iteración asociada al método de aproximación de raíces de Newton-Raphson.

Otro punto de partida para alcanzar el mismo resultado podría ser geométrico: partimos de una raíz aproximada x_o y para hallar la siguiente aproximación de la raíz de la curva, sustituimos la curva $F(x)$ por la recta tangente en x_o, tomando como siguiente aproximación de la raíz buscada la raíz de la recta.

La convergencia local está asegurada por la construcción del método, pero sería útil conocer las condiciones que ha de reunir la función $y=F(x)$ en un intervalo $[a,b]$ para que partiendo de un punto cualquiera de él la convergencia esté garantizada. Se observa que la presencia de puntos singulares como máximos, mínimos y puntos de inflexión perjudica en muchos casos la convergencia del método de Newton y en otros hace que converja a una raíz que no interesa, por lo que, básicamente, las condiciones son que la función tenga una única raíz en $[a,b]$, no presente puntos singulares en el intervalo y que el corte de la tangente con el eje OX en el punto de menor pendiente del intervalo esté dentro de $[a,b]$.

Estas condiciones producen cuatro casos diferentes que se pueden reducir a uno: en nuestro caso el de la función F creciente y concava hacia abajo en el intervalo $[a,b]$. Si partimos de un x_o de la zona de $[a,b]$ donde $f''(x_o)f(x_o)<0$, se genera una sucesión creciente y acotada superiormente, por lo tanto convergente. Y además el límite es el punto fijo de f. Si por el contrario, partimos de un x_o tal que $f''(x_o)f(x_o)>0$, x_1 está al otro lado de la raíz con lo que nos remitimos al caso anterior.

Aunque se ha construido el método de Newton buscando que su orden de convergencia sea dos, existen casos desfavorables como el de raíces múltiples donde el orden de convergencia del método se reduce a uno. Schöder propuso una modificación del método que hace que este retome su orden de convergencia cuadrático. El inconveniente es que es preciso conocer a priori el orden de multiplicidad de la raíz. Otra situación particular, aunque ya no desfavorable, se produce cuando la raíz de $F(x)=0$ es un punto de inflexión con tangente no horizontal. En este caso el orden de convergencia es superior a dos.

Aunque un método de orden dos supone, en general, la necesidad de muy pocas iteraciones para aproximar la solución con gran exactitud, se plantea la obtención de métodos de orden superior. Así, partiendo de un planteamiento geométrico análogo al del método de Newton, se describe la obtención del método de Euler: en lugar de sustituir la curva por la recta tangente la sustituimos por la parábola tangente a la curva con su misma curvatura en el punto correspondiente a la raíz aproximada. Es importante la forma en que se expresa el algoritmo, puesto que una forma inadecuada conduce a importantes errores de redondeo que invalidan la fórmula.

Se plantean como modificaciones del método de Euler los métodos de Halley y de Chebyshev, todos ellos de orden tres.

b) Desarrollo gráfico

Se comienza con una descripción gráfica del planteamiento del problema, mostrando la diferencia entre la función asociada a la ecuación y la correspondiente función de iteración. Se señala cómo la raíz de la ecuación coincide con el punto fijo de la función de iteración. Se describe brevemente en que consiste el método de iteración de punto fijo. A continuación se muestran tanto gráfica como analíticamente distintas posibilidades de funciones de iteración para una misma ecuación.

Pero no todas las funciones de iteración son válidas para resolver el problema. Es necesario que la función de iteración genere una sucesión y que esta converja al punto fijo. Esto se expresa por medio de tres condiciones. Se van estudiando los requerimientos uno a uno y se proponen condiciones para cada uno de ellos. Conforme imponemos condiciones vamos viendo gráficamente cómo se van eliminando posibles funciones de iteración. Al final, las condiciones que ha de reunir la función de iteración pueden limitarse a dos. Se expresa de formalmente el teorema de convergencia global. De él se infieren gráficamente condiciones de convergencia local. Las gráficas ejemplo permiten apreciar la importancia de la elección de un intervalo simétrico centrado en el punto fijo en la demostración del teorema correspondiente.

A continuación se ve una animación de los típicos cuatro casos (convergente/divergente, escalera/tela de araña) de sucesiones generadas por el método, resaltando la ausencia de convergencia cuando no se cumplen las condiciones de convergencia local aunque el punto inicial de la sucesión esté cerca de la solución.

Las diferencias entre distintas velocidades de convergencia y distinto orden de convergencia con sus condiciones se pueden ver en la siguiente animación. Así se observa que para orden de convergencia uno la velocidad de convergencia aumenta al "tumbar" la curva (al disminuir la derivada) y como se pasa a un orden de convergencia dos cuando la curva tiene tangente horizontal en el punto fijo (la derivada es cero). También se aprecia que cuando la curva es plana en este punto (curvatura cero, derivada segunda cero) se pasa al orden de convergencia tres. Así, a partir de una observación gráfica, se da un criterio analítico para aumentar el orden de convergencia (derivadas sucesivas de la función de iteración cero en el punto fijo).

Con las condiciones analíticas obtenidas se plantea la construcción de un método iterativo de orden dos, el de Newton-Raphson. Se deduce la función de iteración del método a partir de la función correspondiente a la ecuación. No obstante nos parece más interesante en el actual contexto, la segunda opción de construcción del método que se presenta, que tiene como no, una aproximación gráfica. En ella, en cada paso se sustituye la curva que representa la función por la aproximación que constituye una recta tangente a dicha curva.

La convergencia local del método de Newton-Raphson está asegurada por la construcción del método, pero parece interesante estudiar condiciones de convergencia global. Cómo partida se muestra en una animación cómo perjudica a la convergencia del método la existencia en las proximidades de la raíz de máximos, mínimos y puntos de inflexión. Esto permite motivar las tres primeras condiciones del teorema de convergencia global. La cuarta se puede justificar diciendo que puesto que estamos hablando de convergencia global toda la sucesión ha de estar incluida dentro de un intervalo $[a,b]$ conocido. La cuarta condición está orientada a garantizar que incluso si partimos del punto más desfavorable del intervalo, el siguiente está también dentro de él. A continuación se demuestra analíticamente el teorema ilustrando gráficamente cada uno de los pasos de ésta demostración.

Junto con las animaciones que presentan el método de Newton-Raphson con esta interpretación geométrica tradicional, es decir, con F, se presentan también las animaciónes con la función de iteración correspondiente f, que permiten apreciar mejor el orden de convergencia. Esta forma de contemplar para un mismo ejemplo la sucesión utilizando F y f permite apreciar mejor cómo aunque en general el orden de sucesión generada por la función de iteración de Newton-Raphson es dos hay casos donde esto no sucede. Así, se ilustra el caso de raíces múltiples, donde el orden de convergencia se convierte en uno y se ve como modificando de forma adecuada la función de iteración se recupera el orden de convergencia dos (modificación de Shöder). También se ve un caso más favorable, donde el orden de convergencia del método de Newton-Raphson es al menos tres, que es el caso en el que la raíz es un punto de inflexión con tangente no horizontal.

Las animaciones permiten abordar la siguiente parte del tema de forma intuitiva y por analogía con el método de Newton-Raphson. En el método de Newton-Raphson sustituíamos en cada paso la curva correspondiente a la ecuación por la recta tangente a la curva en el punto de la sucesión (aproximación de la raíz). El método de Euler aproxima la curva con la parábola tangente en el punto y que tiene en dicho punto la misma curvatura. Se ve con una animación, cómo efectivamente este método es más rápido en su convergencia que el de Newton-Raphson.
Ya de forma analítica se ven otros dos métodos de orden de convergencia tres, el de Halley y el de Chebyshev, planteados cómo aproximaciónes del método de Euler.

Por último, cómo motivación para los alumnos, se ve un caso de la vida real. Se aplica la resolución de ecuaciones al cálculo de las condiciones de una hipoteca o préstamo (quién no ha tenido que pedir un préstamo para comprar un coche o un piso). Se supone que partimos de un presupuesto mensual aproximado para afrontar el préstamo y se resuelve el problema tanto analítica como gráficamente con las condiciones de interés y cuota que se quiera. El resultado será el número de meses que tendremos que estar pagando dicho préstamo.

3.- CONCLUSIONES

La finalidad principal de este trabajo es mostrar la utilización de métodos informáticos sencillos y accesibles a la enseñanza y aprendizaje de las matemáticas.

Supone un avance sobre el papel impreso en el sentido de que básicamente se trata de animaciones que en gran parte se explican por sí mismas. Así que no es sólo la visión de la imagen estática; a ella se añade el movimiento que permite no sólo asimilar situaciones sino también procesos. El alumno también puede actuar para modificar e incluso construir las animaciones con lo que se añade una componente activa por su parte. Así no sólo se facilita el apoyo del aprendizaje de la materia principal sino que permite el acceso intuitivo y comparado con vistas a ampliaciones de la misma.

4.- BIBLIOGRAFIA

- Aubanell, A., Benseny, A. y Delshams, A.(1993) *Útiles básicos de cálculo numérico. Labor.*
- Hamming, R.W.(1986). *Numerical Methods for Scientists and Engineers.* Dover.
- Hildebrand, F.B. (1956). *Introduction to Numerical Analysis.* Dover Publications.
- Jaluria, Y.(1988). *Computer Methods for Engineering.* Allyn and Bacon.
- Pérez, C. (1996) *Matemática informatizada con MATLAB.* RA-MA.
- Sanz-Serna,(1998) J.M. *Diez lecciones de Cálculo Numérico.* Universidad de Valladolid.
- Viaño Rey, J.M. (1997). *Lecciones de Métodos Numéricos. Resolución de ecuaciones numéricas.* Tórculo.

CREACIÓN DE UNA HERRAMIENTA DIDÁCTICA DE APOYO AL ESTUDIO DE SERIES Y CÁLCULO INTEGRAL UTILIZANDO EL *MATHEMATICA*.

Montserrat Galitó[1], Joan Gimbert[2], Héctor Sánchez[3], Magda Valls[2]

Departament de Matemàtica
Escola Universitària Politècnica
Universitat de Lleida
[1]*h4373173@alumnes.eup.udl.es*
[2]*{joangim,magda}@eup.udl.es*
[3]*r4374538@alumnes.eup.udl.es*

RESUMEN: Motivados por la necesidad de facilitar al estudiante la comprensión de los conceptos básicos del Cálculo, hemos diseñado e implementado una herramienta que permita mostrar animadamente tales objetos, empleando para ello el *Mathematica* como programa base. En esta comunicación presentamos la primera parte de nuestro trabajo, dedicado al estudio de las sucesiones y series, tanto numéricas como de funciones, y al cálculo integral.

1.- INTRODUCCIÓN.

Ciertamente las nuevas tecnologías han motivado cambios en la enseñanza de las Matemáticas. Por un lado, en los programas de las asignaturas de Matemáticas, especialmente en las titulaciones de Informática, se constata un mayor peso específico de los contenidos algorítmicos y numéricos. Por otro lado, la existencia de un amplio *software* matemático sugiere dar mayor relevancia a la asimilación de conceptos que a la simple acumulación de técnicas. Ahora bien, a pesar de las múltiples propuestas de integración de las herramientas informáticas en la enseñanza de las Matemáticas, no parece haber aún una respuesta definitiva a cuál debe ser el correcto uso de dichas herramientas con el fin de lograr una mejor comprensión de las matemáticas y de sus aplicaciones.

Nuestra experiencia en la enseñanza de las materias de Análisis Matemático y Cálculo Numérico en las titulaciones de Informática de nuestra universidad, nos plantea dos retos: transmitir mejor las ideas intuitivas que hay detrás de las nociones básicas del Análisis, en las cuales se sustentan los métodos numéricos más elementales, e insistir más en la complementación de los aspectos analíticos y numéricos dando así una visión más realista del Cálculo.

Una manera natural de ilustrar las ideas de aproximación y de límite, comunes en muchos de los procedimientos de Cálculo, es mediante representaciones gráficas animadas acompañadas de una información numérica. Para la realización de dichas animaciones, hemos optado por utilizar el *Mathematica*, programa que integra el cálculo numérico y simbólico con las representaciones gráficas y que además ofrece un lenguaje de programación de alto nivel con el cual se han implementado una amplia librería de aplicaciones.

2.- OBJECTIVOS Y METODOLOGÍA

Nuestro proyecto de diseño y desarrollo de aplicaciones informáticas de soporte a la enseñanza del Cálculo, inicialmente con una sola variable, está estructurado en subproyectos correspondientes a los siguientes bloques temáticos: fundamentos (números reales, límites y continuidad, funciones elementales), cálculo diferencial, cálculo integral, sucesiones y series. Dichos subproyectos, propuestos y coordinados como Trabajos Finales de Carrera en las titulaciones de Ingeniería Técnica en Informática de la *Escola Universitària Politècnica* de la *Universitat de Lleida*, persiguen los siguientes objetivos:

- Determinar qué conceptos y resultados matemáticos admiten una representación gráfica o numérica, y en forma animada, que facilite su comprensión. Analizar las diferentes representaciones y animaciones con el fin de concluir cuál de ellas resulta más adecuada en cada situación.

- Especificar las decisiones globales de diseño de la aplicación: cómo mostrar la información gráfica y numérica, dominio de los parámetros, opciones, etc. Implementar las funciones, en el entorno de *Mathematica*, mediante una programación lo más sencilla, clara y elegante que permita dicho sistema. Documentar exhaustivamente, y de forma autocontenida, el paquete desarrollado indicando sus posibilidades y limitaciones.

- Diseñar, teniendo en cuenta los diferentes enfoques de la Enseñanza Asistida por Ordenador (EAO), estilos distintos de asistentes: demostrativos, tutoriales, ejercicios y problemas, etc. Proponer actividades que permitan al docente evaluar y contrastar hasta qué punto las herramientas didácticas creadas favorecen o estimulan la comprensión de los conceptos fundamentales del Cálculo.

El uso de este tipo de herramientas nos permite entrar en la dinámica de provocar al estudiante, en el sentido de hacerle tomar parte activa en el proceso de aprendizaje. Por ejemplo, se puede retar al estudiante a intentar predecir el comportamiento de las sucesivas animaciones, o incluso llegar a intuir ciertas propiedades de los objetos estudiados. Sin duda, augurar tales resultados exige al estudiante ser protagonista del proceso de descubrimiento de los diversos objetos y requiere, de su parte, un esfuerzo de análisis, imprescindible para la comprensión de los mismos.

3.- DESCRIPCIÓN DE LA HERRAMIENTA

En esta comunicación presentamos los dos Trabajos Finales de Carrera, desarrollados hasta la fecha, que abarcan el estudio de series y el cálculo integral, respectivamente. Los módulos desarrollados son los siguientes:

a) Sucesiones y series

Sucesiones de números reales. Se han diseñado dos representaciones distintas para mostrar los términos de una sucesión $\{a_n\}_{n\in N}$ de números reales. En la primera de ellas, se representan sucesivamente cada uno de los términos mediante un punto sobre la recta real. Esta animación ilustra las nociones de convergencia y divergencia. La segunda de las representaciones, en la que se muestra la sucesión de puntos $\{(n,a_n)\}_{n\in N}$ en el plano real, puede sugerir la relación entre algunas sucesiones y funciones, viendo las primeras como una cierta "discretización" de las segundas (por ejemplo, la relación entre la sucesión $a_n=1/n$ y la función $f(x)=1/x$).

Series numéricas. La representación escogida pretende mostrar la relación entre los conceptos de sucesión de números reales y el de serie numérica. De este modo, la animación para una serie de términos positivos $\sum_{n\in N} a_n$ consta de tres partes: en la primera se muestran sucesivamente los puntos (n,a_n), en la segunda se visualiza el área acumulada de los diferentes rectángulos que tienen por base la unidad y por altura cada uno de los diferentes valores a_i, para $i \leq n$. De este modo, el valor de tal área corresponde a la n-ésima suma parcial, s_n. En la tercera parte de la representación, se muestran los términos (n,s_n) reforzando así la idea que $\{ s_n \}_{n\in N}$ es una nueva sucesión de números reales construida a partir de $\{a_n\}_{n\in N}$. Tal animación permite plantear al estudiante que intente imaginar cuál puede ser la relación, si hay alguna, entre la convergencia de ambas sucesiones.

Sucesiones de funciones. Muestra de forma animada las gráficas de las funciones reales de variable real que constituyen la sucesión $\{f_n(x)\}_{n\in N}$. De este modo, se puede hacer ver al estudiante que para cada valor fijo x_0, se obtiene una sucesión numérica como las mostradas anteriormente. Incluso, éste puede indicar para qué valores x_0, tales sucesiones numéricas van a ser convergentes. La función implementada, también se puede utilizar para ilustrar la diferencia entre convergencia puntual y convergencia uniforme de una sucesión de funciones definidas en un mismo intervalo.

Series de funciones. De forma parecida al caso de series numéricas, una serie de funciones $\sum_{n\in N} f_n(x)$, se muestra como la sucesión de funciones correspondientes a las sumas parciales, es decir, la sucesión $\{F_n(x)\}_{n\in N}$ donde $F_n(x) = f_1(x)+...+ f_n(x)$. Se puede hacer notar al estudiante que las series de potencias o las de Fourier no van a ser más que casos particulares de series funcionales.

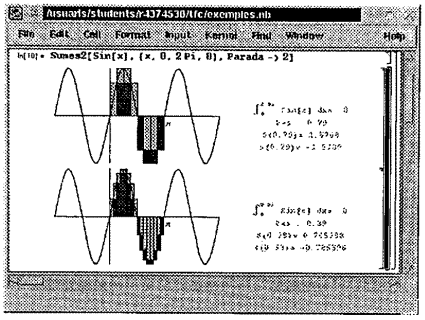
Serie de potencias. La animación de una serie de potencias $\sum_{n\in N} a_n (x-a)^n$ es similar a la de las series de funciones. Se puede proponer al estudiante que halle (intuya) el dominio de convergencia de dichas series o que encuentre sucesiones $\{a_n\}_{n\in N}$ tales que el radio de convergencia de la correspondiente serie de potencias tome determinados valores. Es importante que se den cuenta que para cada valor concreto de x que puedan ir tomando, lo que se obtiene es una serie numérica. Por lo tanto, algunas de las técnicas que van a utilizar para hallar la suma de algunas series de potencias pueden ser también utilizadas para el caso de ciertas series numéricas.

Serie de Taylor. La función implementada permite ver como las gráficas de los sucesivos polinomios interpoladores de Taylor de una función f, o sumas parciales de la serie de Taylor f en un punto a, se aproximan a la gráfica de f en el dominio de convergencia de dicha serie de potencias.

Serie de Fourier. La representación diseñada permite comparar la gráfica de una función periódica con las sucesivas sumas parciales de la correspondiente serie de Fourier, y ver como éstas van aproximándose a la función f en sus puntos de continuidad.

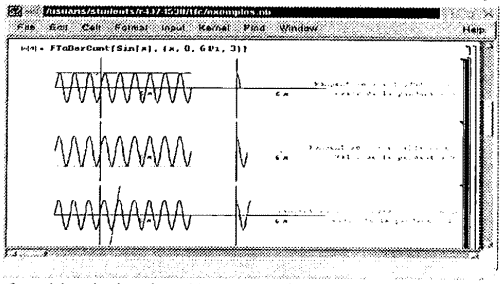
b) Cálculo Integral

Sumas de Riemann. Se pueden mostrar gráficamente las sumas inferiores y superiores de Riemann de una función f en un intervalo $[a,b]$, según se van afinando las particiones de dicho intervalo. El estudiante puede observar, por ejemplo, que si la función f es continua, entonces tales sumas van convergiendo a un mismo valor. A menudo, sin embargo, los estudiantes tienden a confundir tal valor con el del área delimitada por la gráfica de la función f, el eje de abscisas y los extremos del intervalo en que se trabaja. Se les puede proponer que hallen un método que permita calcular tal área utilizando los valores de ciertas integrales definidas.



Métodos elementales de integración numérica. Se muestra gráficamente como el método de los trapecios y el método de Simpson permiten aproximar el valor de la integral definida de una función f en un intervalo $[a,b]$, según se afina la partición que se va tomando de dicho intervalo.

Teorema fundamental del cálculo. En primer lugar, hemos intentado mostrar como se puede construir la primitiva $F(x)$ de una función continua $f(x)$ en $[a,b]$, definida como $F(x)=\int_a^x f(t)dt$, a partir del cálculo acumulado de integrales definidas en ciertos subintervalos de $[a,b]$. Posteriormente, la idea es hacer notar al estudiante que precisamente $F'(x)=f(x)$, $\forall\ x\in[a,b]$. Para ello se ha implementado una rutina que muestra la construcción de



la gráfica de la función derivada $g'(x)$ a partir del cálculo de la pendiente de la recta tangente a la gráfica de $g(x)$ en ciertos puntos.

Integración impropia. La función implementada permite visualizar una integral definida tal que en cada paso uno de los extremos del intervalo tiende a $+\infty$ ($-\infty$) o bien se va acercando al extremo donde la función presenta una asíntota vertical. También se pueden realizar

comparaciones de las integrales impropias de dos funciones, sobre un mismo intervalo, ayudando a intuir los criterios de comparación. A su vez, se puede sugerir al estudiante que estudie la convergencia de una integral del tipo $\int_1^\infty f(x)dx$ y la convergencia de la serie numérica $\sum_{n\geq 1} f(n)$, intentando intuir cierta relación entre ambas.

Es necesario mostrar al estudiante que estas ilustraciones no dejan de ser más que eso: representaciones que pueden ayudar a comprender de forma intuitiva ciertos comportamientos, pero que para el estudio de los mismos es imprescindible hacer un estudio analítico y riguroso. Para ello se pueden mostrar algunos ejemplos, en los que la representación visual pudiera resultar engañosa.

También cabe señalar que la herramienta diseñada es de carácter puramente ilustrativo, y que en ningún momento se ha pretendido implementar una herramienta general que permita trabajar con los casos más complejos o patológicos. Esto es debido al uso de ciertas funciones de algunos paquetes del *Mathematica* en la construcción de las primitivas, o incluso a las propias limitaciones de este *software*.

4.- ENSEÑANZA ASISTIDA POR ORDENADOR

Nuestro siguiente objetivo consiste en analizar las diferentes posibilidades que puede presentar la enseñanza de las matemáticas asistida por ordenador y, posteriormente, *dar forma* a las primitivas implementadas para elaborar un asistente para el aprendizaje.

Las razones que avalan el uso de ordenadores como un apoyo en el proceso de aprendizaje son múltiples y variadas:

- La inmediatez, potencia de cómputo y las posibilidades de animación que ofrece el *software* matemático existente, facilitan el diseño de una herramienta que permita al estudiante experimentar con las matemáticas, de forma interactiva.

- La motivación del estudiante acostumbra a ser mayor, tanto por el uso del ordenador en sí, como por la facilidades de interacción con el mismo, requiriendo por su parte una actitud más activa y una mayor atención.

- El estudiante puede realizar un trabajo más autónomo permitiendo, a su vez, ajustar el ritmo de aprendizaje al nivel de cada usuario. La herramienta puede estar incluso diseñada de modo que sea auto-correctiva, con lo cual cada estudiante pueda comprobar su nivel de aprendizaje y controlar sus progresos.

- El uso de una herramienta de este tipo puede facilitar al profesor información sobre el proceso de aprendizaje de cada estudiante, adquiriendo especial relevancia en sistemas de evaluación continuada.

Por otro lado, en el momento de diseñar una aplicación informática con finalidades didácticas, es necesario tener en cuenta las limitaciones o inconvenientes que se pueden presentar:

- El uso excesivo de un tal asistente puede llegar a ser aburrido e impersonal. Por otro lado,

un uso abusivo e inadecuado del mismo, podría provocar una pérdida de destreza básica y habilidades mentales del estudiante.

- Si los contenidos no están adecuadamente presentados o no se ajustan al nivel del estudiante, éste puede desviar su atención hacia el funcionamiento del *software* o ordenador, en lugar de centrarse en el problema matemático que se le plantee.

- Algunos estudiantes suelen tener una fe ciega en la máquina, con lo que es necesario provocar su espíritu crítico cuestionando ciertos resultados que se puedan obtener.

Todo esto conlleva a que el diseño de los contenidos de una tal herramienta no resulte nada sencillo y deba hacerse cuidadosamente. El enfoque de los diversos problemas a tratar debe realizarse de modo que el estudiante no los pueda resolver con tan solo ejecutar una instrucción, sino que requiera por su parte cierta destreza y análisis de la situación. Ésta es la tarea que pretendemos iniciar en la actualidad, después de la primera fase de implementación de las primitivas básicas y de un primer uso, a nuestro entender satisfactorio, como una herramienta de apoyo a la docencia en el aula.

REFERENCIAS

[AG91] Abellanas M. y García A., Eds. Actas de las Jornadas sobre Enseñanza Experimental de la Matemática en la Universidad. Universidad Politécnica de Madrid, 1991.
[GMM95] García A., Martínez A., Miñano.R; Nuevas tecnologías y enseñanza de las Matemáticas, Síntesis, 1995.
[Guz96] Guzmán M. El rincón de la pizarra. Ensayos de visualización en análisis matemático: elementos básicos del análisis, Pirámide, 1996.
[Hud86] Hudson K. Enseñanza asistida por ordenador. Guia práctica para escribir programas de EAO, Díaz de Santos, 1986.
[Mae91] Maeder R.E. Programming in Mathematica, Addison-Wesley, 1991.
[MB95] Montes A. y Brunat J.M., Eds. Actas de las Jornadas sobre Nuevas Tecnologías en la Enseñanza de las Matemáticas en la Universidad. *Universitat Politècnica de Catalunya*, 1995.
[Spi90] Spivak M. Cálculo infinitesimal, Reverté, 1990.
[Wol96] Wolfram S. The Mathematica Book, Wolfram Media, 1996.

ÁREA DE PROGRAMACIÓN

Responsable: Dr. Joaquín Ezpeleta Mateo
(Universidad de Zaragoza)
E-mail: ezpeleta@posta.unizar.es

UNA PROPUESTA PARA ASIGNATURAS DE INTRODUCCIÓN A LA PROGRAMACIÓN.

Aguilar, Jesús S. [1], González, Mariano[1], Riquelme, José C[1].

[1]*Departamento de Lenguajes y S.I.*
Facultad de Informática. U. de Sevilla
e-mail: {aguilar,mariano,riquelme}@lsi.us.es

RESUMEN: En este artículo se presentan las asignaturas de Introducción a la Programación que se imparten en las titulaciones de Ingeniero en Informática de la Universidad de Sevilla. Los rasgos más destacados son: introducción a la POO y la abstracción de datos, el uso de C y C++ en el laboratorio y una metodología basada en diapositivas, código para experimentos y el acceso a toda la documentación a través de la web.

1.- INTRODUCCIÓN.

Las asignaturas que inician al alumno en la programación durante el primer curso de las tres titulaciones en Ingeniería Informática se denominan Introducción a la Programación I e Introducción a la Programación II. Ambas son cuatrimestrales con 7.5 créditos, es decir, 5 horas de docencia a la semana y un temario común para la titulación de Ingeniero Superior e Ingenieros Técnicos. Estas 5 horas se reparten en 3 horas de clase en el aula convencional y que podríamos denominar clases de Teoría y 2 horas en el Laboratorio o clases de Prácticas. Estos laboratorios constan de una red local de ordenadores personales donde los alumnos trabajan en parejas. Desde este laboratorio los alumnos tienen acceso restringido a Internet, concretamente sólo a las páginas Web de la Facultad y de los departamentos con docencia en ésta.

El número de matriculados en ambas asignaturas durante el pasado curso 1999-2000 es superior a 1000 divididos en 8 grupos. Para el presente curso se prevén un total de 11 grupos, lo que se traduce en un importante esfuerzo en la coordinación y puesta en común por el profesorado, no sólo entre los distintos grupos sino en la imprescindible conexión teoría-laboratorio para que el alumno consolide en el laboratorio lo explicado en el aula.

Cada grupo tiene unos 125 alumnos que para las clases de Prácticas se dividen en dos grupos que reciben docencia con profesores distintos en aulas de Laboratorio diferentes. Durante el curso pasado en cada cuatrimestre había cuatro profesores impartiendo clases de Teoría y otros cuatro impartiendo prácticas.

Los contenidos de las asignaturas son los siguientes

1. Conceptos básicos	5. Secuencias
2. Tipos y expresiones	6. Tablas y registros
3. Notación Algorítmica	7. Ficheros
4. Funciones	

Tabla I. **Programa de IP1**

1. Recursividad	6. TAD Vector Dinámico
2. Tipos Abstractos de Datos	7. TAD Lista
3. Intr. a la Programación OO	8. TAD Pila
4. TAD Fecha	9. TAD Cola
5. TAD Cadena	10. TAD Árbol

Tabla II. **Programa de IP2**

Como se puede observar, el temario de IP1 es un temario clásico de primer curso de programación donde quizás lo único resaltable sea la utilización del lenguaje C en las clases de Laboratorio, en lugar de los más extendidos Modula o Pascal.

El temario de IP2 es menos "tradicional" para una asignatura de primer curso, en el sentido de que incorpora una introducción a la Programación Orientada a Objetos basada en la implementación de los tipos abstractos de datos clásicos (Lista, Pila, Cola, Árbol) mediante el uso de los conceptos de clases y tipos genéricos (*templates*) en C++.

2.- INTRODUCCIÓN A LA PROGRAMACIÓN I.

a) Clases de Teoría.

Los alumnos denominan clases de teoría a aquellas que se imparten en el aula convencional. Los profesores pretendemos tres líneas de actuación que se repiten en el mismo orden y de forma circular a lo largo del cuatrimestre, adaptándose a cada unidad específicamente:

- Presentación de un lenguaje algorítmico simple (que denominamos LEA), compuesto por un conjunto de tipos de datos básicos e instrucciones y estructuras de control elementales.
- Presentación del lenguaje de especificación formal que ayude a la comprensión de la interfaz de algoritmos mediante el concurso de precondiciones y poscondiciones (Pre/Pos).
- Presentación de los esquemas algorítmicos asociados.

Así para cada tema de los enumerados en la tabla I se presentan los tipos de datos con la signatura y las condiciones de definición para cada operación asociada a ellos. Posteriormente, se describe el lenguaje de especificación (operadores, dominios y cuantificadores) y se presentan las instrucciones, junto con sus propiedades, verificando su corrección. De esta forma el alumno dispone de un lenguaje imperativo completo y potente y, además, puede describir el comportamiento de cada algoritmo con las especificaciones Pre/Pos. A continuación se presentan los esquemas algorítmicos genéricos que se usarán para la implementación de los problemas. Estos esquemas (recorrido, búsqueda, filtro y ordenación) se comportan como

unidades independientes y su composición adecuada sirve para resolver los problemas al final de cada unidad.

Mediante esta metodología se ofrece un lenguaje consistente y no ambiguo, del cual los alumnos conocen el uso de una determinada operación asociada a un tipo de datos y la condiciones en las que puede aplicarse. De esta forma se esbozan subliminalmente los conceptos de abstracción de datos y encapsulamiento, conceptos fundamentales en la próxima IP2.

b) Clases de Laboratorio.

En las clases de laboratorio se imparten las prácticas, a razón de una por semana. Las clases se dividen en dos partes: durante la primera, que tiene una duración de entre 30 y 45 minutos, el profesor explica la traducción a lenguaje C de los esquemas algorítmicos estudiados en teoría correspondientes a la práctica de la semana. No se trata, pues, de estudiar el lenguaje C en su totalidad, sino solamente aquellas expresiones que son "traducción" directa del lenguaje LEA. Por ejemplo, se dedica poco tiempo a explicar los formatos de escritura de la función *printf*, no se estudian el operador ternario ? y los operadores += y similares, sólo se permite el uso del operador ++ detrás de una variable, etc.

1. Entorno de trabajo	7. Módulos
2. Introducción al lenguaje C	8. Tablas
3. Estructuras de control selectivas	9. Cadenas de caracteres
4. Estructuras de control repetitivas	10. Estructuras
5. Funciones (I)	11. Ficheros de texto
6. Funciones (II)	12. Ficheros binarios

Tabla III: listado de las prácticas de IP1

Para la impartición de esta docencia el profesor se apoya en una presentación realizada en Powerpoint que se proyecta en la pantalla del aula. Los alumnos pueden seguir la exposición en la pantalla o en sus propios ordenadores, descargando la presentación que se encuentra disponible en la página web de la asignatura. Esto les permite además preparar la clase con antelación y repasar los conceptos explicados a posteriori. La figura 1 muestra una diapositiva de ejemplo de una de las presentaciones.

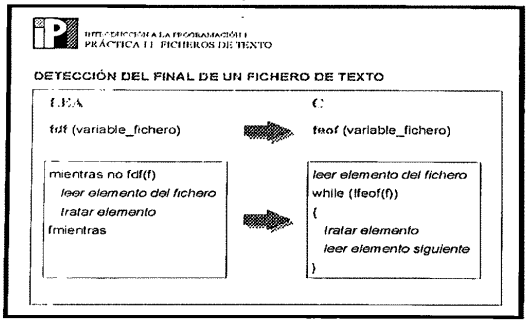


Figura 1: diapositiva de una presentación de las clases de laboratorio de IP1

La segunda parte de la clase de laboratorio se dedica a la realización de una serie de experimentos y ejercicios por parte de los alumnos. Los experimentos son ejercicios semirresueltos donde se plantea al alumno un problema concreto sobre alguno de los conceptos estudiados en la práctica. Si es preciso se proporciona al alumno el código del experimento para que se centre en el problema y no pierda el tiempo tecleando. Por su

parte, los ejercicios enuncian un problema que el alumno debe resolver en su totalidad, planteando en primer lugar el esquema algorítmico y traduciéndolo posteriormente a lenguaje C. Los alumnos cuentan en todo momento con la asistencia del profesor en el aula para resolver sus posibles dudas.

6.2 Complete los huecos del siguiente programa para que muestre en pantalla el mensaje

```
El código es 12 y el tipo es B

#include <stdio.h>
typedef int * Pint;
typedef char * Pchar;
void inicia (Pint, Pchar);

void main (void)
{
   int codigo;
   char tipo;
   inicia ( ___ codigo, ___ tipo);
   printf ("\nEl código es
       %d",codigo);
   printf (" y el tipo es
       %c",tipo);
}
void inicia (___ c,___ t)
{
   ____c = 12;
```

Figura 2: Ejemplo de experimento

La figura 2 muestra un ejemplo de un tipo de experimento consistente en completar el código de un programa que debe realizar una tarea concreta. Todos los experimentos y ejercicios que se realizan en el aula, junto con otros ejercicios adicionales y un resumen de los conceptos teóricos correspondientes, están disponibles en los enunciados de las prácticas, que se encuentran a disposición de los alumnos tanto en la copistería del centro como en la página web con una semana de antelación. Esto permite que los alumnos puedan preparar los esquemas algorítmicos necesarios previos a la resolución en C de los ejercicios, y de esta manera aprovechar mucho mejor el tiempo en el aula. Por otra parte, una vez impartida la práctica se publican las soluciones de los experimentos y ejercicios para que el alumno pueda compararlas con las propuestas por él, así como obtener las de aquellos ejercicios que no haya podido terminar en el aula.

c) Evaluación.

La evaluación de la asignatura se realiza, lógicamente, también por separado, con dos partes claramente diferenciadas. En el examen que podríamos denominar de Teoría se evalúa la utilización de los esquemas algorítmicos en la resolución de determinados problemas sobre ficheros y tablas principalmente. Los conocimientos impartidos en las clases de laboratorio se evalúan en una parte del examen escrito. Esta parte consiste en una batería de preguntas, que pueden ser incluso de tipo test, sobre la sintaxis del lenguaje C. De esta forma, en la evaluación de las prácticas sólo se valoran los conocimientos del lenguaje de programación, separándolos claramente de la habilidad para diseñar algoritmos o para utilizar esquemas algorítmicos.

La figura 3 muestra una muestra de un examen de prácticas de tipo test, extraído de la página web de la asignatura.

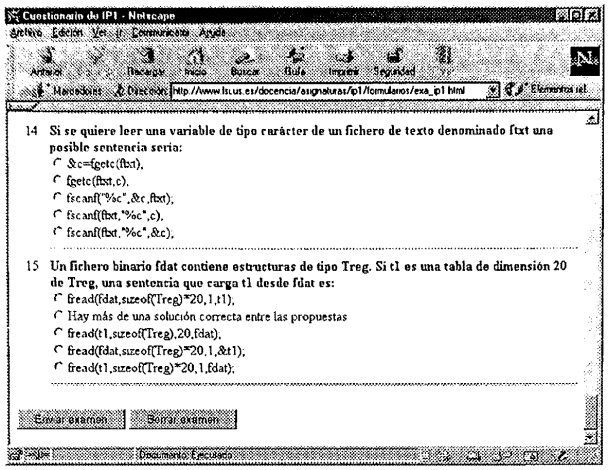


Figura 3: ejemplo de cuestionario sobre lenguaje C

3.- INTRODUCCIÓN A LA PROGRAMACIÓN II.

a) Clases de Teoría.

Las clases de Teoría de este cuatrimestre se dividen temporalmente en tres partes: el primer tema es un estudio clásico de los esquemas recursivos y la resolución de problemas clásicos: factorial, fibonacci, búsqueda binaria, torres de Hanoi, *quicksort*, etc. La segunda parte de la asignatura consiste en una introducción al concepto de tipo abstracto de datos y la programación orientada a objetos. De la POO se resaltan los aspectos de abstracción y encapsulamiento posponiendo la herencia y el polimorfismo para cursos posteriores. Con respecto a la abstracción se modelan los TAD Complejo y Fecha como ejemplos, para posteriormente incluir el concepto de genericidad de los TAD para que sirvan de plantilla. Así podemos utilizar Arbol de Lista de Cadena. El encapsulamiento permite que el alumno sólo utilice la interfaz de los métodos para resolver problemas. Así en las clases de teoría se desconoce el código de las operaciones del TAD, que se estudia en las clases de laboratorio. Por último, la mayor parte de la asignatura (75% del total) consiste en la introducción de los distintos TAD estudiados en la asignatura. Cada uno de ellos se introduce mediante una especificación de sus atributos y métodos en notación pre/pos, y en la realización de ejercicios basados en la interfaz proporcionada. Estos ejercicios se desarrollan en un lenguaje algorítmico similar a LEA pero donde aparecen objetos invocando a métodos de la Clase.

b) Clases de Laboratorio.

Básicamente, el desarrollo de las clases de prácticas es similar al de IP1. Es decir, el profesor explica la implementación que se ha realizado del TAD correspondiente y que se proporciona al alumno en la red local de forma completa o, normalmente, sin algunos de los métodos de la Clase. Posteriormente el alumno dispone de más de una hora para la implementación de una serie de experimentos, ejercicios semiresueltos y propuestos para su realización en los boletines correspondientes.

1. Entorno de Programación	7. TAD Lista
2. Recursividad	8. TAD Pila
3. TAD Fecha	9. TAD Cola
4. TAD Cadena	10. TAD Árbol
5. TAD Vector Dinámico	11. Repaso
6. TAD Polinomio	

Tabla IV: listado de las prácticas de IP2

La implementación de los distintos TAD se realiza mediante la definición de Clases en C++ con plantillas o *templates*, y de forma, que un TAD es utilizado en sucesivas prácticas . Así, por ejemplo, el TAD Fecha y el TAD Cadena se usan como atributos de un TAD Persona que se utiliza como base para el TAD Cola. Asimismo, las Clases Lista y Vector Dinámico se usan como bases para la implementación de las clases Pila y Cola.

En las sesiones de Laboratorio se hace sobre todo especial hincapié en los problemas de implementación, pues los esquemas algorítmicos son estudiados en las clases de Teoría. De esta manera, los experimentos se diseñan para que los alumnos comprueben la diferencia entre codificar un constructor copia o utilizar el que proporciona el compilador, pasar un parámetro por referencia o por valor, uso o no de la cláusula const en parámetros o métodos, etc. Los ejercicios semiresueltos suelen consistir en la implementación de algunos métodos del TAD correspondiente que han sido especificados en las clases de Teoría. Estos ejercicios se proporcionan en ficheros fuentes donde los alumnos sólo tienen que escribir algunas líneas de código para completarlo.

Por último, se proponen otros problemas que el alumno debe resolver previamente en LEA (la mayoría ya se han estudiado en las clases de Teoría) y posteriormente implementar en C++. Los comentarios a los experimentos propuestos y las soluciones de todos los ejercicios son publicados la semana siguiente. Si los alumnos lo desean pueden utilizar las sesiones de Laboratorio para discutir o corregir las propuestas de resolución de prácticas anteriores.

4.- UTILIZACIÓN DE NUEVAS TECNOLOGÍAS.

Las nuevas tecnologías ofrecen una serie de posibilidades que pueden mejorar muchos aspectos de la enseñanza y hacer más dinámica la comunicación entre el alumno y el profesor. Actualmente se están aprovechando algunas de estas posibilidades, como son:

- Uso de presentaciones multimedia en las clases de laboratorio para mostrar los conceptos de una forma didáctica y a la vez amena para los alumnos
- Potenciación de la página web para proporcionar información de interés a los alumnos:
  - Documentación: enunciados, presentaciones y soluciones de las prácticas, apuntes y problemas resueltos de teoría, enunciados y soluciones de exámenes de convocatorias anteriores
  - Calificaciones de los exámenes
  - Recursos de interés en la red
  - Cuestionarios de autoevaluación sobre las prácticas, mediante los cuales el alumno puede comprobar su nivel de conocimientos
- Acceso a tutorías por correo electrónico

Otras posibilidades que se piensan implantar en un futuro póximo son las siguientes:

- Mejora de las presentaciones para incluir más animaciones

- Realización de cuestionarios más flexibles mediante el uso de bases de datos y tecnología ASP
- Creación de una página de FAQs para consulta de los alumnos
- Creación de una lista de correo para distribuir rápidamente información de interés a los alumnos
- Publicación en la web de apuntes de teoría con simulaciones animadas de conceptos de interés o ejercicios resueltos

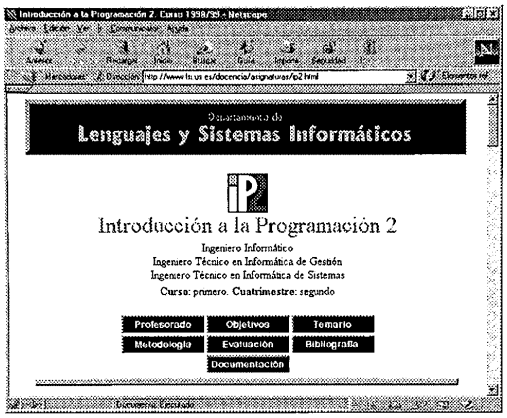


Figura 4: página web de la asignatura IP2

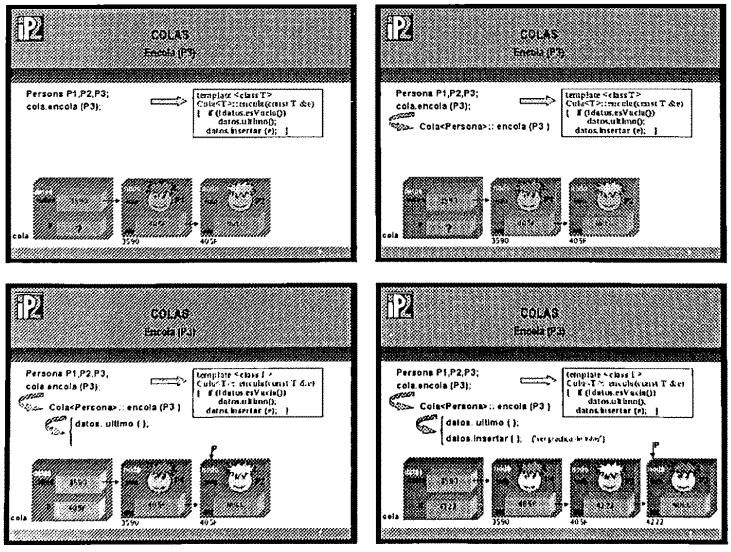


Figura 5: dispositivas de una presentación de las clases de laboratorio de IP1

UN MODELO GENERAL PARA ASIGNATURAS DE PROCESADORES DE LENGUAJES

Antonio Polo Márquez, Miryam Salas Sánchez

Departamento de Informática
Universidad de Extremadura
e-mail: **polo@unex.es**, **miryam@unex.es**

RESUMEN: Habitualmente el contenido de una asignatura de Procesadores de Lenguajes se centra en el estudio de lenguajes imperativos (tipo C o Pascal) y se utilizan herramientas basadas en reconocedores de estados como lex/yacc. En el presente artículo se propone un marco general que permita introducir, de forma natural e integradora, nuevos conceptos en el programa de esta asignatura. De una parte se propone el estudio de lenguajes de programación no imperativos, así como lenguajes de marcado (dentro de la familia de SGML/XML), tan habituales en el ámbito de Internet. De otra, abordar nuevas técnicas de procesamiento basadas en la manipulación directa de la representación intermedia del documento, como las que suelen emplearse en el procesamiento de los lenguajes de marcado. Esta propuesta se realiza mediante un diagrama de clases de los diferentes conceptos, analizando sus relaciones y aportando sugerencias para la elaboración de un programa real de la asignatura que se adapte a las necesidades de cada Centro.

1.- INTRODUCCIÓN.

En el Plan de Estudios de la Ingeniería de Informática que se imparte en la Escuela Politécnica de la Universidad de Extremadura, la asignatura de "*Procesadores de Lenguajes*" (PL) figura como una asignatura troncal, de carácter anual y con 9 créditos asignados (6 teóricos y 3 prácticos).

Cabe señalar que es la única asignatura obligatoria de carácter anual que deben cursar los alumnos de último curso de la Ingeniería, y que esta situación es similar en la mayoría de los Planes en otras Universidades.
Esta asignatura toma como base los conocimientos adquiridos por el alumno en la de "*Teoría de Autómatas y Lenguajes Formales*", que se imparte en 3º.
El contenido clásico de esta asignatura se ha centrado en el estudio de compiladores de lenguajes imperativos de alto nivel, tipo C ó Pascal, y el estudio de herramientas para su construcción. El núcleo de estas herramientas, denominadas de *generación de compiladores*, suele consistir en los reconocedores de lenguajes lex/yacc, los compiladores del lenguaje en el que está escrito el compilador (denominado *lenguaje de implementación*, y que suele ser C) y todo ello integrado mediante el uso de la herramienta *make* para automatizar el proceso de generación automática del compilador.

La guía central para este tipo de asignaturas es el libro clásico de Aho y Ullman [AHO86], que fue el fruto final de una sucesión de *biblias* de dichos autores sobre procesadores de lenguajes desde 1972.

Sin embargo, en los últimos años, todos los que impartimos esta asignatura, tenemos la sensación de que este esquema debe ser extendido para incluir numerosos conceptos que, de forma a veces vertiginosa, se están desarrollando en el campo de los procesadores de lenguajes.

De una parte, se presenta la necesidad de estudiar otros tipos de lenguajes. Así, en el ámbito de la programación, encontramos distintos paradigmas que proponen tipos de lenguajes con filosofías y necesidades de compilación bien diferentes a las de los lenguajes imperativos; como ocurre con lenguajes funcionales, lógicos y orientados a objetos. Y no sólo los lenguajes de programación, sino que lenguajes de propósito general para representación de información, como los lenguajes de marcado (por ejemplo HTML), se presentan con nuevos fines y necesidades.

De otra parte, las técnicas de procesamiento de estos lenguajes están cambiando profundamente. Por ejemplo, el uso de orientación a objetos para manipular directamente una representación del documento, como propone el Modelo de Objeto de Documento [DOM], o nuevas formas de especificar las reglas de transformación, como ocurre con XSL [XSL].

¿Por qué cuesta tanto introducir estos conceptos en los programas de nuestras asignaturas? Pensamos que no es sólo el vertiginoso cambio del mundo de los lenguajes de programación, o la falta de tiempo material para que los profesores puedan asimilar e integrarlos en la asignatura. Más importante nos parece la falta de un modelo general que aglutine el estudio de todos estos lenguajes y presente una metodología común para afrontar su estudio.

Creemos que, en la mayoría de las ocasiones, el profesor renuncia a realizar estos cambios por la imposibilidad de *añadir* más y más conceptos, en la mayoría de los casos absolutamente dispares. Resulta insuficiente el tiempo asignado y la complejidad de la asignatura se hace inmanejable para el alumno al tratar tantos temas inconexos.

En este artículo se propone, no el programa de la asignatura PL, sino un marco general de presentación de todos los conceptos esbozados anteriormente. Somos conscientes de que los contenidos son excesivamente amplios para abordarse en nuestra asignatura. Nuestro objetivo es aportar una visión unificadora de la asignatura de PL, extendiendo el contenido clásico, que permita al alumno asimilar de forma controlada nuevos conceptos y al profesor construir el programa concreto según las necesidades de cada Centro.

2.- MODELO GENERAL PARA EL ESTUDIO DE PROCESADORES DE LENGUAJES.

Para presentar el curriculum que proponemos para la enseñanza de PL, utilizaremos un modelo basado en objetos y expresaremos el conjunto de conceptos mediante un diagrama de clases, como se representa en la Figura 1.

La clase raíz representa una clase abstracta que indica el conjunto de conocimientos y herramientas necesarias para abordar el estudio de una determinada familia de lenguajes. El concepto de *familia de lenguajes* está determinado por el metalenguaje seleccionado en cada caso.

Esto nos lleva a considerar dos *implementaciones* concretas de esta clase abstracta:

a) La familia de lenguajes cuyas gramáticas se pueden expresar mediante notación EBNF. Y que, en nuestro caso, usaremos para expresar las gramáticas de los lenguajes de programación.
b) La familia de lenguajes cuyas gramáticas se pueden expresar mediante notación SGML/XML. Y que constituye la amplia familia de lenguajes de marcado habituales en el mundo de Internet, con su parafernalia de acrónimos correspondientes que a veces tanto desorientan.

Estas dos familias de lenguajes deben verse como *instancias* de un mismo molde (la clase abstracta raíz), y por tanto, deben soportar el mismo conjunto de propiedades y métodos generales para cualquier lenguaje. Bien es cierto que, en algunos puntos, existan diferencias en la *implementación* de algunas de estas características, como se indican en el resto de este artículo.
Los lenguajes concretos que aparecen como hojas en el árbol de la Figura 1, corresponden a objetos o instancias concretas de cada clase y coinciden con un procesador de un lenguaje fuente a una máquina virtual. Nótese que para los lenguajes de marcado no se especifica la máquina virtual, ya que tanto el lenguaje fuente como las reglas de su interpretación aparecen en la declaración del correspondiente espacio de nombres.

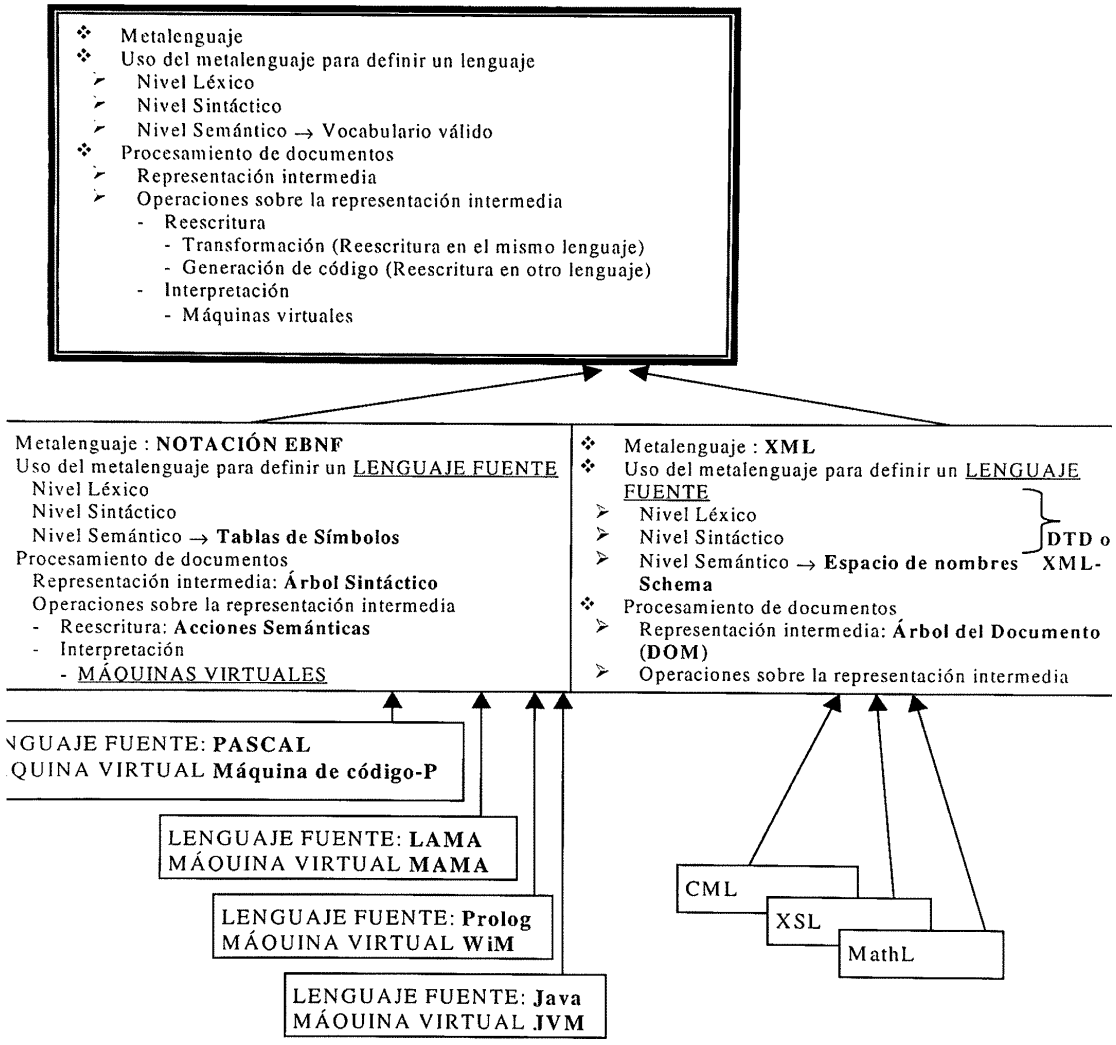


Figura 1. Diagrama de clases para el estudio de lenguajes

3.- PROCESADORES PARA LENGUAJES DE PROGRAMACIÓN.

Para abordar el estudio de los lenguajes de programación, se ha tomado como base el libro de Reinhard Wilhelm y Dieter Maurer [WIL95], que presenta un enfoque innovador en la línea de lo que proponemos en este artículo. La idea es definir siempre la misma estructura de presentación para cada tipo de lenguaje. Esto permite al alumno realizar un estudio comparativo, en los diferentes aspectos relevantes al procesamiento de lenguajes.
Para cada tipo de lenguaje de programación, se realiza el siguiente estudio:

a) Definición de un lenguaje fuente sencillo, pero representativo del paradigma de programación correspondiente. Se estudiarán sus niveles léxico, sintáctico y semántico.
b) Definición de una máquina virtual capaz de interpretar un código intermedio al que se traducirá el lenguaje fuente anterior.
c) Especificación de las funciones de compilación abstractas, que indican la forma de traducción de cada construcción del lenguaje fuente a código para la máquina virtual.
d) Finalmente, se realizarán los ejercicios o se darán las indicaciones necesarias para implementar las funciones de compilación anteriores utilizando herramientas del tipo lex/yacc.

Este esquema de estudio deberá adaptarse en cada caso, tanto al tipo de lenguaje concreto que se esté tratando, como también a la formación previa de cada alumno.

a) Lenguajes imperativos.

Se plantea la construcción de un compilador de un subconjunto de Pascal a código-P [WIL95] o de un subconjunto de C a código de una máquina de pila del tipo código-P [SCH85]. Como en nuestro caso el alumno tiene un conocimiento del lenguaje, apenas se tratan los aspectos léxicos, sintácticos y semánticos del lenguaje fuente. Pero se hace hincapié en el uso de las funciones de compilación y la máquina virtual. A nivel práctico se suele estudiar un compilador ya desarrollado para este propósito y se plantean diferentes cambios y mejoras en el mismo. Se destacan los mecanismos de implementación usando lex/yacc para construir el compilador.
En el futuro del curso, para el resto de lenguajes, ya se parte de estos conocimientos de las herramientas lex/yacc y se debe profundizar en otros aspectos como la definición y semántica del lenguaje fuente.

b) Lenguajes funcionales.

El estudio de un lenguaje funcional resulta, para la mayoría de nuestros alumnos, el primer contacto con este paradigma de programación, pues nuestro plan de estudios presenta escasas asignaturas que lo traten y siempre son de carácter optativo.
Siguiendo [WIL95], se puede proponer un lenguaje funcional muy sencillo, pero con las características principales como definición y aplicación de funciones, así como definiciones de expresiones recursivas mediante la construcción *letrec*. En este caso, los niveles léxico y sintáctico son muy simples, pero cuesta introducir al alumno en la filosofía funcional y que sean, no ya capaces de *interpretar mentalmente* cada programa para poder *leerlo*, sino que sepan capaces de *escribir* situaciones elementales en el lenguaje propuesto.
Otro concepto absolutamente novedoso para el alumno es la máquina virtual que se presenta, fundamentalmente en el uso intensivo de la memoria del *montón* (*heap*) para representar las clausuras y los mecanismos de evaluación de las mismas.
El estudio de este tipo de lenguajes se termina con una sencilla extensión del lenguaje para que admita el tratamiento de listas.

c) Lenguajes lógicos.

[WIL95] propone un subconjunto de Prolog y una máquina virtual, la máquina WiM, que es una simplificación de la máquina Warren [KOG91].
De nuevo, uno de los principales problemas es el caso de que el alumno desconozca la programación lógica. Esta situación requiere de nuevo, no sólo el estudio de las técnicas de procesamiento de estos lenguajes, sino todo un curso de programación en este paradigma.
Esto nos ha llevado en nuestro caso particular a la imposibilidad de incluirlo en el temario general y sólo se ha propuesto como extensión opcional en las prácticas, bien para aquellos alumnos que querían ampliar sus conocimientos, o bien para aquellos alumnos que ya habían tenido un contacto previo con este tipo de lenguajes al haber cursado las asignaturas optativas correspondientes.

El estudio de las máquinas virtuales para lenguajes no imperativos puede extenderse en [KOG91].

d) Lenguajes orientados a objetos.

Para los lenguajes orientados a objetos, proponemos el contenido indicado en [WIL95], pero apostamos reajustarlo para Java, en lugar de C++. La principal razón es presentar la máquina virtual de Java (JVM), e introducir conceptos como interpretación abstracta, validación de código, seguridad, portabilidad frente a eficiencia, etc.
En este caso, más que definir un nuevo lenguaje, se trata de extender los conceptos tratados para los lenguajes imperativos (Pascal o C) y analizar las soluciones para compilar características de orientación a objetos como clases y objetos, herencia, generalización, encapsulación, etc.
De nuevo, es necesaria una *cultura* del alumno en programación orientada a objetos para que pueda afrontar con éxito el estudio de las técnicas de compilación de estos lenguajes.
Un ejercicio interesante es la posibilidad de que el alumno acceda a la lectura del código fuente de los compiladores de estos lenguajes, fácilmente accesibles en la red. En ellos encontrarán *nuevas* formas de procesamiento de lenguajes, distintas de las herramientas lex/yacc.
Otra posibilidad es utilizar como lenguaje de implementación un lenguaje orientado a objetos, como Java en [APP98].

4.- PROCESADORES PARA LENGUAJES DE MARCADO.

La principal fuente para implementar los conceptos de este módulo se hayan en la página **http://www.w3.org** del World Wide Web Consortium [W3C], que debe ser consultada periódicamente para estar al tanto de las direcciones de desarrollo en XML.

a) Herramientas para el procesamiento de lenguajes de marcado.

El punto de partida del modelo de estudio que proponemos consiste en partir de XML[XML][1] como metalenguaje.

[1] Consideramos XML como una simplificación y mejora de SGML, pues aunque XML es una restricción de SGML, podemos decir que prácticamente todos los lenguajes de marcado utilizados se expresan mediante XML. Por ello nos limitaremos a la clase de lenguajes cuya gramática está definida mediante XML.

Se resalta la forma de expresar la gramática del lenguaje, y para ello, preferimos utilizar especificaciones más avanzadas al uso del DTD propuesto inicialmente en la especificación de XML, como son los XML-Schemas [XML-Sch].
En los niveles de definición de un lenguaje de marcado basado en XML, destacamos la simplicidad de los niveles léxico y sintáctico, así como el mecanismo de uso de espacio de nombres para extender la riqueza semántica del lenguaje.
Un punto importante son las operaciones de procesamiento de documentos. En primer lugar se resalta la simplicidad del modelo de objeto de documento [DOM], y el conjunto de métodos que permiten manipular el documento. Esta técnica de procesamiento es radicalmente distinta a la utilizada mediante lex/yacc en el procesamiento de lenguajes de programación.
A continuación, se aborda el estudio de XSL como un lenguaje que permite especificar las operaciones de reescritura. La potencia de XSL es suficiente para realizar las operaciones de transformación que puedan requerirse en el curso. Pensamos que los avances en las mejoras de XSL le convertirán en un lenguaje de manipulación de documentos lo suficientemente potente como para realizar todas las labores necesarias en el procesamiento de lenguajes de marcado.
Una característica a destacar es que el lenguaje XSL es declarativo, pero más próximo al estilo de los lenguajes lógicos (con reglas del tipo si ... entonces ...) que a las reglas de reescritura en las gramáticas definidas mediante notación EBNF.
La realización de prácticas en esta parte [MARC99] está fuertemente condicionada por los constantes cambios en las especificaciones.
Finalmente, se resalta las dos técnicas esenciales de interpretación de documentos XML:

I. Definiendo la semántica de un espacio de nombres. Que será equivalente a definir una máquina virtual.
II. Construyendo programas capaces de interpretar las instrucciones de procesamiento del documento.

Para la implementación de herramientas que puedan interpretar estos documentos suele ser necesario desarrollar programas en Java y librerías que implementen las especificaciones del modelo DOM [MARU99].

b) Ejemplos de lenguajes de marcado.

En el caso de los lenguajes de marcado la variedad de lenguajes disponibles es enorme. No sólo en cantidad, sino en la funcionalidad y objetivos de cada lenguaje propuesto.
Para afrontar este tema, nos parece que lo importante es que el alumno aprenda dos habilidades esenciales:

- ¿Cómo localizar lenguajes ya existentes en un dominio de aplicación concreto? Pues no debemos caer en la trampa de querer resolver cualquier problema que se nos presente definiendo nuestro propio lenguaje específico y empezar a construir las aplicaciones necesarias. Debemos de inculcar al alumno la necesidad de reflexionar previamente, y buscar soluciones existentes para no *reinventar la rueda* constantemente. Para ello se estudiarán lugares de encuentro en la red donde estén disponibles las definiciones de los espacios de nombres y herramientas disponibles para procesar los documentos del correspondiente lenguaje.
- ¿Cómo aplicar esos lenguajes para resolver un problema concreto? Pues la existencia de espacios de nombres y herramientas que resuelvan un dominio del problema no es útil si no se sabe integrar en el sistema concreto que estemos desarrollando. Se trata, por consiguiente, de saber definir nuestro lenguaje y las herramientas adecuadas para

procesar sus documentos, así como la interfaz de publicación/lectura de documentos con el exterior.

Para llevar a cabo estas acciones se proponen dos prácticas:

1. Acceso a lugares de registro de especificaciones XML [XMLorg], y examinar algunas de las propuestas que el profesor considere más representativas de la metodología a seguir para definir estos *dominios de conocimiento.*
2. Definir un lenguaje ML1, que sirva para realizar un marcado de alguno de los lenguajes desarrollados en el bloque de lenguajes de programación.

Esta última práctica es la culminación del curso y servirá para integrar y conectar todos los conocimientos adquiridos. La representación de un programa como un documento sugiere de forma inmediata varias ideas de fácil aplicación que el alumno puede realizar de forma práctica como:

- Dotar al programa de información adicional que pueda ser eliminada/recuperada para su documentación. Por ejemplo un mecanismo de generación automática de documentación como la que realiza *javadoc* en Java, incluyendo guardas, pseudocódigo, etc.

- Escritura de programas XSL que realicen la generación de código a la máquina virtual correspondiente o reescritura hacia otro lenguaje de alto nivel.

- Generar el código destino con estructura de marcado y facilitar operaciones de transformación para optimización del código o decompilación

5.- CONCLUSIONES Y TRABAJOS FUTUROS.

La propuesta de introducir nuevos conceptos en PL, readaptando los ya tradicionales de este tipo de asignaturas, se ha presentado mediante un diagrama de clases. En este diagrama, los conceptos para el estudio de diferentes lenguajes aparecen de forma jerárquica heredando de una misma clase raíz, que representa el modelo de estudio. Se han propuesto dos *implementaciones* de esa clase raíz: una para los lenguajes de programación y otra para los lenguajes de marcado.
La clase de los lenguajes de programación puede presentar diferentes instancias de objetos, que corresponden a lenguajes modelo de diferentes paradigmas de programación. Cada uno de ellos presenta una máquina virtual y las funciones de compilación hacia dicha máquina.

Para la clase de los lenguajes de marcado se resaltan los mecanismos de definición y las técnicas de manipulación de documentos a través del Modelo de Objeto de Documento [DOM] y el lenguaje XSL [XSL].
Creemos que esta visión de alto nivel del diseño de la asignatura PL puede ser de gran utilidad a los profesores implicados en estos temas.

Una de las ventajas de esta forma de modelado de contenidos es la facilidad para realizar adaptaciones a cada caso particular y sobre todo, proporciona una metodología para futuros diseños.
Actualmente, estamos trabajando un rediseño de la propuesta en el que figura como clase raíz la clase de documentos de marcado. Los lenguajes de programación se contemplarían en este

diseño como un caso particular de documento XML. Esta propuesta nos parece más lógica y supone un cambio más radical y atrevido en el planteamiento de esta asignatura. Sin embargo, nos parece que actualmente sería complicado llevarla a la práctica pues la mayoría de las herramientas para XML disponibles en el mercado (tanto a nivel conceptual como de software), se encuentran en proceso de febril desarrollo y por ello preferimos dar un poco de tiempo para que madure este campo.

ANEXO: Esquema De Programa Para Una Asignatura De Procesadores De Lenguajes

Objetivo General:

Partiendo de distintos modelos de comunicación, se analizarán los principales tipos de lenguajes que el hombre utiliza para expresar conceptos y cuya representación material es un documento. El curso se centra en el estudio de las técnicas de procesamiento de dichos documentos y cómo se pueden desarrollar herramientas informáticas que realicen o sirvan de ayuda en dicha tarea. Se distinguirán dos tipos especiales de documentos:

- Documentos de propósito específico, cuyo objetivo es representar conocimiento interpretable por dispositivos informáticos (máquinas virtuales), y que se corresponden con la categoría de los lenguajes de programación.
- Documentos genéricos, cuyo objetivo es simplemente representar información, combinando lenguajes multimedia (texto, imagen y sonido); y que se corresponden con la categoría de los denominados lenguajes de marcado.

Objetivos Específicos:

Teoría:

- Obtener un marco general de estudio de técnicas y herramientas para el procesamiento de lenguajes, que abarque tanto a los lenguajes de programación como a los lenguajes de marcado.
- Conocer y saber utilizar herramientas prácticas para el diseño léxico, sintáctico y semántico de los lenguajes de programación, así como para el proceso de reconocimiento, generación y optimización de código en la construcción de compiladores.
- Extender los conceptos anteriores a tipos de lenguajes distintos del imperativo, como son lenguajes funcionales, lógicos y orientados a objetos.
- Entender la importancia de la arquitectura de la máquina destino y la necesidad de definición del entorno de ejecución al que se va a traducir.
- Conocer y saber aplicar los principios de diseño e implementación de intérpretes como base para la definición de máquinas virtuales.
- Obtener la base suficiente para aplicar los principios básicos de la asignatura al procesamiento de información mediante su representación en documentos.

Prácticas:

- En la primera parte del curso se estudiará el diseño y realización de un compilador clásico sencillo para/con lenguajes imperativos. Para ello se seleccionará un subconjunto de un lenguaje imperativo conocido como Pascal ó C, proponiéndose distintas extensiones y modificaciones al mismo. El alumno se familiarizará con el uso de herramientas de generación de compiladores del tipo Lex/Yacc. Este lenguaje se denominará L1.
- En la segunda parte se aplicarán las técnicas anteriores a la construcción de compiladores para/con nuevos tipos de lenguajes: un lenguaje funcional, un lenguaje lógico y un lenguaje orientado a objetos.
- Finalmente deberá realizar la definición de un lenguaje de marcado (que denominaremos ML1) que almacene representaciones equivalentes a las del lenguaje L1 y deberá construir distintas herramientas de transformación de estos documentos. Una de ellas traducirá de ML1 a L1.

PROGRAMA DE CONTENIDOS

BLOQUE 1: MARCO GENERAL PARA EL PROCESAMIENTO DE LENGUAJES

TEMA 1: CONCEPTOS BÁSICOS

- Metalenguaje
- Uso del metalenguaje para definir un lenguaje
  - Nivel Léxico / Nivel Sintáctico
  - Nivel Semántico → Vocabulario válido
    - Ámbitos de validez / Homonimia, sinonimia y polisemia
- Procesamiento de documentos
  - Representación intermedia
  - Operaciones sobre la representación intermedia
    - Reescritura
      - Transformación (Reescritura en el mismo lenguaje)
      - Generación de código (Reescritura en otro lenguaje)
    - Interpretación
      - Máquinas virtuales
- Técnicas de construcción de procesadores de lenguajes

BLOQUE 2: LENGUAJES DE PROGRAMACIÓN

TEMA 2: HERRAMIENTAS PARA EL PROCESAMIENTO DE LENGUAJES DE PROGRAMACIÓN

- Metalenguaje: Notación EBNF
  - Nivel Léxico / Nivel Sintáctico
- Técnicas y herramientas para la construcción de procesadores de lenguajes de programación
- Nivel Semántico
  - Tablas de símbolos: Tipos / Ámbito de validez de una palabra
- Interpretación
  - Máquinas virtuales
- Generación de código
  - Organización de la memoria
- Optimización de código

TEMA 3: LENGUAJES IMPERATIVOS

- Lenguaje fuente: Subconjunto de Pascal
  - Nivel Léxico / Nivel Sintáctico
  - Nivel Semántico → Tabla de símbolos
- Máquina virtual: Máquina de código-P
- Funciones de compilación
  - Asignaciones y expresiones
  - Flujo de control
  - Asignación de memoria
  - Procedimientos
  - Programa principal
- Optimización

TEMA 4: LENGUAJES FUNCIONALES

- Introducción a los lenguajes de programación funcional
- Lenguaje fuente: LAMA
  - Nivel Léxico / Nivel Sintáctico
  - Nivel Semántico → Entornos y ligaduras
- Máquina virtual: MAMA
- Funciones de compilación
  - Expresiones programas
  - Expresiones simples
  - Compilación de ocurrencias de variables
  - Definiciones de funciones
  - Aplicación de funciones
  - Construcción y evaluación de clausuras
  - Expresiones *letrec* y variables locales
- Implementación de listas

TEMA 5: LENGUAJES DE PROGRAMACIÓN LÓGICA

- Introducción a los lenguajes de programación lógica
  - Fundamentos de lógica
  - Unificación
  - Ejecución de programas lógicos
- Lenguaje fuente: Prolog
  - Árbol de prueba
  - Entornos
- Máquina abstracta de Prolog: WiM
- Funciones de compilación
  - Objetivos
  - Términos cabeceras
  - Compilación de cláusulas
  - Retroceso (*backtracking*)
  - Procedimientos, programas y consultas
- Optimización

TEMA 6: LENGUAJES ORIENTADOS A OBJETOS

- Conceptos de lenguajes orientados a objetos
  - Objetos
  - Clases
  - Herencia
  - Generalización
  - Encapsulación
- Máquina virtual Java: JVM
  - Verificación de código e Interpretación abstracta
- Lenguaje fuente: JAVA
- Funciones de compilación
  - Compilación de métodos
  - Esquemas para una compilación de herencia
  - Generalización

BLOQUE 3: LENGUAJES DE MARCADO

TEMA 7: HERRAMIENTAS PARA EL PROCESAMIENTO DE LENGUAJES DE MARCADO

- Metalenguaje: SGML-XML
- Uso del metalenguaje para definir un lenguaje: DTDs y XML-Schemas
  - Nivel Léxico / Nivel Sintáctico
  - Nivel Semántico → Vocabulario válido
    - Ámbitos de validez: Espacios de nombres
    - Homonimia, sinonimia y polisemia
- Procesamiento de documentos
  - Representación intermedia: Modelo DOM
  - Operaciones sobre la representación intermedia
    - Reescritura: XSLT
      - Transformación (Reescritura en el mismo lenguaje)
      - Generación de código (Reescritura en otro lenguaje)
    - Interpretación
      - Técnicas de construcción de programas que implementen instrucciones de procesamiento.
      - Máquinas virtuales
- Técnicas de definición e implementación de Dominios de Conocimiento.

TEMA 8: EJEMPLOS DE LENGUAJES DE MARCADO

- Lugares comunes en la WEB para las definiciones de lenguajes de marcado.
- Ejemplos de aplicación

REFERENCIAS.

[AHO86] Aho, A.V.; Sethi, R.; y Ullman, J.D. *Compilers: Principles, techniques and tools*. Addison Wesley. 1986. (Existe traducción al castellàno).

[APP98] Appel, Andrew W. *Modern Compiler Implementation in Java/C*. Cambridge University Press. 1.998.

[DOM] http://www.w3.org/DOM/. Ver:

- Document Object Model (DOM) Requirements W3C Working Draft *12 April, 2000* (http://www.w3.org/TR/2000/WD-DOM-Requirements-20000412/)
- Document Object Model (DOM) Level 2 Specification Version 1.0 W3C Candidate Recommendation *10 May, 2000* (http://www.w3.org/TR/DOM-Level-2/)

[KOG91] Kogge, Peter M.. *The architecture of symbolic computers*, Mc Graw-Hill, 1.991.

[MARC99] Marchal, Benoit. *XML by example* Edt. QUE, 1999.

[MARU99] Maruyama, Hiroshi; Uramoto, Naohiko; Tamura, Kent. *XML and Java : developing Web applications*. Addison Wesley, 1999.

[SCH85] Schreiner, Axel T.; Friedman, George H. Jr. *Introduction to Compiler Construction with UNIX*. Prentice Hall, 1.985

[WIL95] Wilhelm, Reinhard; Maurer, Dieter. *Compiler Design*. Addison-Wesley, 1995.

[W3C] World Wide Web Consortium. http://www.w3.org

[XML] eXtensible Markup Language (XML) 1.0 W3C Recommendation 10-Feb-1998 (http://www.w3.org/TR/1998/REC-xml-19980210/). Ver http://www.w3.org/XML/.

[XMLorg] http://www.xml.org/xmlorg_registry/index.shtml

[XML-Sch] http://www.w3.org/XML/Schema.htmlVer:

- XML Schema Requirements W3C Note 15 February 1999 (http://www.w3.org/TR/NOTE-xml-schema-req/)
- XML Schema Part 0: Primer W3C Working Draft, 7-04-2000 (http://www.w3.org/TR/xmlschema-0/)
- XML Schema Part 2: Datatypes W3C Working Draft 7-4-2000 (http://www.w3.org/TR/xmlschema-2/)
- XML Schema Part 1: Structures W3C Working Draft 7-4-2000 (http://www.w3.org/TR/xmlschema-1)

[XSL] http://www.w3.org/Style/XSL/. Ver:

- Extensible Stylesheet Language (XSL) Version 1.0 W3C Working Draft 27 March 2000 (http://www.w3.org/TR/xsl/)

UNA HERRAMIENTA DE APOYO A LA EVALUACIÓN DE TRABAJOS PRÁCTICOS

José Luis Sánchez Romero[1], Daniel Ruiz Fernández[1]

[1]*Departamento de Tecnología Informática y Computación*
Universidad de Alicante
e-mail: {sanchez, druiz}@dtic.ua.es

RESUMEN: En el presente artículo, se presenta una herramienta que sirve como apoyo a la evaluación de diversos ejercicios prácticos que realizan los alumnos. Se describe el método seguido para la implementación del sistema desarrollado, junto a algunos ejemplos de manejo del mismo, el tipo de resultados que se obtienen y la ventajas derivadas de la utilización de la herramienta.

1.- INTRODUCCIÓN.

En el proceso de enseñanza y aprendizaje de la mayoría de las áreas de la Informática, la aplicación práctica por parte de los alumnos de los conocimientos teóricos adquiridos es de suma importancia. De hecho, en la mayor parte de los estudios universitarios de esta materia, el número de créditos de enseñanza práctica representa un elevado porcentaje de la carga lectiva total [1], [2], [3], [4]. Resulta obvio, por tanto, que la evaluación del trabajo práctico desarrollado por los alumnos representa un componente fundamental en el citado proceso, puesto que, por una parte, estos trabajos permiten al alumno desarrollar destrezas a partir de los fundamentos teóricos y afrontar una amplia casuística de problemas que puede aparecer en el desarrollo de un proyecto informático real; y por otra parte, esta evaluación supone para el docente una retroalimentación que le puede ayudar a modificar sus métodos, con el objetivo de conseguir una mayor calidad en la enseñanza [5], [6].

En muchas ocasiones, los trabajos prácticos que presentan los alumnos consisten en diversos programas que pueden corresponder a problemas puramente relacionados con el software, o bien a simulaciones de implementaciones hardware en relación con el área de la Arquitectura de Computadores.
El análisis de tales programas o códigos fuente constituye una tarea fundamental a la hora de evaluar estos trabajos prácticos realizados por los alumnos [7]. Aunque parte del proceso de desarrollo de estos programas se basa en la aplicación de unas estructuras sintácticas y de unas metodologías de programación preestablecidas, no hay que olvidar el hecho de que es igualmente importante la creatividad y la originalidad, las cuales ayudan a proporcionar distintas soluciones ante un mismo problema. Del posterior estudio de esta diversidad de soluciones, se podrá determinar cuáles presentan mayores ventajas y, en el caso de que alguna de ellas represente una innovación, intentar aplicarlas a otro tipo de problemas.
Resulta indiscutible por tanto que, aunque los alumnos deben compartir ideas y ayudarse mutuamente en la resolución de los problemas que se les plantea, el trabajo individual y personal es el que va a producir una solución definitiva original y va a aportar mayor calidad al proceso de enseñanza y aprendizaje. Dicho trabajo debe ser suficientemente valorado, tanto por el docente como por el propio alumno, quienes deben disponer de algún tipo de mecanismo para medir la originalidad y personalización de los programas entregados.

Con vistas a mejorar la calidad del proceso de enseñanza y aprendizaje y conseguir que los alumnos valoren cada vez más su trabajo personal y la originalidad de sus programas, se ha desarrollado la herramienta SEVAP (Sistema de apoyo a la EVAluación de Prácticas) que intenta establecer una clasificación dentro de un conjunto de trabajos prácticos presentados por los alumnos, proporcionando información sobre las estructuras sintácticas utilizadas, de modo que se ponga de manifiesto la mayor o menor creatividad de dichos trabajos.

2.- DESCRIPCIÓN DE LA HERRAMIENTA

Inicialmente, la herramienta SEVAP ha sido desarrollada para el sistema operativo Windows9x/2000®, aunque no se descarta la posibilidad de su adaptación a otros entornos. Del mismo modo, los programas fuente que la herramienta es capaz de analizar y clasificar deben haber sido desarrollados utilizando el lenguaje de programación C/C++ por ser éste uno de los lenguajes más ampliamente utilizandos en el ámbito docente universitario.

a) Método utilizado

El método utilizado para llevar a cabo la tarea de clasificación de los trabajos prácticos consiste, básicamente, en la realización de un análisis de *clustering* jerárquico [8], [9]; es decir, se trata de construir agrupaciones que, en cada paso de iteración, abarquen un mayor número de objetos analizados, hasta que cada uno de estos objetos quede caracterizado como perteneciente a uno u otro *cluster*. Este análisis puede resultar desglosado en las siguientes fases:

1. Determinación de las características o parámetros asociados a cada uno de los programas fuente. En el caso de la herramienta SICOP, los parámetros que se ha considerado pueden asociarse a cada programa fuente son el número de ocurrencias de los *tokens* siguientes: '*if*', '*while*', '*for*', '=' y ';' .

2. Normalización lineal de los distintos parámetros para cada uno de los programas fuente, mediante la formula

$$x_{ij}' = \frac{x_{ij} - \min_i x_{ij}}{\max_i x_{ij} - \min_i x_{ij}}$$

donde x_{ij} es el valor del parámetro j para el programa i, x_{ij}' es el valor normalizado del parámetro j para el programa i, y donde $\max_i x_{ij}$ y $\min_i x_{ij}$ son, respectivamente, el valor máximo y mínimo del parámetro j para el conjunto de programas analizados.

3. Realización del análisis de *clustering* jerárquico, de forma que se van seleccionando programas de forma individual, cada uno de los cuales pasa a formar parte de un *cluster* ya existente o bien constituye un nuevo *cluster*. Los pasos que se llevan a cabo para obtener las distintas agrupaciones de programas son los siguientes:

   3.1. Obtención de una matriz inicial de distancias entre los distintos programas analizados. Como medida de la distancia, se utiliza la distancia Euclídea, caracterizada por la fórmula

$$d(x_i, x_k) = \sqrt{\sum_j (x_{ij} - x_{kj})^2}$$

donde x_i y x_k son dos programas cualesquiera, y donde x_{ij} es el valor del parámetro j para el programa x_i.

3.2. Agrupación de los dos puntos más próximos que, inicialmente, formarán el primer *cluster*

3.3. Recálculo de las distancias entre los puntos que no pertenecen a ningún *cluster* y la nueva situación de los *clusters*. Hay que tener en cuenta que, en el caso del cálculo de la distancia de un punto a un *cluster*, no es directamente aplicable la fórmula de la distancia Euclídea, sino que se ha de utilizar la fórmula denominada *group average*, que proporciona la distancia entre un punto individual y un conjunto de puntos. Esta distancia viene determinada por la siguiente expresión:

$$d_{xi}(c_j, c_k) = \sqrt{\frac{|c_j|}{|c_j|+|c_k|} d^2(x_i, c_j) + \frac{|c_k|}{|c_j|+|c_k|} d^2(x_i, c_k)}$$

de tal modo que x_i representa un programa cualquiera, c_j y c_k son dos *clusters*, $|\cdot|$ representa el número de programas que conforman cada *cluster*, y $d^2(x_i, c_j)$ identifica el cuadrado de la distancia entre el programa x_i y el cluster c_j. Resulta obvio que, inicialmente, puede utilizarse esta fórmula para hallar la distancia de un punto a un *cluster* formado por dos puntos.

3.4. Selección de la menor distancia entre dos programas o entre un programa y un *cluster*. En el primer caso, los dos programas formarán un nuevo *cluster*; en el segundo caso, el programa pasará a formar parte del *cluster* ya existente.

3.5. Repetición del proceso anterior a partir del paso 3.3, hasta llegar al punto en que todos los programas han pasado a formar parte de algún *cluster*.

b) Ejemplo de utilización de la herramienta

El manejo de la herramienta se detalla a continuación: tras la puesta en ejecución de la misma, se ha de seleccionar la unidad de disco y el directorio que contiene los programas fuente que se desea clasificar, los cuales deben tener la extensión .c o .cc .

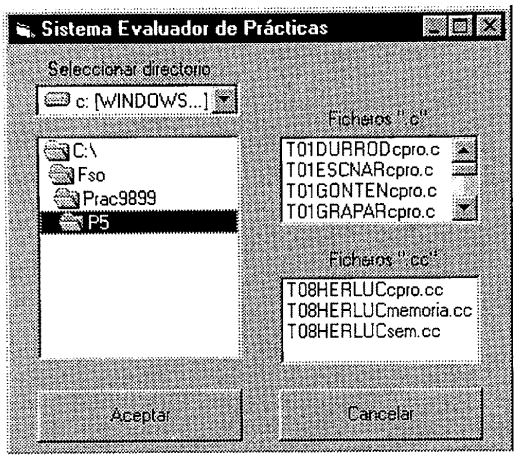


Figura 1. Interfaz del usuario de la herramienta SEVAP

Una vez finalizado el proceso, el usuario recibe un mensaje por pantalla en el que se le informa de que los resultados del análisis han quedado almacenados en el fichero clusters.txt, que se encontrará en el mismo directorio seleccionado en el paso anterior. Este fichero presentará un aspecto semejante al de la Figura 2.

Como se puede observar en la Figura 2, en el fichero de resultados clusters.txt aparecen los distintos *clusters* hallados junto con los programas fuente que forman parte de cada uno de ellos. Además, hay que señalar que, junto a cada programa aparecen, separados por dos puntos, los valores de los distintos parámetros que lo caracterizan, es decir, número de ocurrencias de los *tokens* '*if*', '*while*', '*for*', '=' y ';' , lo cual ayuda a localizar de forma más precisa aquellos programas fuente que, dentro de un *cluster*, presenten una mayor similitud. Así mismo, es preciso destacar que los programas que pertenecen a cada *cluster* aparecen ordenados por su antigüedad de pertenencia al mismo en orden decreciente, lo cual constituye un nuevo factor que se puede tener en cuenta a la hora de intentar localizar aquellos programas fuente más similares.

```
Cluster 1
=============
T10ESOMEcpro.c -->  24: 1: 1: 66: 240
T10SATSEMcpro.c -->  24: 1: 1: 66: 240

Cluster 2
=============
T10GARcpro.c -->  37: 1: 4: 89: 230
T10MARcpro.c -->  37: 1: 4: 89: 230

Cluster 3
=============
T01GRAPARcpro.c -->  42: 2: 2: 107: 257
T06BERPARcpro.c -->  41: 2: 2: 107: 261
T06CANFRUcpro.c -->  33: 2: 2: 103: 279
T08CALcpro.c -->  31: 2: 1: 109: 257
T10BERSORcpro.c -->  49: 1: 3: 118: 302
T10ESOMELcpro.c -->  36: 1: 2: 123: 300
T11MARZORcpro.c -->  47: 2: 1: 104: 229
```

Figura 2. Ejemplo de fichero de resultados de la herramienta SEVAP

Siguiendo con el ejemplo de la Figura 2, observamos que el *cluster* 1 está formado únicamente por dos programas fuente, T10ESOMEcpro.c y T10SATSEMcpro.c , lo cual indica que entre estos dos programas existe un grado de similitud mucho mayor que entre cualquiera de ellos y el resto de programas del conjunto total. Una circunstancia análoga ocurre con respecto al *cluster* 2. En cuanto al *cluster* 3, vemos que está formado por un conjunto mayor de programas, entre los cuales el grado de similitud va disminuyendo paulatinamente, es decir, los programas fuente T01GRAPARcpro.c y T06BERPARcpro.c serían los que presentarían, dentro de este *cluster*, un mayor parecido en cuanto a las estructuras sintácticas utilizadas.
A partir de la Figura 3, se observa que la situación que denotará un mejor grado de aprendizaje y una mayor originalidad en los trabajos realizados será aquella en la que no se produzcan grupos, es decir, el nivel inferior de la pirámide en el que cada grupo queda formado por un solo programa. De la misma forma la situación que habrá que intentar evitar será la que proporcione directamente el nivel superior del diagrama, es decir, un solo grupo que englobe todos los programas analizados

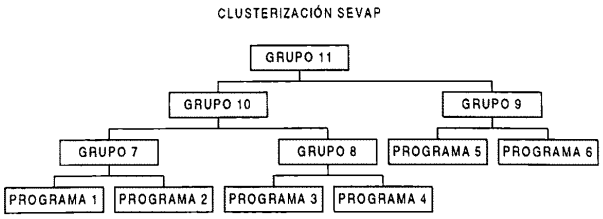


Figura 3. Diagrama representativo del proceso de *clustering* jerárquico

3.- CONCLUSIONES

Se ha desarrollado una herramienta automática que facilita la labor de evaluación de diversos trabajos prácticos realizados por el alumno, labor asumible tanto por el propio alumno como por el docente. Más concretamente, simplifica la evaluación de aquellos trabajos consistentes en uno o varios programas fuente desarrollado para plantear soluciones alternativas a distintos problemas. Esta herramienta realiza un análisis de *clustering* jerárquico sobre un conjunto de programas fuente, los cuales son preprocesados para extraer las características o parámetros representativos de cada uno de ellos. Se obtiene un fichero de resultados, del cual se puede deducir cuáles son los programas que aportan una solución con mayor grado de originalidad.
La herramienta funciona inicialmente bajo Windows 9x/2000® y analiza código fuente en C/C++; no se descarta su adaptación a otros entornos y lenguajes de programación. Del mismo modo, podrían escogerse otras características propias de los programas fuente que supongan un aumento del grado de representatividad.
La herramienta puede solicitarse a los autores, mediante el envío de un mensaje de correo electrónico o por cualquier otro medio alternativo.

4.- BIBLIOGRAFÍA

[1] "Directrices Generales Propias de los Planes de Estudio conducentes a la obtención de los títulos oficiales de Ingeniero en Informática, Ingeniero Técnico en Informática de Gestión e Ingeniero Técnico en Informática de Sistemas". Consejo de Universidades. Reales Decretos 1459, 1460 y 1461 de 26 de octubre de 1990. BOE nº 278 de 20 de noviembre de 1990, págs. 34401-34405

[2] "Plan de Estudios conducente al título de Ingeniero Técnico en Informática de Gestión de la Escuela Politécnica Superior de la Universidad de Alicante". Resolución de 18 de septiembre de 1992 de la Universidad de Alicante. BOE nº 18 de 21 de enero de 1993, págs. 1565-1578

[3] "Plan de Estudios conducente al título de Ingeniero en Informática de la Escuela Politécnica Superior de la Universidad de Alicante". Resolución de 18 de septiembre de 1992 de la Universidad de Alicante. BOE nº 37 de 12 de febrero de 1993, págs. 4467-4478

[4] "Plan de Estudios conducente al título de Ingeniero Técnico en Informática de Sistemas de la Escuela Politécnica Superior de la Universidad de Alicante". Resolución de 18 de septiembre de 1992 de la Universidad de Alicante. BOE nº 37 de 12 de febrero de 1993, págs. 4478-4489

[5] Medina Rivilla, A. et al. "Evaluación de los procesos y resultados del aprendizaje de los estudiantes". Universidad Nacional de Educación a Distancia, 1998

[6] Alonso Tapia, J. "Evaluación del conocimiento y su adquisición". Centro de Publicaciones del Ministerio de Educación, 1997

[7] Flórez Revuelta, F.; Fuster Guilló, A.; Grediaga Olivo, A.; Mora Pascual, J. M.; Soriano Payá, A. "Sistema Automático de Ayuda a la Evaluación Práctica Docente". Actas de las IV Jornades sobre l'Ensenyament Universitari de la Informàtica, págs. 273-277. Enginyeria i Arquitectura La Salle, Universitat Ramón Llull, 1998

[8] Ferrari, D.; Serazzi, G.; Zeigner, A. "Measurement and Tuning of Computer Systems". Prentice-Hall, 1983

[9] Hartigan, J. A. "Clustering Algorithms". Wiley, 1975

EXERCITA: UNA BASE DE DATOS DE MATERIAL PARA PRÁCTICAS DE PROGRAMACIÓN[§]

C. Gregorio Rodríguez[1], L. F. Llana Díaz[1], R. Martínez Unanue[1], P. Palao Gostanza[1], C. Pareja Flores[2] y J. Á. Velázquez Iturbide[3]

[1]*Facultad de Informática,*
Depto. Sistemas Informáticos y Programación, U. Complutense de Madrid
e-mail: cgregorio@sip.ucm.es, llana@sip.ucm.es, raquel@eucmos.sim.ucm.es, ecceso@eucmos.sim.ucm.es

[2]*Escuela Universitaria de Estadística,*
Depto. Sistemas Informáticos y Programación, U. Complutense de Madrid
e-mail: cpareja@sip.ucm.es

[3]*Escuela Superior de Ciencias Experimentales y Tecnología,*
Universidad Rey Juan Carlos de Madrid
e-mail: a.velazquez@escet.urjc.es

RESUMEN: Presentamos en este artículo el diseño y realización de una base de datos de materiales y herramientas para la práctica de la programación. El núcleo de dicha base de datos es el lenguaje *eXercita*, cuya descripción, implementación y uso se explican en estas páginas. Se expone igualmente la organización del contenido de la base de datos (principalmente ejercicios y prácticas), que da cuenta de sus posibilidades, flexibilidad y amplitud. Estas herramientas permiten distribuir el material en papel o en formato electrónico, ya sea mediante CD-Rom o a través de Internet.

1.- INTRODUCCIÓN: HISTORIA Y MOTIVACIÓN

Este trabajo nació como respuesta a una necesidad de varios profesores que han venido enseñando distintas asignaturas de programación en los últimos años. Dicha necesidad no es nueva (como recogen un buen número de referencias y depósitos de materiales; véase la bibliografía): consiste esencialmente en preparar y renovar materiales educativos, atractivos y originales; y la solución más simple era reciclar el material propio y el de los demás. El primer beneficio es tener más tiempo, ya sea para ampliar la colección o para otras de nuestras múltiples tareas. Además, tal compilación de recursos abundantes, frescos, variados y de calidad supone para el alumno un estímulo hacia una tarea tan bonita como es la programación.

[§]Este trabajo ha sido financiado por la Universidad Complutense de Madrid, mediante el proyecto de innovación educativa *Elaboración de material educativo para prácticas de programación*, ref. PIE 99/32, 2000.

Pero los múltiples colaboradores dispuestos a aportar material quisieran poder usar toda la colección a su medida: con distintos estilos de presentación, con la posibilidad de incluir o no pistas, todos o parte de los apartados, soluciones, ejemplos, juegos de datos, indicaciones de otros materiales en que se basan, lecturas complementarias, grado de dificultad previsto, etc. En resumen, la estructura de los ejercicios debía ser capaz de abarcar una casuística notablemente compleja debido a la diversidad de la información que puede interesar al plantear un ejercicio o una práctica.

Así pues, cada ejercicio individual se compone de múltiples piezas que se han de identificar para su procesamiento, y cada colección que se desea producir también. Para que toda esta información se pueda manejar de forma automática, ha habido que empezar por diseñar un lenguaje de marcado a la medida de estas necesidades y otras parecidas, y para que todo ello funcione, se han desarrollado compiladores y herramientas apropiadas.

Este artículo presenta el resumen de dicho lenguaje y herramientas. El manual de uso de eXercita [GLMPPV00] da cuenta de un sinfín de pormenores técnicos y de uso que no tienen cabida aquí.

En los siguientes apartados describimos el lenguaje de marcas usado (apartado 2), así como los compiladores de este lenguaje hacia PostScript y HTML, en que se presentará el material a los usuarios finales: alumnos y profesores (apartado 3). En el apartado 4 hablamos del contenido de dicha base de datos, y de la organización del mismo. Finalmente, en el apartado 5 resumimos nuestras conclusiones.

2.- ESPECIFICACIÓN: LENGUAJE Y CONCEPTOS

En este apartado describimos el lenguaje eXercita, que esencialmente es una notación para almacenar y posteriormente gestionar enunciados y soluciones de ejercicios, exámenes y prácticas.

El lenguaje de elaboración tipográfica es LaTeX; esto quiere decir que todas las piezas de texto dentro de un ejercicio se escriben con la notación de ese lenguaje. En cierto sentido, eXercita es sólo una capa de marcado que envuelve LaTeX, dando lugar a un lenguaje orientado hacia un dominio específico, que es el problema que nos atañe. En nuestra opinión, las alternativas más razonables son LaTeX, HTML y XML; el resto de los lenguajes no son legibles o están demasiado orientados a su generación y tratamiento automatizado (PDF, RTF), o no permiten manejar la estructura de los documentos, sino sólo su presentación. Hemos elegido LaTeX frente a HTML porque es más conciso, es más usual generarlo a mano, está orientado a la producción de documentos impresos en alta calidad y existen buenos traductores de LaTeX a HTML. En cuanto a XML, entra de lleno en nuestros planes.

La descripción del lenguaje eXercita en su totalidad excede ampliamente el espacio disponible en estas páginas. No obstante, enumeramos algunas de las marcas que lo definen para dar una idea de sus posibilidades expresivas:

- Parámetros de particularización: fecha (con distintos estilos), asignatura, curso, grupo, año académico, titulación, centro, institución, lenguaje de programación, etc.
- Piezas: título, propósito, resumen, texto, prólogo, enunciado, apartado, solución, pista, ejemplo, juego de datos de prueba, epílogo, procedencia, historia del problema, bibliografía, etc.
- Elementos diversos: para incluir logotipos, parámetros, referencias cruzadas a otros ejercicios, tratamientos condicionales diversos, etc.

- Requisitos de uso: dependencias entre ejercicios, estilos, formatos y paquetes de LaTeX, etc.
- Etiquetas de catalogación: modelo de programación, lenguaje, conceptos del lenguaje usados, grado de dificultad previsto, envergadura del ejercicio, etc.

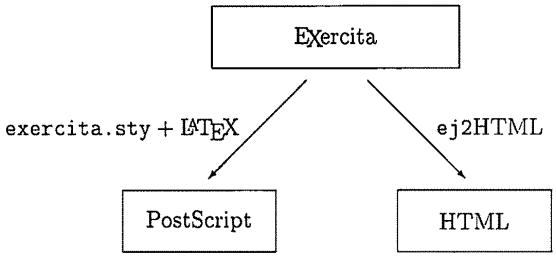
En el manual de uso de eXercita se puede consultar el detalle de todas las marcas mencionadas, así como su significado y manejo

.

3.- IMPLEMENTACIÓN: HERRAMIENTAS

En este apartado se describen dos de las herramientas que hemos construido para transformar ejercicios en algo legible: o un documento impreso en papel, o un lugar interactivo en la red. Estas herramientas cubren nichos completamente distintos, uno tradicional y otro muy moderno, pero ambos reales y hoyados en la práctica. También muestra dos extremos en la relación entre ejercicio y estudiante: la primera estática, decidida por los docentes, y la otra dinámica y elegible por quien estudia para ejercitar sus carencias o afinidades.

Aunque no es éste el momento de entrar en detalles técnicos, las herramientas desarrolladas operan como sigue



donde el paquete exercita.sty (escrito en TEX) permite empotrar eXercita en LaTeX y, por tanto, generar PostScript, mientras que ej2HTML (escrito en Perl) compila a HTML. En el manual de eXercita se explica detalladamente el uso de estas herramientas. Ahora, repasemos brevemente estos dos modos de publicación.

a) Publicación de ejercicios con LaTeX.

El uso más convencional de eXercita es generar un documento impreso. Lo más fácil habría sido construir un compilador que transformara código eXercita a código LaTeX, de donde se podría generar un documento en una segunda fase. Pero es mejor construir una serie de paquetes LaTeX que traduzcan directamente texto escrito en eXercita en un documento. De esta forma evitamos tener que hacer un procesamiento en dos o más pasos y se gana en portabilidad, porque TeX es igual en toda máquina.

El procesamiento de eXercita con LaTeX se divide en dos niveles:

- El nivel inferior está gobernado por un motor de *análisis de las marcas* de eXercita. Aunque este motor sólo analiza estas marcas, sin saber qué hacer con ellas, se puede configurar completamente, indicando qué marcas se han de capturar y qué debe hacerse con ellas.

Aunque se puede configurar esta capa con muy poco trabajo, exige unos conocimientos mínimos de programación con LaTeX (e incluso TeX) que no todo usuario posterior tendrá obviamente, además de ser necesario repetir ese esfuerzo para cada documento.

- Pero como, salvo en contadas ocasiones, siempre se quiere cargar las mismas partes y hacer lo mismo con ellas, sobre este nivel se pueden desarrollar paquetes que lo configuren de cualquiera de estas formas usuales. El superior puede ser ocupado por infinidad de *paquetes y clases* (estilos). Aunque actualmente sólo se ha desarrollado una clase (uniexer), ésta incluye parámetros que permiten configurarla para propósitos muy variados.

b) Publicación de ejercicios en Internet.

El lenguaje destinatario para la publicación en la red es HTML. También usamos JavaScript para hacer las presentaciones más cómodas, activas y comunicativas. El aspecto final es fácilmente modificable gracias a las hojas de estilo.

La herramienta concreta utilizada para generar un documento de red a partir de una colección de ejercicios es ej2HTML, que básicamente consiste en la composición secuencial de nuestra ej2TeX y la conocida LaTeX2HTML de Nikos Dragos.

La estructura del archivo HTML obtenido se basa en el encapsulamiento de la información en capas (*layers*) que son manejables con el lenguaje JavaScript y permite presentar la información procedente de la descripción de los ejercicios de un modo práctico y dinámico, mostrando la información menos relevante sólo si el usuario lo requiere.

Por supuesto, el aspecto del documento generado conserva toda la calidad que permite LaTeX, incluyendo imágenes de las fórmulas matemáticas, figuras, etc.

Junto con el documento final, la presentación final generada integra, en la misma ventana de navegación, el índice de ejercicios generado por ej2TeX y los códigos JavaScript necesarios para que todo funcione correctamente. Se ha incluido un apéndice con sendas figuras para mostrar el aspecto que presenta dicha ventana de navegación, así como una página en papel del documento generado.

4.- CONTENIDO DE LA BASE DE DATOS

a) Criterios de catalogación del contenido.

El contenido de la base de datos está estructurado físicamente por los temas argumentales sobre los que versan los ejercicios. No obstante, la organización física no constituye más que uno entre otros muchos criterios usuales de catalogación: por nivel de dificultad, por extensión, por mecanismos de programación requeridos en su resolución, por modelo(s) de programación recomendados, etc. El lenguaje de marcas de eXercita también contempla estas posibilidades de catalogación, para su posterior selección por los potenciales usuarios. Un par de ejemplos:

Puede interesar a un estudiante de programación en el paradigma imperativo repasar con ejercicios sencillos y no muy largos, sobre bucles.

Un profesor podría estar interesado en buscar enunciados para una práctica final, o desear preparar una hoja de problemas para practicar bucles con ejercicios sobre el bonito tema del número π y las múltiples formas de aproximarse al mismo.

En el manual de eXercita se detallan las posibilidades de catalogación consideradas por el momento.

b) Catalogación y mantenimiento colaborativos.

El planteamiento inicial de eXercita era y sigue siendo la compartición de recursos por distintas personas. Por consiguiente, la misma base de datos ha de permitir a distintas personas la aportación de material nuevo y el mantenimiento, posiblemente en paralelo, evitando las colisiones entre ellos. Este tipo de trabajo cooperativo ya se ha explorado en cientos de proyectos de *software* libre. Sus experiencias han revelado que CVS (*Concurrent Versions System*) [CVS] es una herramienta idónea. Actualmente, nuestro servidor de CVS sólo admite conexiones a los miembros de nuestro grupo, pero en un futuro permitiremos acceso de lectura público.

5.- TRABAJO FUTURO

Los objetivos planteados más a corto plazo para mejorar las herramientas y el material descritos con los siguientes:

- Preparar el sitio de Internet que facilite el uso libre de estos materiales. Dicho sitio pretende servir a la vez como punto de partida para otros recursos útiles a quien aprende o enseña. Este objetivo debe estar cumplido al principio del curso 2000-2001.
- Construir las herramientas necesarias para flexibilizar la obtención de los materiales de la base de datos. Concretamente, para seleccionar los recursos y el estilo deseados, mediante simple marcado durante su revisión por Internet.
- Difundir la base de datos, asegurando en lo posible la retroalimentación.
- En particular con nuestros alumnos, establecer criterios para medir de un modo fiable la eficacia educativa que tiene el uso de estos materiales.
- Definir en XML un equivalente a nuestras marcas de eXercita, y desarrollar un tercer traductor de eXercita a XML que lleve las marcas normales de LaTeX a marcas normales de HTML, y las marcas de eXercita a las nuevas marcas en XML.

BIBLIOGRAFÍA.

[B*99] J. Bergin y otros, *Resources for Next Generation Introductory CS Courses*, SIGCSE Bulletin, vol. 31, núm. 4, diciembre de 1999.

[CVS] Open Source Version Control Software, en la dirección **http://www.cvshome.org/**

[GLMPPV00] C. Gregorio Rodríguez, L. F. Llana Díaz, R. Martínez Unanue, P. Palao Gostanza, C. Pareja Flores y J. Á. Velázquez Iturbide, *EXercita: manual de uso*, Informe técnico SIP 102-00, Departamento de Sistemas Informáticos y Programación, mayo de 2000.

[K99] D. Knox, *On-line publication of CS laboratories*, SIGCSE Bulletin, vol. 29, núm. 1, marzo de 1997.

[K99] D. Knox, *The Computer Science Teaching Center*, disponible en **http://www.cstc.org/**, julio de 1999.

[K*00] S. Kovourov y otros, "SAIL: A system for generating, archiving and retrieving specialized assignments using LaTeX", Proceedings of the Thirty-First SIGCSE Technical Symposium on Computer Science Education, SIGCSE Bulletin, vol. 32, núm. 1, marzo de 2000.

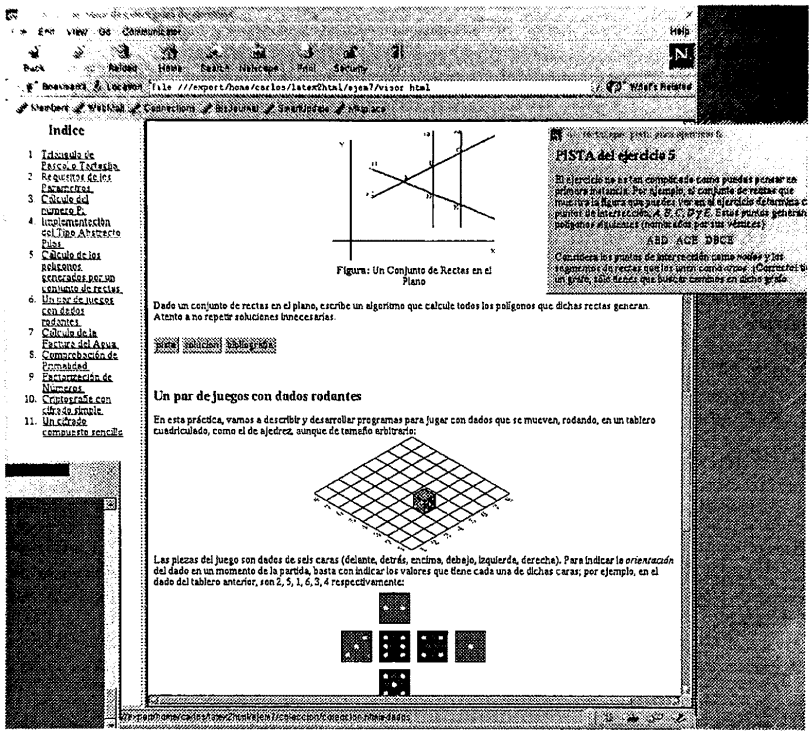
[JPPV00a] R. Jiménez Peris, C. Pareja Flores, M. Patiño Martínez y J. Á. Velázquez Iturbide, "New technologies in computer science education", en *Computer Science Education in the 21st Century*, Tony Greening (ed.), Springer-Verlag, 2000.

[M99] R. McCauley, *Computer Science Education Links*, disponible en **http://www.cacs.usl.edu/~mccauley/edlinks/**, julio de 1999.

[V96] J. Á. Velázquez Iturbide, "Estado del arte en enseñanza universitaria de la informática", *Novática*, 122, julio-agosto de 1996.

APÉNDICE

A) Aspecto de la ventana de navegación.



b) Un fragmento de un documento generado.

Año 1999/2000

Hoja de ejercicios

7 de junio de 2000

▷ 1. Un problema que a primera vista puede resultar muy difícil es el de calcular los polígonos que generan un conjunto de rectas en el plano. Considera el ejemplo de la figura 1. En dicho ejemplo puedes apreciar como cuatro rectas r_1, r_2, r_3 y r_4, han determinado cinco puntos de interseccion A, B, C, D y E. Los polígonos que pueden generar dichas rectas son aquellos que tienen por vértices las intersecciones y como lados segmentos incluidos en dichas rectas

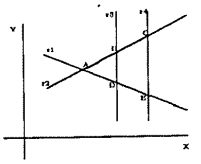
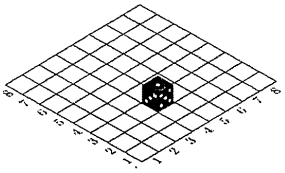


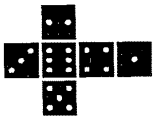
Figura 1: Un Conjunto de Rectas en el Plano

Dado un conjunto de rectas en el plano, escribe un algoritmo que calcule todos los polígonos que dichas rectas generan. Atento a no repetir soluciones innecesarias.

▷ 2. En esta práctica, vamos a describir y desarrollar programas para jugar con dados que se mueven, rodando, en un tablero cuadriculado, como el de ajedrez, aunque de tamaño arbitrario:



Las piezas del juego son dados de seis caras (delante, detrás, encima, debajo, izquierda, derecha). Para indicar la *orientación* del dado en un momento de la partida, basta con indicar los valores que tiene cada una de dichas caras; por ejemplo, en el dado del tablero anterior, son 2, 5, 1, 6, 3, 4 respectivamente:



Además, su *posición* en un tablero de $N \times N$ queda registrada en un par de números del conjunto $\{1 \ldots N\}$. Por ejemplo, el dado de la figura de arriba está en la posición (4,3). Se pide:

1

LA ENSEÑANZA DE JAVA EN LOS ESTUDIOS DE INFORMÁTICA

Elvira Albert[1], Javier Oliver[1], Germán Vidal[1]

[1] *Departamento de Sistemas Informáticos y Computación*
Universidad Politécnica de Valencia
e-mail: {ealbert,fjoliver,gvidal}@dsic.upv.es

RESUMEN: Java es un lenguaje de programación orientado a objetos, concurrente, robusto, portable y adecuado para el desarrollo de aplicaciones distribuidas. Pese a todas sus buenas características, cabe preguntarse si resulta adecuado incluir una asignatura en el plan de estudios de informática que contemple su enseñanza. En este artículo, analizamos esta cuestión y presentamos las características básicas de un curso centrado en la enseñanza de Java. Finalmente, describimos la experiencia en la implantación de una asignatura concreta que sigue las directrices expuestas.

1.- INTRODUCCIÓN.

Actualmente, el lenguaje de programación Java se encuentra en un proceso de continua evolución. La primera versión oficial data de Febrero de 1996. Desde entonces, el lenguaje se ha visto sujeto a un continuo cambio; fundamentalmente, se han ido añadiendo nuevas capacidades: conectividad a bases de datos, *JavaBeans*, RMI (*Remote Method Invocation*), *applets* firmadas, etc. Este dinamismo se ha visto reflejado en los temarios de los cursos de programación en Java, que se han ido actualizando conforme evolucionaba el lenguaje. Pese a todo, el núcleo del lenguaje se encuentra lo suficientemente maduro como para establecer una propuesta de contenidos para un curso de programación basado en Java.

En este artículo nos centramos en los estudios de primer ciclo en Informática. La primera cuestión que se plantea es si resulta adecuado plantear una asignatura en dichos estudios centrada en la enseñanza del lenguaje de programación Java, o si, por el contrario, esta materia es más propia de un curso externo al plan de estudios (por ejemplo, un curso de postgrado).

Por un lado, es indudable el impacto que Java ha conseguido en el ámbito industrial, donde se suelen requerir conocimientos de Java en los candidatos a puestos de trabajo relacionados con tareas de programación (y, especialmente, en áreas relacionadas con Internet). Por otro lado, el lenguaje Java puede también reportar a los alumnos una serie de conocimientos beneficiosos para cursar otras asignaturas del plan de estudios (como, por ejemplo, asignaturas relacionadas con la programación orientada a objetos, distribuida, concurrente o la asignatura *Proyecto Fin de Carrera*, en la que los alumnos a menudo muestran su interés hacia temas relacionados con la realidad empresarial y, por tanto, con el uso de un lenguaje como Java).

Cabe destacar también que, en los últimos años, se está llevando a cabo una remodelación de los contenidos de las asignaturas relacionadas directamente con la programación para incorporar, en muchos casos, las aportaciones que conlleva el uso de Java. Las Escuelas de Informática del

estado español se encuentran en estos momentos en un proceso de reforma de sus planes de estudio. Observando con detalle la información que sobre sus planes de estudio ponen accesible en Internet las diferentes universidades, detectamos que muy pocas han trasladado ya el uso de Java a los cursos básicos sobre programación. En algunos casos, es posible encontrar asignaturas (de carácter optativo, generalmente) centradas en la enseñanza de Java con los nombres más diversos: Programación Orientada a Objetos, Lenguajes de Programación o Fundamentos de Programación II. Con un objetivo distinto, cabe mencionar la experiencia llevada a cabo por la Universitat Politècnica de Catalunya sobre el uso de Java en una primera asignatura de "Introducción a la Programación". Como los profesores implicados en la misma describen en [FG99], el resultado ha sido ampliamente positivo.

Se podría argumentar que nos encontramos actualmente en un transitorio en los contenidos de las asignaturas de Programación en las Escuelas de Informática. La propuesta de asignatura que vamos a presentar se enmarca en ese intervalo de tiempo. Pensamos que en un futuro próximo no será de utilidad incluir una asignatura centrada en Java, porque los contenidos estarán ya desarrollados en diversas materias pero, hasta ese momento, la Universidad debe dar respuesta a la demanda que el mercado laboral exige.

2.- CARACTERÍSTICAS DE LA PROPUESTA.

A continuación presentamos las principales características de una asignatura de "programación en Java" enfocada a cursos avanzados de la enseñanza de programación. Como se ha discutido en la introducción, la materia tiene un carácter (implícitamente) transitorio. Por tanto, pensamos que el carácter más adecuado para la misma es que sea optativa.

Una primera cuestión es el tema de los *prerrequisitos* de la asignatura. Por un lado, sería deseable que los alumnos tuvieran unos conocimientos suficientes de programación (en cualquier paradigma, por ejemplo, en un lenguaje imperativo). Además, ya que Java es un lenguaje orientado a objetos, sería conveniente que tuvieran algunas nociones básicas sobre orientación a objetos (herencia, clases, objetos, etc). Sin embargo, pensamos que este último punto no es imprescindible, ya que nuestra experiencia con alumnos que no poseen dichos conocimientos ha resultado satisfactoria (dedicando relativamente poco esfuerzo a introducir los conceptos más relevantes). Por ello, pensamos que la ubicación más adecuada para la asignatura es el cuarto o quinto cuatrimestre de la titulación.

Como ya se ha comentado, la rápida evolución del lenguaje Java puede dificultar el diseño de contenidos para la asignatura. Asimismo, se trata de un lenguaje muy extenso, por lo que resultaría imposible contemplar su totalidad en el marco de una asignatura optativa típica (6 créditos). Teniendo en cuenta todos los factores apuntados, a continuación describimos las características básicas de nuestra propuesta.

a) Objetivos.

Los objetivos docentes de la asignatura son los siguientes:

- Introducir las nociones básicas asociadas al paradigma de la Programación Orientada a Objetos (POO).
- Desarrollar soluciones algorítmicas a problemas de dificultad media utilizando POO e implementarlas en el lenguaje Java.

- Aprender a construir aplicaciones distribuidas compuestas por código móvil que se ejecuta a través de una red.
- Proporcionar al alumno los conocimientos básicos sobre interfaces de usuario visuales con gestión de eventos.
- Presentar al alumno las características del lenguaje Java que permiten desarrollar aplicaciones relacionadas con Internet.

b) Contenidos Básicos.

Los siguientes contenidos son básicos y deberían contemplarse en el programa de una asignatura cuyo objetivo principal sea proporcionar al alumno los conocimientos suficientes para abordar el desarrollo de aplicaciones industriales en el lenguaje Java.

- *Introducción al paradigma de programación orientada a objetos y al lenguaje Java.* Aquí incluimos características esenciales de la POO, como los conceptos de *clase* y *objeto*, así como los mecanismos de *herencia*, *polimorfismo* y *genericidad*. Respecto al lenguaje Java, consideramos interesante introducir las principales diferencias con otros lenguajes de programación, el concepto de *applet* y su sintaxis básica.

- *Flujos de entrada y salida.* La entrada y salida en Java está basada en el uso de *flujos*. Su conocimiento resulta esencial para el manejo de ficheros, buffers de memoria e, incluso, la entrada/salida estándar. Los flujos son secuencias de bytes que se transfieren de una fuente a un destino a través de un canal de comunicación. La potencia de los flujos se debe a que abstraen los detalles de las operaciones de entrada y salida que se utilizarán en la comunicación.

- *Trabajo en red.* A menudo, entre las aplicaciones cliente/servidor que se encuentran en Internet, sólo un pequeño grupo es utilizado. Entre ellas, se incluyen: `ftp`, `telnet`, `e-mail`, `http`. En este bloque, se pretende que el alumno aprenda a desarrollar programas que implementen dichas aplicaciones cliente/servidor en red usando las facilidades del lenguaje Java.

- *Aplicaciones distribuidas.* Una aplicación distribuida es una aplicación cuyo procedimiento se distribuye por múltiples computadoras de una red. Las aplicaciones distribuidas pueden servir a la vez a múltiples usuarios y, dependiendo del diseño, conseguir un uso más optimizado de los recursos del sistema. En esta parte, se proporcionará al alumno la información necesaria para comprender las capacidades de programación distribuida a través de Java (mediante la RMI).

- *Programación WWW.* En este bloque se estudiará el acceso al WWW a través de Java, analizando las facilidades proporcionadas por los paquetes estándar del lenguaje. En particular, se debe introducir el desarrollo de *applets*, es decir, pequeños programas Java que se transmiten dinámicamente a través de la red como cualquier otro elemento de un documento HTML.

- *Interfaces de usuario.* El paquete `awt` es la aproximación que usa Java para la creación de interfaces de usuario independientes de la plataforma con la que se trabaja. Por lo tanto, resulta esencial para todo programador Java aprender cómo se organiza el `awt` y cubrir las clases que este paquete proporciona.

c) Prácticas de Laboratorio.

Las prácticas de laboratorio se deberían ajustar a los contenidos del apartado anterior, de forma que el alumno pueda llevar al terreno experimental los conceptos vistos en las sesiones de teoría.

Una cuestión relevante es el entorno utilizado para las sesiones de prácticas. El sistema que proponemos utilizar en el laboratorio para compilar programas Java es el JDK (Java Developer's Kit) cuya distribución es gratuita. El hecho de que Java sea gratuito resulta atractivo para las universidades que se enfrentan a presupuestos escasos. Es más, a medida que aparezcan nuevas versiones de Java, éstas estarán disponibles de inmediato a través de Internet, de modo que las universidades pueden mantener su *software* de Java actualizado. No obstante, se le puede ofrecer al alumno la posibilidad de utilizar un entorno de desarrollo más avanzado. Por ejemplo, el entorno de desarrollo Kawa aparece como una capa sobre el JDK. Por tanto, para poder utilizarlo, es necesario disponer del JDK. Entre las ventajas de Kawa, destacamos que puede trabajar con cualquier versión del JDK (por tanto, es fácilmente actualizable) y que tiene un precio asequible.

d) Bibliografía.

Existe una gran cantidad de material adecuado para el desarrollo de los contenidos de esta propuesta. De entre las distintas referencias, vamos a seleccionar algunas, que agrupamos en dos bloques:

- Las referencias [Joy98] y [Pet87] corresponden a textos introductorios al paradigma de programación orientada a objetos.

- El manual de Naughton [Nau96] ofrece una visión general del lenguaje Java, aunque las nuevas versiones del JDK han adoptado un enfoque diferente con respecto a ciertos temas (e.g., la entrada/salida). Una referencia más actualizada es [DD98]. Como manuales de referencia de Java, se recomiendan los textos de Jaworski [Jaw97,Jaw98]. Por último, citamos [AG97] y [Fla97] como material que ofrece una base sólida en estrategias y técnicas de programación en Java.

3.- UNA EXPERIENCIA CONCRETA.

Esta propuesta ha sido llevada a la práctica en la asignatura optativa PVI (Programación Visual para Internet) impartida a alumnos de tercer curso (primer cuatrimestre) de la Escuela Universitaria de Informática de la Universidad Politécnica de Valencia. El programa que se presenta ha sido elaborado para cubrir un total de 6 créditos que se distribuyen en dos partes: 3 para el desarrollo de teoría (y problemas en el aula) y los 3 restantes para prácticas de laboratorio.

El temario concreto de la asignatura PVI es el siguiente:

Tema 1. Introducción a Java

Introducción a la programación en Java. Sintaxis, tipos, operadores. Control de flujo.
Tema 2. Programación Orientada a Objetos
Clase / Objeto. Mecanismos fundamentales de la POO. Otras características.
Tema 3. Interfaces
Diferencias con clases abstractas. Variables en interfaces.
Tema 4. Paquetes
Restricción de la visibilidad. Protección de acceso.
Tema 5. Gestión de cadenas
Constructores. Sintaxis de cadenas. Comparación en cadenas. La clase StringBuffer.
Tema 6. Gestión de excepciones
Tipos de excepciones. Excepciones asíncronas. Excepciones creadas manualmente. Subclases de excepción propias.
Tema 7. Entrada / Salida
Archivos como flujos secuenciales de bytes. Entrada y Salida de archivos.
Tema 8. Trabajo en red
InetAddress. Conexiones mediante URLs. Puertos y conectores.
Tema 9. Conexiones a Bases de Datos
Estándar JDBC. Acceso mediante SQL.
Tema 10. Acceso a objetos remotos (RMI)
El modelo distribuido de Java. Implementación de la RMI.
Tema 11. Applets
HTML básico. Incorporación de datos en un applet. Imágenes y sonido.
Tema 12. El AWT
Componentes. Organización. Frames. Programación dirigida por eventos.

Las prácticas de la asignatura complementan los temas anteriores. Se trata de sesiones de prácticas dirigidas, de dos horas semanales, en las que los alumnos deben desarrollar pequeñas aplicaciones relacionadas con la materia que se esté estudiando en las sesiones de teoría.

Con respecto a la evaluación, ésta viene determinada por un único examen en el que se valoran los conocimientos adquiridos por el alumno. El examen incluye tanto preguntas acerca de los contenidos teóricos desarrollados a lo largo del curso, como ejercicios de carácter práctico enfocados a la implementación de programas en Java (implícitamente relacionados con el trabajo desarrollado en el laboratorio). Por lo que se refiere a resultados académicos durante el presente curso 1999/2000 (el primero en el que se ha implantado la asignatura PVI), cabe destacar que han sido muy satisfactorios. En la primera convocatoria se obtuvo un 72% de alumnos aprobados (de entre los presentados). El porcentaje ascendió ligeramente en la segunda convocatoria, llegando a un 84% de alumnos aprobados.

Para una información más detallada sobre la asignatura PVI, puede consultarse la URL:

http://www.dsic.upv.es/users/elp/ealbert/pvi.html

4.- CONCLUSIONES.

Como ya hemos argumentado, el conocimiento del lenguaje de programación Java es, en la actualidad, prácticamente imprescindible para los estudiantes de Informática. Esto se debe, principalmente, a su gran implantación y demanda en la industria (debido, sobre todo, al auge que están tomando Internet y las tecnologías relacionadas). Puesto que aún no se encuentra totalmente asentado en los planes de estudio, la existencia de una asignatura específica para su

desarrollo es, en estos momentos, vital. Desde nuestra experiencia en la impartición de la asignatura PVI, podemos concluir que una propuesta como la que hemos presentado tiene una gran aceptación por parte de los alumnos, que se encuentran muy motivados y eso ha llevado a que los resultados obtenidos hayan sido muy satisfactorios. Pensamos, como ya se ha dicho, que esta asignatura cubre un hueco de forma transitoria, pero necesaria, y que en un tiempo cercano estos contenidos serán posiblemente desarrollados en otras asignaturas a lo largo de la titulación. Hasta ese momento, una asignatura con estas características parece fundamental en cualquier plan de estudios, al menos con carácter optativo.

5.- BIBLIOGRAFÍA.

[AG97] Arnold, K. y Gosling, J. *El lenguaje de programación Java*. Addison-Wesley/DOMO, 1997.

[Anu96] Annuff, E. *Java Source Book: A complete guide to creating java applets for the Web*. John Wiley & Sons, Inc, 1996.

[DD98] Deitel, H. y Deitel, P. *Cómo programar en Java*. Prentice-Hall, Englewood Cliffs, NJ, 1998.

[FG99] Xavier Franch y Joaquim Gabarró. *Transición de Modula-2 a Java en un curso de iniciación a la programación*. JENUI'99, páginas 319-325.

[Fla97] Flanagan, H. *Java in a nutshell*. O'Reilly, 1997.

[Jaw97] Jaworski, J. *Java 1.1 Developer's Guide*. Prentice-Hall, 1997.

[Jaw98] Jaworski, J. *Java 1.2 al descubierto*. Prentice-Hall, 1998.

[Joy98] Joyanes, L. *Programación Orientada a Objetos*. McGraw-Hill, 1998.

[Nau96] Naughton, P. *The Java Handbook*. Osborne-McGraw-Hill, 1996.

[Pet87] Peterson, G. *Object-Oriented Computing. Volume 1: Concepts*. Computer Society Press of IEEE, 1987.

METODOLOGÍA Y HERRAMIENTAS PARA EL APRENDIZAJE DE LA VERIFICACIÓN Y DERIVACIÓN FORMAL DE PROGRAMAS

Inés Jacob[1]

[1]*I.J.(Universidad de Deusto)*
e-mail: ines@eside.deusto.es

RESUMEN: En este artículo se presenta la experiencia en la enseñanza de la asignatura Metodología de la Programación en la Facultad de Ingeniería de la Universidad de Deusto. Los contenidos de esta asignatura tratan la verificación y derivación formal de programas. Se han diseñado actividades en grupo en el aula, se utiliza una red interna como medio para compartir recursos y se cuenta con el correo electrónico como herramienta de comunicación asíncrona.

Se está trabajando además en la elaboración automática de listados de enunciados de problemas y ejercicios resueltos. Se basa en la utilización de la tecnología XML que permitirá, entre otras cosas, que los materiales obtenidos puedan ser accedidos vía web.

1. OBJETIVOS DE LA ASIGNATURA

El planteamiento de objetivos viene orientado por el análisis del lugar que la asignatura ocupa en el plan de estudios en Ingeniería Técnica en Informática de Gestión y en las áreas de conocimiento. Los objetivos elegidos persiguen:

- Cubrir el contenido fijado en el plan de estudios para esta asignatura troncal,
- Completar el itinerario del alumno por las asignaturas del área de Programación y Lenguajes,
- Fijar y afianzar las bases para el aprendizaje de contenidos de disciplinas relacionadas y abordadas en parte en cursos posteriores.
- Dividimos los objetivos, por su nivel de formulación, en generales y específicos. Los objetivos generales son:
- Familiarizarse con la verificación y derivación formal de programas
- Razonar formalmente el diseño recursivo de funciones
- Entender la recursividad y su relación con los programas iterativos

Los objetivos específicos se han elegido en función de los objetivos generales intentando que el alumno vea en ellos la identificación de los logros que ha de conseguir a lo largo de la asignatura. Coinciden además con los aspectos que serán evaluados, lo que facilita al estudiante la identificación de las áreas en las que debe detenerse y al profesor la calificación.

Se enumeran a continuación los objetivos específicos asociados a cada objetivo general (OG). Algunos objetivos pueden ser considerados medios (M) por ser necesarios para la consecución de otros considerados terminales (T).

OG1: Familiarizarse con la verificación y derivación formal de programas

- Especificar formalmente el comportamiento esperado de un programa (M)
- Verificar y derivar las instrucciones: nula, composición secuencial, asignación, alternativa, repetitiva y llamada a función. (M)
- Distinguir los conceptos implicados en la derivación y verificación de programas (M)
- Identificar la necesidad de realizar inmersiones (T)
- Diseñar llamadas iniciales a las funciones inmersoras (T)
- Mejorar la eficiencia de los programas (T)

OG2: Razonar formalmente el diseño recursivo de funciones

- Demostrar que los programas iterativos y recursivos terminan (M)
- Elegir casos base y recursivos (M)
- Derivar programas recursivos (T)
- Obtener recursividad final (T)

OG3: Entender la recursividad y su relación con los programas iterativos

- Pasar a postcondición constante (M)
- Transformar programas recursivos en iterativos (T)
- Derivar programas iterativos (T)

2. CARACTERÍSTICAS DE LA ASIGNATURA

Metodología de la Programación se imparte en la Universidad de Deusto en segundo de Ingeniería Técnica de Informática de Gestión. Se basa en los conocimientos de lógica formal y de programación que el alumno adquiere en primero. El alumno se supone que sabe programar, aunque de forma algo intuitiva, y comprueba si el programa obtenido es "correcto" haciéndolo funcionar (de forma simulada o real).
Los métodos de verificación y derivación formal se basan en el análisis estático de los programas. No se "ve" si los programas funcionan, sino que se "demuestra" matemáticamente que así es.
Algunas características de esta asignatura que la hacen diferente a otras de la misma área son:

- Los programas obtenidos no pueden probarse por ejecución (aunque podrían transformarse para que así fuera no es claro que sea conveniente hacerlo).
- Contradice en cierta forma el modo en el que los estudiantes han aprendido a programar en su primer año de carrera.
- Los métodos que se explican no se aplican habitualmente.
- Exige gran rigor, pulcritud, exactitud, claridad y orden.
- Los temas que componen el programa de la asignatura se apoyan unos sobre otros, de forma que es casi imprescindible que el alumno lleve la materia al día.

3. METODOLOGÍA Y MATERIALES

El diseño de la actividad docente se ve muy afectada por el tamaño de los grupos a los que se imparte la asignatura (más de 100 alumnos por grupo). Las diferentes actividades planteadas pretenden hacer posible la participación activa de los alumnos en el proceso de enseñanza-aprendizaje de forma que tanto el alumno como el profesor tengan ocasión de comprobar los avances realizados y ser conscientes de los errores que se cometen.

Se da especial importancia a potenciar el aprendizaje significativo de los contenidos evitando estudiar de memoria. Por ello se intenta reducir el tiempo dedicado a enunciar contenidos aumentando el dedicado a la experimentación.

a) Lecciones magistrales

En estas sesiones se trata de explicar los conceptos básicos en cada uno de los capítulos y exponer los procedimientos de aplicación de las diferentes técnicas. La explicación se ilustra con la realización de ejemplos y ejercicios. Para que haya cambios de ritmo que faciliten el mantenimiento de la atención del alumno se intenta que la exposición del profesor no ocupe toda la sesión. Algunos autores [3] recomiendan que su duración no supere nunca el 65% de la de la sesión completa. Normalmente esto se consigue solicitando la participación de los alumnos en la realización de ejercicios. Se logra además que

- el alumno ponga en práctica las técnicas aprendidas
- el alumno pueda autoevaluar su evolución
- la clase sea más participativa
- el profesor pueda observar el nivel de aprendizaje de los alumnos, no sólo en cuanto a resultado obtenido sino en cuanto a forma de proceder

En algunas ocasiones los ejercicios propuestos son los aparecidos en exámenes de cursos anteriores. Esto les permite por una parte conocer el tipn de examen con el que serán evaluados y por otra, autoevaluar su proceso de aprendizaje.

Con el fin de facilitar la motivación intrínseca del alumno se le proporcionan apuntes elaborados por el profesor en los que se recoge la explicación de los diferentes capítulos siguiendo la estructura de las clases magistrales impartidas. Los contenidos de estos apuntes pueden ser completados con el texto base de la asignatura y con el resto de la bibliografía recomendada al alumno. Al no tener que ocuparse de tomar apuntes pueden centrarse más en las explicaciones del profesor.

b) Grupos

Las actividades en grupo suponen una alternativa a la tradicional clase magistral. La experiencia con este tipo de actividades nos hace llegar a la conclusión de que el factor sorpresa/novedad es importante, aunque es difícil mantenerlo cuando la actividad se repite en más de un grupo. Esta repetición se hace recomendable por motivos de uniformidad en los diferentes grupos de un mismo curso y asignatura.

Otras ventajas de emplear esta actividad son:

- facilita la autoevaluación del alumno contrastada con otros compañeros
- puede ser un instrumento de evaluación continua a tener en cuenta: el número de trabajos a corregir sería notablemente inferior que con actividades individuales.

- permite al profesor observar a los alumnos en su forma de proceder
- facilita el diálogo entre el profesor y los alumnos.

Para esta asignatura se han diseñado tres actividades definiendo objetivo, materiales necesarios, composición de los grupos, sistema de puesta en común, organización y programación.

La primera actividad grupal pretende que los alumnos entiendan *la utilidad de conocer métodos de programación distintos a los aprendidos en cursos previos.* Sobre un programa sencillo en Pascal del que no se indica qué debe hacer, no tiene comentarios y además tiene un comportamiento erróneo, se plantean una serie de cuestiones a trabajar en grupo por los alumnos: ¿está bien el programa?; ¿qué habéis hecho para averiguarlo?, recoged todas las estrategias seguidas por los integrantes del grupo; si el programa no está bien indicad las razones y cómo lo modificaríais para mejorarlo; escribid el programa que obtenéis entre todos. En la puesta en común oral se recogen las aportaciones de los grupos a través del moderador de cada uno de ellos.

La segunda actividad grupal tiene como objetivo que los alumnos *practiquen la sistematización de lo aprendido y desarrollen la capacidad de extraer las ideas fundamentales.* Se distribuye entre los alumnos un documento con el enunciado para la derivación de una función recursiva y los pasos a dar para resolver el ejercicio. Se pide determinar en qué orden pueden darse estos pasos para derivar una función correcta. Puede haber varias soluciones válidas.

Por último, en la tercera actividad, se solicita a los alumnos la preparación de su propia autoevaluación. Consiste en diseñar un test con las preguntas que los alumnos propongan. Cada grupo elige 4 preguntas de test de las 16 aportadas por sus integrantes, asegurándose de que propone una respuesta correcta y dos incorrectas para cada una de las preguntas. Es importante que las respuestas incorrectas sean funcionales [5], es decir, sean elegidas por más del 5% de la muestra y tengan una correlación negativa con el total.

La puesta en común de esta actividad en grupo consiste en la realización de un test formado por las preguntas propuestas por los alumnos. Las preguntas acertadas suman 1 punto, las falladas restan medio punto y las que se dejan en blanco no tienen puntuación. Este sistema de puntuación se corresponde con el empleado en el examen final y trata de evitar que los alumnos intenten dar la respuesta correcta por adivinación [6].

Las actividades grupales realizadas en esta asignatura se plantean con fines de autoevaluación aunque se podría considerar la posibilidad de tomar en cuenta la puntuación obtenida por el alumno en cada una de ellas matizándola con el resultado del grupo al que pertenece.

c) Utilización de la red interna y el correo electrónico

Los alumnos son atendidos en tutorías presenciales. Estas tutorías están programadas tres horas a la semana. Pueden acudir sin cita previa. También pueden realizar consultas por correo electrónico lo que permite la comunicación asíncrona entre profesor y alumnos.

Este medio también se utiliza para recopilar materiales elaborados por los propios alumnos para ser compartidos con sus compañeros. Este tipo de materiales se ponen a disposición de los alumnos en la red interna de la Facultad.

El profesor deposita también materiales que considera de interés para los alumnos por servir de ayuda para el estudio personal o para trabajar en el aula.

GENERACIÓN AUTOMÁTICA DE EJERCICIOS

Una de las dificultades para el aprendizaje de la verificación y derivación formal de programas y, en concreto, de la asignatura Metodología de la Programación, es la escasez de ejercicios y problemas básicos. Los contenidos de la asignatura son fundamentalmente procedimentales y su aprendizaje se basa en la práctica.

Estamos trabajando en un proyecto para la extracción automática de enunciados de problemas y ejercicios resueltos contenidos en documentos en formato electrónico. Estos documentos recogen ejercicios completamente desarrollados en los que se han aplicado todas las técnicas y procedimientos que constituyen el contenido de la asignatura.

Este trabajo se basa en los resultados preliminares de un proyecto de fin de carrera de Ingeniería en Informática [1].

a) Planteamiento de objetivos

Partiendo de ejercicios completos dispuestos en documentos Word se extraen diferentes listados de ejercicios adecuados para la práctica de las técnicas estudiadas en la asignatura. Llamamos *ejercicio completo* al que partiendo de una especificación de función en lenguaje natural incorpora obtención de la especificación en lenguaje formal, solución recursiva del programa, inmersiones de eficiencia, transformación a recursividad final, obtención de verificación a postcondición constante, paso a iterativo y solución iterativa del programa.

Un programa así desarrollado, además de contener ejemplos de aplicación de estas técnicas, presenta ejemplos de derivación de instrucciones simples como asignaciones, composiciones secuenciales, llamadas a función, alternativas y repetitivas.

El tratamiento adecuado del documento que contiene este ejercicio completo, al que a partir de ahora llamaremos *documento fuente*, permitirá obtener para cada etapa del proceso de aprendizaje relaciones de ejemplos y propuestas de ejercicios. La extracción manual de estos ejemplos y ejercicios resueltos es muy costosa por lo que este proyecto pretende automatizar el proceso.

b) Fases principales del proceso automatizado

La metodología contempla tres fases de procesamiento: (a) transformación del documento fuente en texto puro; (b) etiquetado en XML del texto puro; y (c) diseño e implementación de filtros en XSL.

Transformación del documento fuente en texto puro. La edición de los ejemplos completos se realizó en Word por comodidad del profesor que los preparó. La nomenclatura utilizada se corresponde con el pseudocódigo propuesto por Balcázar [2]. En esta primera fase se transforma el documento fuente en texto puro, tratando adecuadamente los símbolos especiales ($\forall$, $\exists$, α, $\geq$, $\rightarrow$, etc.) para que puedan seguir siendo reconocidos.

Etiquetado del documento en XML. Optamos por el lenguaje XML (Extensible Markup Language) [4] que permite diseñar un conjunto de etiquetas adecuado a la estructura del documento tratado. Además será posible presentar en páginas web la información obtenida. El repertorio de etiquetas permitidas y la estructura del documento quedan recogidas en la DTD (Document Type Definition). La DTD presenta la gramática del pseudocódigo empleado en la

codificación de los programas, lo que permite validar su sintaxis. En esta fase se comprueba que el documento se adapta a la gramática del lenguaje y se etiquetan todos sus componentes.

Diseño e implementación de filtros en XSL. Una vez que el documento ha sido convenientemente etiquetado queda explotarlo de acuerdo a las necesidades de los alumnos. Esto se hace mediante la aplicación de filtros implementados en XSL (Extensible Stylesheet Language). Este lenguaje permite crear hojas de estilo que ofrecen visiones diferentes de un documento XML. Se desarrolla por lo tanto una XSL por cada uno de los diferentes listados a obtener: de asignaciones, de composiciones, secuenciales, etc. Cada uno de estos listados podría obtenerse en dos versiones: enunciados y ejercicios resueltos. En el caso de utilizar navegadores que no permitan la visualización de documentos XML se podría diseñar e implementar un filtro adicional que obtuviese documentos en HTML.

5. CONCLUSIONES

Hemos presentado en este artículo las acciones de mejora para una asignatura del plan de estudios conducente a la obtención del título de Ingeniero Técnico en Informática de Gestión. Se trata de un proceso continuo que pretende mejorar el aprendizaje de los alumnos mediante la incorporación de metodologías variadas y la utilización de tecnologías actuales.

La metodología diseñada para obtener automáticamente listados de ejercicios puede aplicarse a asignaturas en las que las técnicas básicas aprendidas se aplican ejercicios completos siempre y cuando la resolución de los ejercicios está muy estructurada de forma que puede definirse su gramática.

El material didáctico obtenido puede ser visualizado por navegadores de Internet con capacidad para tratar documentos XML mediante hojas de estilos XSL (como MSExplorer5.x y Netescape6.x).

El profesor simplemente deberá desarrollar ejercicios completos dejando que la elaboración de propuestas de ejercicios y ejemplos adecuados a cada etapa del proceso de aprendizaje se automatice.

BIBLIOGRAFÍA

[1] I. Azofra y C. de Prado, 1999. *Etiquetado automático de documentos para la extracción de información.* Proyecto fin de carrera. ESIDE, Universidad de Deusto.
[2] J.L. Balcázar, 1993. *Programación metódica.* McGraw-Hill.
[3] Dide, 1999. *La exposición como técnica didáctica.* Dirección de Investigación y Desarrollo Educativo de la Vicerrectoría Académica del Instituto Tecnológico y de Estudios Superiores de Monterrey. **http://www.sistema.itesm.mx/va/dide/inf-doc/estrategias/exposicion.html**
[4] C. Goldfarb y P. Prescod, 1999. *Manual de XML.* Prentice-Hall.
[5] T.M. Haladyna y S.M. Downing, 1988. *Functional Distractors: Implications for Test-Item and Test Design,* ponencia presentada en el congreso anual de la American Educational Research Association, New Orleans.
[6] P. Morales, 1995. *Las pruebas objetivas.* Cuadernos monográficos del ICE, Universidad de Deusto, Bilbao.

UNA HERRAMIENTA DIDÁCTICA PARA LA GENERACIÓN Y UTILIZACIÓN DE ANALIZADORES LR DIRECTOS E INVERSOS

José Fortes Gálvez

Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
E-mail: jfortes@dis.ulpgc.es

Resumen: Se presenta una herramienta didáctica desarrollada por el autor, que facilita al alumno trabajar con varios métodos de análisis sintáctico de las familiar LR directo e inverso. Es posible además generar de manera automática sentencias aleatorias no triviales.

1.- INTRODUCCIÓN.

En asignaturas donde se introduce al alumno en lenguajes formales, y en particular en el análisis sintáctico, se suele poner énfasis en el análisis LR, por razones evidentes de potencia de análisis (familia de gramáticas aceptadas) y de utilidad práctica (disponibilidad de herramientas como yacc). Desafortunadamente, los métodos de dicha familia, y en general, la mayoría de los métodos de análisis sintáctico, con la prácticamente única excepción del descenso recursivo, presentan el inconveniente de ser muy laboriosos de desarrollar a mano. Si no se dispone de una buena herramienta, el alumno puede perder mucho tiempo en desarrollar un ejemplo, muchas veces obteniendo un resultado erróneo, por lo minucioso y prolijo de la construcción, lo que induce a frustración y rechazo. En todo caso es impensable construir a mano dichos generadores para gramáticas de un cierto tamaño real. Y, por último, tal dificultad impide al alumno experimentar con sus diseños y analizar las consecuencias del mismo.

El autor ha desarrollado, durante su trabajo de tesis doctoral en el que presenta un enfoque nuevo al análisis de potencia LR, denominado discriminante inverso (siglas DR en inglés), una herramienta que permite:

1. Generar analizadores LR(0), SLR(1), LALR(1), LR(1) y otros de la familia DR. Se pueden visualizar los items, acciones, y tabla de análisis, tanto en formato texto como en formato LaTeX para ser incluidos en la documentación del trabajo.

2. Generar de forma automática sentencias para gramáticas, de forma aleatoria no-trivial.

3. Seguir paso a paso el análisis sintáctico de una entrada (escrita manualmente o generada automáticamente), en cualquiera de los métodos anteriores.

La herramienta ha sido desarrollada en Pascal estándar (con opciones de compilación para varios compiladores de amplio uso) y está disponible libremente (licencia GPL).

En este artículo presentamos brevemente las posibilidades de la herramienta, así como una sesión demostrativa del uso de la misma.

Es de destacar que, además de en nuestra universidad, dicha herramienta se usa desde hace algunos años en los estudios de informática de la Universidad de Niza Sophia Antipolis (Francia), y está siendo objeto de reescritura en Java con el objeto de mejorar la accesibilidad y el interface con el usuario.

Por otro lado, se está reimplementando en C++ de forma eficiente para ser presentada como alternativa para la construcción realista de analizadores de gramáticas de lenguajes de programación, donde además se incorporarán extensiones no canónicas al método discriminante inverso que permitan analizar gramáticas no-LR.

2.- UTILIZACIÓN DE LA HERRAMIENTA

La herramienta se ejecuta desde el prompt del sistema operativo, con el nombre de aut, seguido de una opción que indica el modo de funcionamiento:

- aut gr Genera analizador *discriminante inverso*.
- aut gd Genera analizador de la familia LR directa (Knuth).
- aut s *L* Genera una o más sentencias aleatorias de longitud *L*, si ésta no se especifica, la longitud es asimismo aleatoria.
- aut pr Analiza una secuencia de entrada utilizando el método discriminante inverso.
- aut gr Igualmente para la familia LR directa.

Normalmente se invoca en primer lugar un generador (aut gd), luego se utiliza opcionalmente el generador de sentencias, y por último, se invoca el analizador correspondiente.

a) Ficheros Utilizados por la Herramienta.

La herramienta utiliza los tres ficheros siguientes. Se muestran los nombres por defecto en el directorio actual, si bien se pueden indicar otros nombres o directorios al invocarla, mediante el uso de los especificadores **–gf, -vf, -pf** y **–sf.**

g.aut Fichero de texto plano que contiene la especificación de la gramática en notación BNF. Se verifica que la gramática es reducida, y que no contiene reglas repetidas. Constituye la entrada a los generadores.

v.aut, p.aut Ficheros de vocabulario y de tabla de análisis, respectivamente, en formato interno. Son producidas por los generadores y leídas por los analizadores.

s.aut Fichero de texto conteniendo la secuencia de símbolos a ser analizada. Puede ser escrita manualmente, o bien generada de forma automática con **aut.s**.

No es necesario el uso de analizador léxico. El alumno puede escribir directamente los nombres de los símbolos terminales, lo que hace que su uso sea más inmediato e intuitivo.

Algunas opciones de línea de comando.

Aquí presentamos algunas de las opciones más interesantes de la línea de comandos. El resto pueden verse más adelante.

-n Especifica el número de sentencias a generar automáticamente.

+r S Por defecto, cada vez que se invoca el generador de sentencias, éste produce aleatoriamente una nueva sentencia. A veces es importante la repetibilidad, y entonces podemos usar esta opción donde indicamos un valor de semilla S que nos permitirá producir posteriormente la misma sentencia aleatoria. Por defecto, S=0.

+t Utilizamos normalmente una técnica propia que denominamos de longitudes típicas de no-terminales para la generación de sentencias de forma no trivial. Si utilizamos esta opción inhibidora, se toman las distintas reglas al azar; lo que usualmente da unos resultados muy pobres y distantes de las sentencias de uso normal.

+v Inhibe el modo "verbose" para la invocación en scripts.

-d Prefijo de las muy variadas opciones de visualización. Véase el listado completo más adelante.

3.- UNA SESIÓN EJEMPLO

Cuando se ejecuta el programa sin argumento alguno, se obtiene la siguiente ayuda, así como cuando hay algún error en la línea de comando, se le indica al usuario si desea que se visualice esta ayuda.

```
Host$ aut

Direct and reverse LR tool (v10.0.5 28-1-1999)

Usage:

  aut g r|d[#] {-|+c -d{ailbfgsv} +f -g[#] -i +|-l -r[a] +s -t. +v -x file}
  aut s[#] {-# -d{bdfgvem} +e -g[#] +n[#] +v file}
  aut p r|d {-d{psu} -g[#] +v file}


 Functioning modes:
  g       parser Generation (reads gf; writes vf, pf)
  s       random Sentence generation (reads gf; writes sf), with random or
          given (#) length (quite approximately)
  p       Parse (reads vf, pf, sf) with mostly manual error recovery


 Automaton classes:
  r       Reverse discriminating
  d       Direct (classical)
  #       lookahead length: 0 or 1 (default)
```

```
Files:
 -gf path  BNF Grammar File (default g.aut) with rules like:
              proc_list = "proc" heading ";" proc_list | .
 -sf path  Sentence text File (default s.aut) to parse; spaces separate
           terminals.
 -pf path  Parsing table File (default p.aut) in internal format.
 -vf path  Vocabulary File (default v.aut) in internal format.


Options (some do not produce a usable parser):
 -#      Number of sentences
 -c      resolve (reverse) Conflicts (by asking)
 +c      Continue (reverse) state construction beyond conflicts
 -d      Display options:
   a      Automaton states; suboptions:
    i       state situations (also called Items)
    l       List actions for (stack symbol, lookahead) pairs
   b       intput BNF grammar
   d       sentential forms during Derivation
   f       First and follow sets
   g       internal Grammar
   p       (action counts and) stack windows during Parsing
   s       Statistics
   u       strict parsing User time
   v       Vocabulary; suboptions:
    e       estimated typical lEnghts of phrases
    m       Minimum and maximum lenghts of phrases
 +e      disable use of typical lEngths of phrases
 +f      disable Full right-parts computation in situations
 -g[#]   debuGging [option; default 0]
 -i      ask (reverse) Initial state
 +l      build (direct) LALR(1) automaton
 -l      signal (direct) LALR(1) conflicts in LR(1) automaton
 +n[#]   seed [Number; default 0]: disable randomize
 -r[a]   full (reverse) Recognizer [consider Actions]
 +s      build (direct) SLR(1) automaton
 -t.     produce parsing table in plain Text using '.' for empty entries
 +v      disable Verbose output
 -x      produce output and parsing table in LaTeX notation

Author address: <Fortes Gálvez, José> jfortes@dis.ulpgc.es.
```

Veamos en primer lugar la gramática:

```
Host$ cat g.aut
E = E "+" T | T .
T = T "*" F | F .
F = "(" E ")" | "a" .
```

Ejecutemos el generador LR(1):

```
Host$ aut gd -dais

Direct and reverse LR tool (v10.0.5 28-1-1999)

Invoked:  aut gd -dais

Input grammar statistics:
 Number of rules = 6
 Average length of rightparts = 2.00
 Average number of terminal symbols per rightpart = 0.83
 Vocabulary:
  Number of terminal symbols = 5
  Number of nonterminal symbols = 3

Direct (classical) LR(1) automaton


State 1
S' -> ^ . E $ , $
E -> . E + T , $
E -> . T , $
E -> . E + T , +
```

```
E -> . T , +
T -> . T * F , $
T -> . F , $
T -> . T * F , +
T -> . F , +
T -> . T * F , *
T -> . F , *
F -> . ( E ) , $
F -> . a , $
F -> . ( E ) , +
F -> . a , +
F -> . ( E ) , *
F -> . a , *

NS(E) = Go To 2
NS(T) = Go To 3
NS(F) = Go To 4
NS(() = Shift 5
NS(a) = Shift 6


State 2
S' -> ^ E . $ , $
E -> E . + T , $
E -> E . + T , +

NS($) = Shift 7
NS(+) = Shift 8


State 3
E -> T . , $
E -> T . , +
T -> T . * F , $
T -> T . * F , +
T -> T . * F , *

NS($) = Reduce 3
NS(+) = Reduce 3
NS(*) = Shift 9

.
.
(Estados eliminados)
.
.

State 22
T -> T * F . , )
T -> T * F . , +
T -> T * F . , *

NS()) = Reduce 4
NS(+) = Reduce 4
NS(*) = Reduce 4


State 23
F -> ( E ) . , )
F -> ( E ) . , +
F -> ( E ) . , *

NS()) = Reduce 6
NS(+) = Reduce 6
NS(*) = Reduce 6


End of generation.


Automaton statistics:
 Number of states = 23
 Number of transitions & actions = 72
```

Generemos automáticamente una sentencia sobre s.aut.

```
Host$ aut s15 -dd

Direct and reverse LR tool (v10.0.5 28-1-1999)
```

```
Invoked:  aut s15 -dd
<E>:15
<T>:15
<T>:10 * <F>:3
<T>:6 * <F>:6 * a
<T>:2 * <F>:2 * ( <E>:4 ) * a
<F>:2 * a * ( <E>:2 + <T>:2 ) * a
a * a * ( <T>:3 + <F>:4 ) * a
a * a * ( <T>:1 * <F>:1 + ( <E>:2 ) ) * a
a * a * ( <F>:1 * a + ( <T>:1 ) ) * a
a * a * ( a * a + ( <F>:1 ) ) * a
a * a * ( a * a + ( a ) ) * a
a * a * ( a * a + ( a ) ) * a
```

Obsérvese que se señalan los tamaños de las frases que se van a generar, de forma descendente, para cada no-terminal. Por último, se utilizaría el analizador para esta entrada, que nos confirmaría que corresponde a una sentencia y que es analizada correctamente:

```
Host$ aut pd
Direct and reverse LR tool (v10.0.5 28-1-1999)
Invoked:  aut pd
Successful end of parsing.
```

Y, por último, pidamos que genere en forma de fichero LaTeX la tabla de análisis LALR(1) de esta gramática ejemplo.

```
Host$ aut gd +l -x +v
Host$ cat paut.tex
%
% Direct (classical) LR parsing table in LaTeX format
% VersionNumber= 10.0.5 VersionDate= 28-1-1999
%
% GrammarFileName = g.aut
%
%
\begin{figure}\center
{\small\begin{tabular}{|c|cccccc|ccc|}
\hline
  & $\eof $       & $+$    & $*$    & $($   & $)$    & $a$    & $E$    & $T$ & $F$ \\
\hline
 s1     &       &       &       & s5    &       & s6    & 2     & 3     & 4 \\
 s2     & s7    & s8    &       &       &       &       &       &       & \\
 s3     & r3    & r3    & s9    &       & r3    &       &       &       & \\
 s4     & r5    & r5    & r5    &       & r5    &       &       &       & \\
 s5     &       &       &       & s5    &       & s6    & 10    & 3     & 4 \\
 s6     & r7    & r7    & r7    &       & r7    &       &       &       & \\
 s7     & r1    &       &       &       &       &       &       &       & \\
 s8     &       &       &       & s5    &       & s6    &       & 11    & 4 \\
 s9     &       &       &       & s5    &       & s6    &       &       & 12 \\
 s10    &       & s8    &       &       & s13   &       &       &       & \\
 s11    & r2    & r2    & s9    &       & r2    &       &       &       & \\
 s12    & r4    & r4    & r4    &       & r4    &       &       &       & \\
 s13    & r6    & r6    & r6    &       & r6    &       &       &       & \\
\hline \end{tabular}}
```

```
\caption{\label{g}%
%caption goes here
}\end{figure}
%
```

CLASES DIRIGIDAS A GRUPOS ESPECIALES DE ALUMNOS PARA LA ASIGNATURA ALGORITMOS Y ESTRUCTURAS DE DATOS 2.

Paolo Rosso[1]

[1]*Departamento de Sistemas Informáticos y Computación*
Universidad Politécnica de Valencia
e-mail: prosso@dsic.upv.es

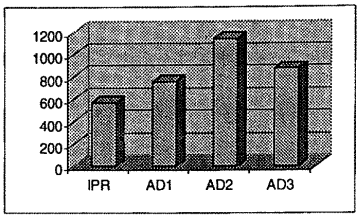
Resumen: En el año académico 1999/2000 la Escuela Universitaria de Informática (EUI) ha organizado unos grupos especiales de alumnos repetidores para algunas asignaturas de primer y segundo curso. Todas estas asignaturas se cursan según el Plan de Ordenación Académica del Centro en el segundo cuatrimestre, donde generalmente es mayor el porcentaje de no-presentados. Estos grupos especiales han permitido a los alumnos repetidores que así lo han deseado, cursarlas en el primer cuatrimestre. El objetivo ha sido dirigir la enseñanza de las materias de manera que el enfoque fuera el adecuado para un grupo de repetidores, alternando adecuadamente los conceptos teóricos con la realización de más ejercicios prácticos y trabajos en el laboratorio. En este artículo se hace una primera reflexión de esta experiencia docente para la asignatura de Algoritmos y Estructuras de Datos 2 (AD2), en función de los resultados de las encuestas al alumnado y al rendimiento académico; el objetivo es evaluar el interés de seguir manteniendo y ampliando estas iniciativas.

1.- LA ASIGNATURA DE ALGORITMOS Y ESTRUCTURAS DE DATOS 2.

La asignatura de Algoritmos y Estructura de Datos II es una de las asignaturas troncales en las titulaciones de Ingeniero Técnico en Informática de Sistemas (ITIS) y de Gestión (ITIG), y en la titulación superior de Ingeniero en Informática (II) de la Universidad Politécnica de Valencia. AD2 se imparte en el segundo semestre del primer curso y tiene una carga lectiva de seis créditos, que se dividen en cuatro de teoría y problemas y dos en prácticas en el laboratorio. Se trata de una asignatura puente entre Introducción a la Programación (IPR), Algoritmos y Estructuras de Datos 1 (AD1) y Algoritmos y Estructuras de Datos 3 (AD3) [1]. Las asignaturas IPR, AD1 y AD2 conforman el núcleo básico en materia de diseño y análisis de algoritmos y de estructuras de datos, y tienen su continuación en la asignatura obligatoria AD3. En comparación con las otras asignaturas de programación cursadas en el primer cuatrimestre, AD2 presenta una tasa de no presentados muy alta [2]. En nuestra opinión, creemos que la baja tasa de aprobados con respecto a los estudiantes matriculados en esta asignatura del segundo cuatrimestre (un poco más de uno de cada cuatro [2]), puede ser debido también a la mayor dificultad que encuentran

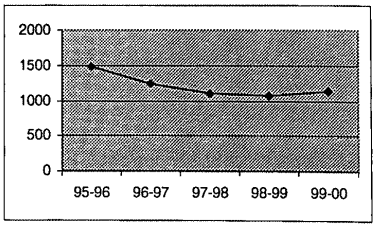
los estudiantes al abordar, a través de la abstracción, el diseño de soluciones a problemas que son, con respecto a AD1, más complejos. En AD2 la abstracción tiene un papel central ya que esta asignatura puente supone una evolución en la formación hacia la madurez del alumno en los temas específicos del diseño y análisis de algoritmos y estructuras de datos. Un análisis más detallado de los resultados en función de los estudiantes presentados, indica que la tasa de aprobados no es muy distinta de la tasa obtenida en alguna de las otras asignaturas de programación, por ejemplo AD1[3].

Figura 1. Estudiantes matriculados en las asignaturas de programación en 1999-2000.



El problema está en el número de alumnos que no se han presentado en ninguna de las dos convocatorias del año académico, que crece de IPR a AD1 y de ésta a AD2, prácticamente duplicándose de una asignatura a otra (véase Figura 1). Como se puede apreciar, la asignatura de AD2 tiene un número muy elevado de alumnos matriculados y debido a su relativamente baja tasa de aprobados se convierte en un cuello de botella de primer curso y en particular para la asignatura AD3, ya que AD2 es prerrequisito recomendado. En la Figura 2 se presenta la evolución del número de alumnos matriculados en AD2 a lo largo de los últimos años académicos. A partir de esta gráfica se puede observar que el número de alumnos sigue siendo elevado pese a que ha experimentado un descenso en los últimos años.

Figura 2. Estudiantes matriculados en AD2.



Una alta tasa de aprobados, si no está acompañada de un alto nivel de formación, proporcionaría sólo una masificación de titulados. Aunque no estamos totalmente de acuerdo con que una enseñanza de calidad se mida en función del número de aprobados (sobre todo en asignaturas básicas en la formación del estudiante que requieren el alcance de un sólido nivel de preparación y deben realizar un efecto de filtro) es indudable que la alta tasa de no-presentados es síntoma de que existen ciertas deficiencias.

En el tercer apartado de este artículo mencionaremos como en los futuros planes de estudios para las titulaciones de ingeniería técnica y superior en informática, la EUI y la Facultad de Informática (FI) han intentado subsanar algunas de las carencias y defectos causados por la

excesiva fragmentación de la materia de programación en distintas asignaturas. A la espera de que en el año académico 2001/2002 se implante el nuevo plan de estudios, la EUI intenta resolver algunos de los problemas, ofertando una enseñanza de la asignatura AD2 dirigida a estudiantes que hubiesen hecho uso por lo menos de una convocatoria en los años anteriores. El objetivo es proporcionar una docencia centrada en la resolución de ejercicios y problemas, permitiendo así una mejor comprensión de la materia, sin repetir todos los conceptos teóricos que el alumno repetidor ya conoce.

2.- LA EXPERIENCIA SIGNIFICATIVA DEL GRUPO ESPECIAL DE REPETIDORES.

Los alumnos matriculados en el grupo especial de alumnos repetidores para las asignaturas Algoritmos y Estructuras de Datos 2 de la EUI han sido 126. En este apartado queremos hacer una primera reflexión de esta experiencia docente para evaluar el interés de seguir manteniendo y ampliando este tipo de iniciativas. Vamos a analizar dos aspectos: la opinión del alumnado sobre el desarrollo de las clases y el rendimiento académico.

a) Evaluación del aprendizaje.

Debido a la excepcionalidad del grupo, hemos decidido hacer una evaluación continuada de los conocimientos, incluyendo una prueba parcial eliminatoria por cada uno de los tres temas (eficiencia, recursión, tipos lineales), de teoría y de prácticas. Para garantizar un aprendizaje adecuado de cada tema, hemos exigido que la nota correspondiente al examen parcial o al ejercicio del examen final de cada tema fuera, como mínimo el 40% de la puntuación máxima de esa parte. El peso específico de las notas de teoría y de prácticas sobre la nota final de la asignatura ha sido, respectivamente, del 60% y del 40%, siendo necesario alcanzar una nota de 4.5 sobre 10 en cada una de las dos partes. La asistencia al laboratorio de prácticas ha sido obligatoria, siendo requisito imprescindible para poder realizar los exámenes. Los alumnos que, por cualquier motivo, han tenido más de tres faltas de asistencia han realizado un examen adicional (con calificación apto o no apto) en el laboratorio de la asignatura. Hemos exigido también que los estudiantes que no tuviesen la calificación de apto en laboratorio de años anteriores, entregasen una memoria de cada práctica con las actividades desarrolladas en el laboratorio y las actividades complementarias. Se han guardado las notas de los parciales para la segunda convocatoria.

b) Análisis de la encuesta al alumnado matriculado.

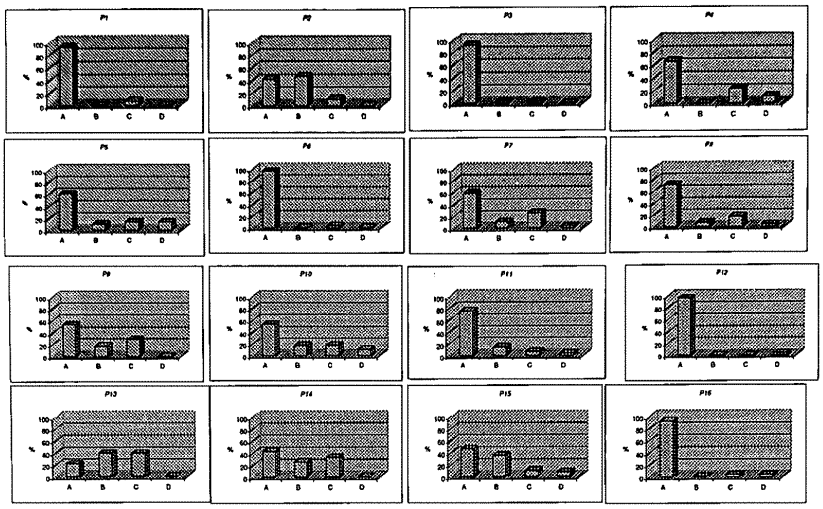
Al final del primer cuatrimestre, la Jefatura de Estudios de la EUI ha realizado una encuesta con el fin de conocer la opinión del alumnado sobre el desarrollo de las clases [4]. Un 25% del total de alumnos del grupo especial de AD2 han contestado a la encuesta. A cada pregunta el alumno podía contestar: *A: Sí; B: No; C: Algunas veces; D: No sé.*

1. Asisto a clase de teoría y problemas
2. Asisto a clase en el laboratorio
3. Es positivo la reducción de explicaciones teóricas a cambio de la realización de más ejercicios prácticos
4. Resulta positivo que el profesor proponga tareas de trabajo personal: ejercicio, estudio de terminados temas, etc.

5. Resulta positivo que las tareas de trabajo personal sean evaluados por el profesor aunque no contribuyan a la calificación de la asignatura.
6. Resulta positivo hacer parciales eliminatorios
7. Considero que el profesor ha alternado adecuadamente la presentación de conceptos teóricos con la realización de ejercicios prácticos y trabajos en el laboratorio.
8. El enfoque dado a la asignatura es el adecuado para un grupo de repetidores
9. El ritmo de las clases me parece adecuado
10. Considero útil el material proporcionado por el profesor
11. El profesor se esfuerza por motivar a los alumnos durante las clases
12. Las sugerencias de los alumnos, si se han producido, han sido atendidas por el profesor
13. Participo activamente en las clases de teoría/problemas/laboratorio planteando preguntas o sugerencias
14. Participo activamente en las clases de teoría/problemas/laboratorio realizando los ejercicios propuestos
15. Me he sentido integrado/a en el grupo
16. Teniendo en cuenta las limitaciones. ¿Crees que vale la pena la puesta en marcha de este tipo de iniciativas?.

El resumen de los resultados de la encuesta se muestra en la Figura 3 [4]. El 92,8% de los estudiantes que entregaron la encuesta considera que vale la pena poner en marcha iniciativas de docencia dirigida (Pregunta 16). Se puede destacar que los estudiantes han considerado: positivo que se hayan hecho parciales eliminatorios (P6: 96,4%) y se haya reducido las explicaciones teóricas a cambio de la realización de más ejercicios prácticos (P3: 92,8%); que el enfoque haya sido el adecuado para un grupo de repetidores (P8: 74%) y que a través de alternar adecuadamente la presentación de conceptos teóricos con ejercicios en la clase de teoría, y trabajos prácticos en el laboratorio (P7: 59,2%) se haya permitido una mayor motivación de los alumnos durante las clases (P11: 75%). También se puede destacar que los alumnos opinan mayoritariamente (P5: 58,6%) que las tareas de trabajo personal, como las memorias de las prácticas, hayan sido evaluadas por el profesor aunque no hayan contribuido a la nota final. Al principio del curso, para que los estudiantes pudiesen manifestar las dificultades encontradas en AD2 durante el/los curso(s) anterior(es), hemos pedido que nos dijesen por escrito, ya que no todos están muy acostumbrados a hablar en público, sus sugerencias. Ello ha sido evaluado positivamente por los estudiantes ya que en su opinión, las sugerencias producidas por ellos han sido atendidas, en la medida de lo posible (P12: 96,4%). Además la encuesta inicial, y otra a mitad del curso (en la que se pedía expresar por escrito puntos fuertes y débiles de la asignatura, del profesor, del alumno y del grupo), ha permitido que los alumnos puedan hacer sugerencias (como por ejemplo resolver más ejercicios prácticos, y exámenes de cursos anteriores) y que al mismo tiempo tomen conciencia de lo que sus sugerencias conllevan (resolver más ejercicios conlleva a mayor participación en las clases).

Figura 3. Resultados de la encuesta para la asignatura AD2.



c) Estudio del rendimiento académico.

Después de esta experiencia docente, podemos hacer un primer balance positivo de ella. El rendimiento académico de los alumnos ha sido bueno, ya que con estas clases dirigidas se ha logrado que el nivel de aprobados superara los obtenidos en cursos anteriores. El porcentaje de estudiantes que se han presentado es del 69% y el porcentaje de aprobados con respecto al de presentados es del 74% (con respecto al de matriculados es del 52%). Entre los alumnos que no han aprobado (o no se han presentado), un 16% tiene aprobada la nota de teoría o de prácticas para las convocatorias del próximo año académico.

d) Características del alumnado matriculado.

Para poder analizar mejor el rendimiento académico y otros aspectos del alumnado de AD2 del grupo especial de repetidores, consideramos de interés conocer las características de los alumnos que han participado en esta iniciativa. La encuesta contenía las siguientes preguntas [4]:

I. Curso académico en qué se matriculó la primera vez:
A: En el curso 93/94 B: En 94/95 C: En 95/96 D: En 96/97 E: En 97/98

II. Procedencia de estudios: A: F. Profesional B: C.O.U. C: Acceso 25 años

III. Nota de entrada: A: Entre 5 y 6 B: Entre 6'1 y 7 C: Entre 7'1 y 8 D: Más de 8

IV. Número de créditos que te quedan para finalizar la carrera:
A: Menos de 15 B: Entre 15 y 25 C: Entre 25 y 45 D: Entre 45 y 60 E: Más de 60

V. Razones por las que te has matriculado en este curso:
A: Para poder acabar la carrera en el primer cuatrimestre
B: Porque la he estudiado para la convocatoria de sept y prefiero aprovechar lo aprendido
C: Porque espero con unas clases dirigidas a alumnos que ya han cursado la asignatura, entender mejor la materia.
D: Otras: indica cuáles.

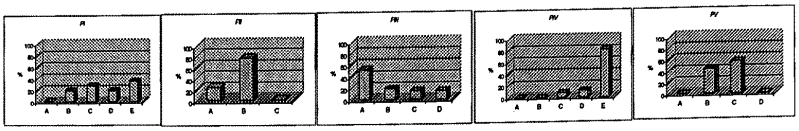


Figura 4. Características del alumnado matriculado en el grupo especial de AD2.

Los resultados obtenidos se muestran en la Figura 4 [4]. Destacamos que el 51,8% de los estudiantes del grupo especial de AD2 (véase pregunta III) tienen la nota mínima de entrada en la EUI (entre 5 y 6). Si además se tiene en cuenta el porcentaje de estudiantes de informática que eligieron la titulación en primera opción (un 50% en ITIS y un 53,2% en ITIG [5], lo que significa que alrededor de la mitad eligió informática como segunda o tercera opción), se puede entender mejor el porqué del escaso interés general y motivación del alumnado. El interés en aprobar la asignatura más que en aprender se refleja en la escasa motivación de los alumnos: la mayoría de ellos reconocieron que no participan activamente en las clases de teoría/problemas/laboratorio planteando preguntas o sugerencias (Pregunta 13 de la Figura 3: 78,6%) y realizando los ejercicios propuestos (P14: 53,6%).

3.- CONCLUSIONES

El plan de estudio todavía vigente ha subsanado algunas de las deficiencias de los anteriores en el contenido práctico al aumentar los recursos para sesiones de laboratorio. Pero al mismo tiempo, con la disminución de créditos y la semestralización de las asignaturas de programación, ha creado distintos problemas. Así por ejemplo, se nota la falta de tiempo para poder favorecer la redundancia con ejemplos, problemas y ejercicios (aprendizaje inductivo vs. deductivo) sobre los mismos conceptos vistos en teoría: una repetición de ideas fundamental para el proceso de aprendizaje del alumno. En el nuevo plan de estudios se ha modificado esta situación de fragmentación de las asignaturas semestrales de programación para reconvertirlas en dos anuales, modificando la distribución temporal, la relación teoría / práctica, y el número total de créditos que pasa de 21 a 24 [6]. A la espera de que se implante este nuevo plan de estudios, la iniciativa tomada por la EUI para la asignatura AD2 ha resultado adecuada para solventar algunos de los problemas analizados en este artículo. Para el próximo año académico, la EUI va a ampliar a dos grupos esta iniciativa, y la FI va a poner en marcha un primer grupo.

AGRADECIMIENTOS.

Deseo expresar mi reconocimiento a Natividad Prieto, subdirectora de ordenación académica de la EUI, y Francisco Marqués, responsable de la asignatura AD2, por sus sugerencias durante la realización de este artículo.

BIBLIOGRAFÍA.

[1] Casanova A. et al. *La Enseñanza de la Programación en los Estudios de Informática de la UPV.* V Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI). 1998.
[2] Herrero C., Rosso P. *Reorganización de Algoritmos y Estructuras de Datos II en el Marco de la Reforma del Plan de Estudios.* V JENUI. 1999.
[3] Rosso P. *Proyecto Docente.* Perfil: Algoritmos y Estructuras de Datos I, II y III. Departamento de Sistemas Informáticos y Computación. Universidad Politécnica Valencia (UPV). Marzo 2000.
[4] Subdirección de Ordenación Académica EUI, *Grupos Especiales. Curso 1999/2000.* Informe Técnico. UPV. Marzo 2000.
[5] Plan Nacional de Evaluación de la Calidad de las Universidades, Informe Técnico (PNECU). EUI, UPV. Marzo 2000.
[6] Herrero C., Rosso P. *Efectos de la Reforma del Plan de Estudios en las Asignaturas de Algoritmos y Estructuras de Datos.* VII Congreso Universitario de Innovación Educativa en las Enseñanza Técnicas (CUIE). 1999.

PROGRAMACIÓN EN INGENIERÍA INDUSTRIAL

Juan Lizandra, M. Carmen

DSIC (Universidad Politécnica de Valencia)
e-mail: mcarmen@dsic.upv.es

RESUMEN: En la mayoría de las titulaciones de Ingeniero Industrial o Ingeniero Técnico Industrial es común encontrar asignaturas básicas de programación. En ellas se suele incluir conceptos de programación (tipos de datos simples, sentencias de control, estructuras de datos, subprogramas, etc.) y la sintaxis de algún lenguaje de programación (C, Pascal, etc.).

Este artículo presenta un estudio de la importancia que tiene la impartición de tales conocimientos en dichas titulaciones. Se muestra una relación de Universidades que ofertan dichas titulaciones y que incorporan en sus planes de estudios asignaturas de este tipo. También se adjunta un estudio comparativo del temario de las distintas asignaturas. Todo ello basándose en la información publicada en las páginas Web de las distintas Universidades.

1.- INTRODUCCIÓN.

Adquirir conocimientos básicos de programación y conocer la sintaxis de algún lenguaje de programación es una exigencia que se está haciendo patente, cada vez más, en todas las titulaciones técnicas universitarias. Las titulaciones de Ingeniero Técnico e Ingeniero Técnico Industrial no son una excepción. El estudio aquí presentado se centrará en ellas.

Con el objetivo de comprobar cuál es el nivel de impartición de este tipo de conocimientos, se ha estudiado el plan de estudios de dichas titulaciones en distintas Universidades Españolas. Concretamente a partir de sus páginas Web [1]. Se han visitado las páginas Web de 61 Universidades [2]-[62], de entre ellas, 42 imparten titulaciones de Ingeniero Técnico Industrial y/o Ingeniero Industrial e incorporan en sus planes de estudios asignaturas de programación básica [2]-[43]. Las 19 universidades restantes, bien no ofertan dichas titulaciones [44]-[59], bien no se ha podido acceder a su página Web [60]-[61] o bien sí que imparten las titulaciones, pero no asignaturas de programación básica [62].

Los Ingenieros Técnicos e Ingenieros Industriales, en sus distintas especialidades, representan uno de los colectivos en los que este tipo de conocimientos son totalmente necesarios. Un porcentaje elevado de ellos deberá estar en contacto con ordenadores, deberá utilizarlos en las mejores condiciones posibles y solventar los distintos problemas con los que se puedan encontrar. Algunas de las tareas, que en algún momento, puede que tengan que realizar y para las cuales necesitan conocimientos de programación, podrían ser las siguientes:
Creación o modificación de programas para realizar cálculos de estructuras, etc.
Personalización de programas de CAD, CAM

2.- ESTUDIO COMPARATIVO

La tabla 1 muestra la relación de Universidades visitadas y cuyos planes de estudios incorporan asignaturas de programación básica. La información sobre asignaturas presenta: el curso en el que se imparte; si es obligatoria, troncal u optativa; y los créditos de teoría y práctica. En algunos casos no aparece la totalidad de esta información debido a que en la página Web no se encuentra publicada la misma. En el tipo de asignatura puede aparecer "Troncal u Obligatoria", ya que a menudo se especifica conjuntamente. Es decir, se listan todas las asignaturas, independientemente del tipo (troncal u obligatoria). Respecto a los créditos impartidos se indica únicamente el total de los mismos si no figura desglosada la carga teórica y práctica. En ella y en la tabla 2 aparecen las siglas ITEI (Ingenieros Técnicos en Electrónica Industrial), ITIEI (Ingeniero Técnico Industrial en Electrónica Industrial), TTIM (Ingeniero Técnico Industrial en Mecánica), ITIE (Ingeniero Técnico Industrial en Electricidad), ITIQI (Ingeniero Técnico Industrial en Química Industrial), Ingeniero Industrial (Ingeniero Industrial), ITIEII (Ingeniero Técnico Industrial en estructuras e instalaciones industriales), ITIT (Ingeniero Técnico Industrial en Textil), IIE (Ingeniero Industrial en Electricidad), IIOI (Ingeniero Industrial en Organización Industrial), ITIQ (Ingeniero Técnico Industrial en Química) e ITIEl (Ingeniero Técnico Industrial en Electrónica).

UNIVERSIDAD. TITULACIÓN Y ASIGNATURAS
Universidad de Cádiz [2]. ITEI. 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad de Córdoba [3]. ITIEI / ITIM / ITIE. 1 curso. Fundamentos de Informática. Troncal (3T/3P)
Universidad de Huelva [4]. ITIE / ITIEI / ITIM / ITIQI. 1 curso. Fundamentos de Informática. Troncal (3T/3P)
Universidad de Jaén [5]. ITIE / ITIM / ITIQI / ITIEI. 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad de Málaga [6]. ITIEI / ITIE / ITIM. 1 curso. Elementos de programación. Troncal (3T/3P) **II.** 1 curso. Fundamentos de Informática. Troncal (6T/3P)
Universidad de Sevilla [7]. II. 1 curso. Fundamentos de Informática. Troncal (9C)
Universidad de Zaragoza [8]. II. 1 curso. Fundamentos de Informática. Troncal (7.5C). 3 curso. Arquitectura de computadores. Optativa (2.4C)
Universidad de Oviedo [9]. ITIE / ITIM / ITIQI. 2 curso. Ampl. de matemáticas y programación. Troncal. 3 curso. Computadores. Optativa **II.** 3 curso. Informática. Troncal. 5 curso. Computadores I. Optativa. 6 curso. Computadores II. Optativa
Universidad de Las Palmas de Gran Canaria [10]. II. 3° y curso de acceso. Informática Básica. Troncal (7.5C)
Universidad de Cantabria. ITIE / ITIEI / ITIM / ITIQI. 1 curso. Fundamentos de Informática. Troncal (6C) **II.** 1 curso. Fundamentos de Informática. Troncal (7.5C). Computadores I. Optativa en 2° y 5° (6C). Computadores II. Optativa en 2° y 5° (6C)
Universidad Católica de Ávila [12]. ITIM. 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad de León [13]. ITIEII. 3 curso. Electrónica básica industrial e informática
Universidad de Salamanca [14]. ITIE / ITIEI. 1 curso. Informática I. Optativa (3T/3P) 2 curso. Informática II. Optativa (3T/3P) **ITIM.** 1 curso. Informática. Troncal (3T/3P). 2 curso. Programación. Optativa (1.5T/3P)
Universidad de Valladolid [15]. ITIE / ITIEI / ITIQI / ITIM. 1 curso. Fundamentos de Informática. Troncal (3T/3P) **II.** 1 curso. Introducción a la Informática. Obligatoria (3C). 1 curso. Fundamentos de Informática. Troncal (6C). 5 curso. Computadores. Optativa (7.5C)

UNIVERSIDAD. TITULACIÓN Y ASIGNATURAS
Universidad de Castilla-La Mancha [16]. II. 1 curso. Fundamentos de Informática. Troncal (6C). 1 curso. Ampliación de informática. Obligatoria (6C). 1er ciclo. Computadores. Optativa (4.5C)
Universidad Autónoma de Barcelona [17]. ITIQI. Informática. Obligatoria **ITIE / ITIEI.** Fundamentos de Informática. Obligatoria. Programación. Obligatoria
Universidad de Lérida [18]. ITIM. 1 curso. Fundamentos de Informática. Troncal (3T/3P). 1 curso. Programación de computadores. Obligatoria (4.5C)
Universidad Politécnica de Cataluña [19]. ITIQI / ITIT / ITIEI / ITIM. 1 curso. Fundamentos de Informática. Troncal (4.5T/1.5P) **II.** 1 curso. Fundamentos de Informática. 2 curso. Informática
Universidad Rovira i Virgili [20]. ITIEI / ITIE. Introducción a la informática. Obligatoria (6C)
Universidad de Vic [21]. ITIEI. 1 curso. Fundamentos de Informática
Universidad de Deusto [22]. ITIEI. 1 curso. Programación. Optativa (7.5C). 2 curso. Informática I. Optativa (6C)
Universidad de Mondragón [23]. ITIM. 1 curso. Informática Básica **II.** 1 curso. Informática Básica
Universidad del País Vasco [24]. ITIE / ITIEI / ITIQI / ITIM. 1 curso. Fundamentos de Informática. Troncal (1.5T/4.5P) **II.** 1 curso. Fundamentos de Informática (6C). 1 curso. Laboratorio de Informática Básica (3C)
Universidad de Extremadura [25]. ITIM / ITIEI / ITIE. 1 curso. Fundamentos de Informática. Troncal (6C) **II.** 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad de La Coruña [26]. ITIEI / ITIE. 1 curso. Fundamentos de Informática. Troncal (3T/3P) **II.** 1 curso. Métodos Informáticos. Troncal (3T/4.5P)
Universidad de Santiago de Compostela [27]. ITIQI. 1 curso. Fundamentos de Informática. Troncal (3T/4.5P)
Universidad de Vigo [28]. II. 5 curso. Computadores I. Obligatoria. 6 curso. Computadores II. Obligatoria
Universidad de Alcalá de Henares [29]. ITIEI. 1 curso. Fundamentos de Informática. Troncal (4.5T/1.5P). 1 curso. Laboratorio de fundamentos de informática. Obligatoria (3C)
Universidad Alfonso X El Sabio [30]. II. 1 curso. Fundamentos de Informática. (9C)
Universidad Autónoma de Madrid [31]. ITIQI. 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad Carlos III de Madrid [32]. **ITIEI / ITIE / ITIM** 1 curso. Programación de Computadores. Troncal (6C) **II.** 1 curso. Programación. Troncal (7C)
Universidad Politécnica de Madrid [33]. II. 4 curso. Métodos Informáticos en Ingeniería. (4T/2P) **IIE.** 5 curso. Computadores I. (6T/4P). 6 curso. Computadores II. (6T/4P) **IIOI.** 6 curso. Integración de la Información. (6T/4P)
Universidad Pontificia de Comillas [34]. ITIE / ITIM / ITIEI. 1 curso. Informática (6C) **II.** 1 curso. Programación. Troncal u obligatoria (6C). 2 curso. Informática. Troncal u obligatoria (6C)
Universidad Rey Juan Carlos [35]. ITIQ. 1°. Fundamentos de Informática. Troncal/Obligatoria (6C)
Universidad Politécnica de Cartagena [36]. ITIEI / ITIE / ITIQI / ITIM. 1 curso. Fundamentos de Informática. Troncal (3T/3P) **II.** 1 curso. Fundamentos de Informática. Troncal (4.5T/3P)
Universidad de Navarra [37]. II. 1 curso. Informática I. 2 curso. Informática Ingeniero Industrial
Universidad Pública de Navarra [38]. ITIM / ITIE. 1°. Fundamentos de Informática. Troncal u Obligatoria (9C). 1°. Laboratorio de Informática. Troncal u Obligatoria (6C)

UNIVERSIDAD. TITULACIÓN Y ASIGNATURAS
Universidad de La Rioja [39]. ITIE / ITIEI. 1 curso. Fundamentos de Informática. Troncal. (6C). 1 curso. Lenguajes de Programación. Obligatoria (4.5) **ITIM.** 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad Jaime I [40]. ITIM. 1 curso. Fundamentos de Informática. Troncal (6C) **II.** 1 curso. Fundamentos de Informática. Troncal (6C)
Universidad Miguel Hernández [41]. II. 1 curso. Fundamentos de Informática. Troncal (3T/3P). 1 curso. Informática Aplicada. Obligatoria (4.5P)
Universidad Politécnica de Valencia[42]. ITIM / ITIQ /ITIE / ITIEI. 1 curso. Fundamentos de Informática. Troncal (3T/3P) **II.** 1 curso. Fundamentos de Informática I. Troncal (1.4T/1.35P). 1 curso. Fundamentos de Informática II. Troncal (2T/1.75P). 1 curso. Introducción a los Computadores. Optativa. (2T/1.75P)
Universidad Nacional de Educación a Distancia [43]. II. 2 curso. Informática I. 5 curso. Computadores I. 5 curso. Informática II. 6 curso. Computadores II

Tabla 1. Universidades Españolas en las que se oferta la titulación de Ingeniero Industrial y/o Ingeniero Técnico Industrial y las asignaturas relacionadas con Programación Básica

El temario de las asignaturas se puede agrupar en dos grandes bloques, que podrían ser los siguientes:

(1) Conceptos de Programación (tipos de datos simples, sentencias de control, estructuras de datos, subprogramas, etc.)

(2) Lenguaje de Programación (C, Pascal, etc.)

De entre las 42 universidades estudiadas, únicamente 12 publican el temario de las asignaturas incluidas en sus planes de estudios. Por lo tanto sólo se podrán utilizar éstas para realizar un estudio más pormenorizado.

Para estas 12 Universidades, en la tabla 2 se muestran las características del temario de las asignaturas. La parte derecha de la tabla muestra los 2 bloques anteriores. En las intersecciones de los bloques y las asignaturas aparece una X en el caso de que esa asignatura tenga en su programa conocimientos del bloque. En la columna (2) en lugar de aparecer una X figura el Lenguaje de Programación utilizado.

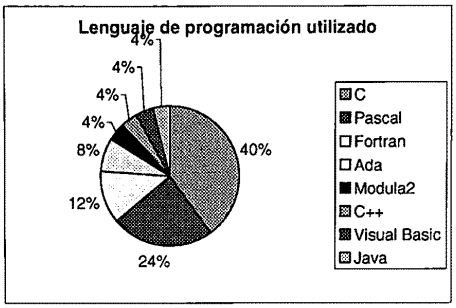
UNIVERSIDAD / ASIGNATURA	(1)	(2)
Universidad de Zaragoza [8] II		
Fundamentos de Informática	X	Pascal
Universidad de Valladolid [15].II		
Fundamentos de Informática	X	C
Universidad de Castilla-La Mancha [16] II		
Ampliación de informática	X	Modula-2
Universidad de Lérida [18]. ITIM		
Fundamentos de Informática	X	C
Programación de computadores	X	C
Universidad Politécnica de Cataluña [19]. ITIQI / ITIT / ITIEI / ITIM		
Fundamentos de Informática	X	C
Universidad del País Vasco [24]. ITIE /ITIEI / ITIQI / ITIM		
(San Sebastián)		
Fundamentos de Informática	X	Pascal
(Eibar)		
Fundamentos de Informática	X	C
II		
Fundamentos de Informática	X	Pascal
Laboratorio de Informática Básica	X	Pascal
Universidad Carlos III de Madrid [32]. ITIEI		
Programación de computadores	X	Fortran
ITIEl / ITIM		
Programación de computadores	X	Ada
II		
Programación	X	Fortran
Universidad Politécnica de Madrid [33]. II		
Métodos Informáticos en Ingeniería	X	Pascal
IIE		
Computadores I	X	C/C++
Universidad Rey Juan Carlos [35]. ITIQ		
Fundamentos de Informática	X	Fortran/Ada
Universidad de Navarra [37]. II		
Informática I	X	Visual BASIC
Informática II	X	Java
Universidad de La Rioja [39]. ITIEl / ITIEI		
Fundamentos de Informática	X	C
Lenguajes de Programación	X	C
ITIM		
Fundamentos de Informática	X	Pascal
Universidad Politécnica de Valencia [42]. II		
Fundamentos de Informática I	X	C
Fundamentos de Informática	X	C

Tabla 2. Relación del temario y asignaturas impartidas en las 12 Universidades incluidas en este estudio

Observando la tabla 2 se pueden extraer los siguientes datos, cuyos valores se muestran en la gráfica 1:

(1) Conceptos de Programación (tipos de datos simples, sentencias de control, estructuras de datos, subprogramas, etc.). 23 imparten conocimientos de programación.

(2) Lenguaje de Programación (C, Pascal, etc.). 23 asignaturas imparten algún lenguaje de programación. Los lenguajes impartidos ordenados por mayor número de ocurrencias son: C se imparte en 10 asignaturas, Pascal en 6 asignaturas, Fortran en 3, Ada en 2, y Modula2, C++, Visual Basic y Java en 1 asignatura.



Gráficas 1. Lenguajes de programación impartidos en las 27 asignaturas analizadas

Del número de asignaturas con respecto al total analizado, se deduce que en el 85% de ellas se imparten conceptos y sintaxis de algún lenguaje de programación.

En cuanto a los lenguajes de programación se puede decir que C es el que mayoritariamente se imparte, 40%, después Pascal, 24%, Fortran, 12%, Ada 8% y por último Modula2, C++, Visual Basic y Java con 4%.

3.- CONCLUSIONES

En este artículo se ha presentado un estudio de las asignaturas de Programación Básica y sus programas, en las titulaciones de Ingeniero Técnico Industrial e Ingeniero Industrial en las Universidades Españolas. Para ello, se han visitado las páginas Web de 61 Universidades Españolas. 43 de ellas ofertan las titulaciones de Ingeniero Técnico Industrial e Ingeniero Industrial y de éstas 42 incorporan en sus planes de estudios asignaturas de Programación Básica. 16 de ellas no ofertan dichas titulaciones y a 2 no se ha podido acceder. De estos datos se deduce la importancia que tienen este tipo de asignaturas en los actuales planes de estudio ya que en la vida real se deberán enfrentar a situaciones cuya resolución dependerá del manejo de ordenadores (creación o modificación de programas).

Se ha realizado un segundo estudio pormenorizado de 12 de las 42 Universidades. Éstas publican el temario de sus asignaturas en sus páginas Web y en ellas se han localizado 27 asignaturas de Programación Básica.

De los datos obtenidos en este segundo estudio se han extraído las siguientes conclusiones. Generalmente, este tipo de asignaturas se imparten en el primer curso e incorporan mayormente conceptos y sintaxis de algún lenguaje de programación, 85%.

C es el lenguaje mayoritariamente impartido, 40%, seguido de Pascal, 24%, Fortran, 12%, Ada 8% y por último Modula2, C++, Visual Basic y Java con 4%.

AGRADECIMIENTOS:

Deseo expresar mi agradecimiento a Vicente Escuderos por toda la ayuda que me ha prestado para la elaboración de este artículo

REFERENCIAS BIBLIOGRÁFICAS

[1] Universidades de España, http://lem.eui.upm.es/edues.html
[2] Universidad de Cádiz, http://www.uca.es
[3] Universidad de Córdoba, http://www.uco.es
[4] Universidad de Huelva, http://www.uhu.es
[5] Universidad de Jaén, http://www.ujaen.es
[6] Universidad de Málaga, http://www.uma.es
[7] Universidad de Sevilla, http://www.us.es
[8] Universidad de Zaragoza, http://www.unizar.es
[9] Universidad de Oviedo, http://www.uniovi.es
[10] Universidad de Las Palmas de Gran Canaria, http://www.ulpgc.es
[11] Universidad de Cantabria, http://www.unican.es
[12] Universidad Católica de Ávila, http://www.ucavila.es
[13] Universidad de León, http://www.unileon.es
[14] Universidad de Salamanca, http://www.usal.es http://www.usal.es
[15] Universidad de Valladolid, http://www.uva.es
[16] Universidad de Castilla-La Mancha, http://www.uclm.es
[17] Universidad Autónoma de Barcelona, http://www.uab.es
[18] Universidad de Lérida, http://www.udl.es
[19] Universidad Politécnica de Cataluña, http://www.upc.es
[20] Universidad Rovira i Virgili, http://www.urv.es
[21] Universidad de Vic, http://www.uvic.es
[22] Universidad de Deusto, http://www.deusto.es
[23] Universidad de Mondragón, http://www.muni.es
[24] Universidad del País Vasco, http://www.ehu.es
[25] Universidad de Extremadura, http://www.unex.es
[26] Universidad de La Coruña, http://www.udc.es
[27] Universidad de Santiago de Compostela, http://www.usc.es
[28] Universidad de Vigo, http://www.uvigo.es
[29] Universidad de Alcalá de Henares, http://www.alcala.es
[30] Universidad Alfonso X El Sabio, http://www.uax.es
[31] Universidad Autónoma de Madrid, http://www.uam.es
[32] Universidad Carlos III de Madrid, http://www.uc3m.es
[33] Universidad Politécnica de Madrid, http://www.upm.es
[34] Universidad Pontificia de Comillas, http://www.upco.es
[35] Universidad Rey Juan Carlos, http://www.urjc.es
[36] Universidad Politécnica de Cartagena, http://www.upct.es
[37] Universidad de Navarra, http://www.unav.es
[38] Universidad Pública de Navarra, http://www.unavarra.es
[39] Universidad de La Rioja, http://www.unirioja.es
[40] Universidad Jaime I, http://www.uji.es

[41] Universidad Miguel Hernández, de Elche, http://www.umh.es
[42] Universidad Politécnica de Valencia, http://www.upv.es
[43] Universidad Nacional de Educación a Distancia, http://www.uned.es
[44] Universidad de Almería, http://www.ualm.es
[45] Universidad de Granada, http://www.ugr.es
[46] Universidad Pablo de Olavide, http://www.upo.es
[47] Universidad de las Islas Baleares, http://www.uib.es
[48] Universidad de La Laguna, http://www.ull.es
[49] Universidad Internacional SEK en Segovia, http://www.sek.edu/UnivEspanaIndex.htm
[50] Universidad de Barcelona, http://www.ub.es
[51] Universidad Pompeu Fabra, http://www.upf.es
[52] Universidad Ramón Llull, http://www.url.es
[53] Universidad Complutense de Madrid, http://www.ucm.es
[54] Universidad Pontificia de Salamanca en Madrid, http://www.upsa.es
[55] Universidad San Pablo CEU de Madrid, http://www.ceu.es
[56] Universidad de Murcia, http://www.um.es
[57] Universidad de Alicante, http://www.ua.es
[58] Universidad de Valencia, http://www.uv.es
[59] Universitat Oberta de Catalunya, http://www.uoc.es
[60] Universidad de Girona, http://www.udg.es
[61] Universidad Antonio de Nebrija, http://www.unnet.es
[62] Universidad de Burgos, http://www.ubu.es

PROGRAMACIÓN LITERARIA

Manuel Perera Domínguez

Dpto. Ciencias de la Computación e Inteligencia Artificial – Univ. Sevilla
e-mail: perer@cica.es

RESUMEN: Es un hecho admitido que la documentación de los programas es usualmente muy pobre y poco comprensible para las personas que no han participado en su desarrollo lo que dificulta enormemente las tareas de mantenimiento del software y su reutilización posterior. La escasa legibilidad de los programas limita en gran medida el uso docente de los listados de código fuente como medio de aprendizaje de la programación a través de ejemplos. La "programación literaria" (*literate programming*) permite redactar programas más legibles, mantenibles, reusables y portables, adecuados para uso docente y correctamente documentados.

1.- LEGIBILIDAD DE LOS PROGRAMAS DE ORDENADOR.

Desde los años 50 hasta la actualidad la edición y publicación del código fuente de los programas de ordenador ha permanecido esencialmente invariable: un listado completo del código tal y como será compilado al que se han añadido comentarios explicativos. Los programas suelen ser difíciles de entender como lo prueba el que apenas aparezcan grandes listados en los textos universitarios de informática. Dos ejemplos extensos y complejos de lo máximo que puede conseguirse con las técnicas tradicionales de documentación de código se encuentran en conocidos textos académicos sobre sistemas operativos [12] y compiladores [5]. Muchas veces los programas son más difíciles de leer que de escribir.

En la práctica profesional de la programación se concede gran importancia a la lectura de programas: "...creo que uno de los mejores tests de capacidad de programación consiste en entregar al programador unas treinta páginas de código y ver con qué velocidad es capaz de leerlas y comprenderlas ... la mejor forma de prepararse es escribir programas y estudiar grandes programas que hayan escrito otros ... Uno tiene que estar siempre dispuesto a leer código escrito por otros, luego escribir el suyo propio y luego dárselo a revisar a otras personas" (Bill Gates en [11] pág. 94). Esta importancia también ha sido reconocida por la comunidad científica que plantea la conveniencia del aprendizaje de la programación a través de ejemplos: un buen programa debe enseñar al lector. Citemos a N. Wirth en su clásica obra "Algoritmos + estructuras de datos = programas": "Estos ejemplos están también pensados como ejercicios de aprendizaje de la lectura de programas que demasiado frecuentemente se olvida en favor de la escritura... lo que permanece del arsenal de métodos y enseñanza es la selección y presentación cuidadosa de ejemplos clave" ([14] pág. XV). Para observar cómo la falta de legibilidad dificulta el uso docente de programas de ejemplo véase el deficiente libro de soluciones [13] de los problemas del conocido texto de Kernighan y Ritchie sobre el lenguaje C [6].

Por supuesto la legibilidad es muy importante para un estudiante de programación pues un profesor humano ha de leer y comprender sus programas para puntuarlos [2].

2.- MANTENIBILIDAD Y REUSABILIDAD DEL SOFTWARE.

El que un programa de ordenador sea legible es condición necesaria para que éste sea mantenible y reutilizable, sobre todo si grupos diferentes de personas redactan y mantienen el software. Los estudios de ingeniería del software indican que el tiempo necesario para entender código representa muchas veces más del 50% del coste de mantenimiento. La escasa legibilidad del código y la falta de una documentación actualizada es lo habitual por lo que muchas veces el mantenimiento degenera en una especie de "programación experimental" en la que no se entiende bien el programa y no se sabe cómo afectarán los cambios. Puede generarse código ya existente, eliminar o introducir dependencias desconocidas, etc. en un frenético proceso de agregación de código nuevo, ejecución del programa y observación de los nuevos errores producidos. Un programa mal documentado y poco legible será casi con toda seguridad desechado si su programador abandona la empresa u organismo que lo desarrolló.

3.- PROGRAMACIÓN LITERARIA.

La idea central de la programación literaria es que los programas deberían ser escritos para que sean entendibles por un humano y no para que sean directamente legibles por un compilador. Para ello se aúnan las labores de programación y documentación, tradicionalmente separadas, utilizando conjuntamente un lenguaje de programación y un lenguaje de descripción textual. El programa se considera como un documento estructurado dirigido hacia un lector humano por lo que se fomenta una exposición clara de las ideas del programador y un cierto estilo "literario" de escritura, algo totalmente inexistente en los típicos y crípticos listados de programas solo entendibles por un experto. Un programa literario puede incluir tablas, figuras, referencias bibliográficas, ecuaciones, índices automáticos, etc. para lograr una documentación de muy alta calidad que permita percibir la estructura de una compleja pieza de software. Para facilitar la comprensión del documento (programa) éste se estructura en pequeñas unidades conceptuales interrelacionadas entre sí de manera que la apariencia final de un programa literario es la de un documento en el que se ha intercalado código fuente, justo al contrario de lo habitual. La programación literaria es en principio adaptable a cualquier lenguaje de programación y no restringe la libertad del programador si bien de manera natural fomenta un enfoque estructurado, descendente (*top-down*) y de refinamiento sucesivo.

4.- ENSEÑANZA LITERARIA DE LA PROGRAMACIÓN.

La enseñanza literaria de la programación persigue los siguientes objetivos:
Los estudiantes deben comprender la importancia de la documentación para redactar programas legibles y mantenibles. La documentación ha de ser parte intrínseca del programa y no un añadido improvisado en el último momento como es actualmente.

- Cambiar la actitud del estudiante: su tarea principal no ha de ser instruir a un computador sobre qué hacer sino poder comunicar a otro humano qué quiere que haga

el programa. Ha de ser tan importante "comunicar" con las personas que lean el programa como con el compilador del lenguaje de programación.

- Desarrollar el trabajo cooperativo en pequeños grupos. Los estudiantes deben leer y revisar el código redactado por sus compañeros, señalando posibles defectos y mejoras y aprendiendo de sus aciertos en un ciclo de realimentación positiva.

- Considerar un programa como un diseño dirigido a la resolución de un problema y que se plantea en términos de objetivos para los que ha de crearse código para cumplirlos.

- Mostrar el método por el cual se derivó la solución del problema. La mayor parte de los libros de texto sobre programación dan soluciones "totales" sin mostrar el proceso de diseño. En este sentido la programación literaria refuerza el papel del diseño y permite ver el programa como una consecuencia de unas decisiones de diseño bien documentadas.

- Obligar a especificar qué hace el programa en su totalidad y cada parte del mismo. Esto permite comprobar si los estudiantes han entendido bien el problema que han de resolver. Es una actitud muy común ante un problema de programación el empezar a codificar de inmediato y ejecutar repetidas veces observando los errores producidos. Muchas veces los errores son debidos a una inadecuada comprensión del problema planteado y a que el estudiante no sabe (no es capaz de explicar) qué hace exactamente el código que desarrolla y por qué. Se ha de procurar reducir en lo posible el uso irreflexivo de un depurador.

- Ayudar a desarrollar programas claros, limpios, bien estructurados y coherentes.

Los lenguajes de programación se dirigen fundamentalmente a obtener código legible por los compiladores, lo cual obliga a una serie de convenciones sintácticas que dificultan la comprensión de los programas. El listado de un programa se presenta y se lee de manera lineal en un orden que no coincide con el de escritura: un programador no escribe desde la primera hasta la última línea de código "de una vez" sino que realiza varios ciclos de correcciones, vueltas atrás, saltos hacia adelante, etc. Desde el punto de vista de la comprensión de un programa es mucho más conveniente presentarlo siguiendo su orden lógico de desarrollo, prescindiendo en la medida de lo posible de las limitaciones sintácticas del lenguaje de programación empleado. Hay que evitar que los estudiantes organicen su conocimiento recién adquirido de programación en términos de la sintaxis del lenguaje concreto que utilicen. Los programadores expertos organizan su contenido en unidades conceptuales que transcienden la sintaxis concreta del lenguaje de programación y que pueden encontrar una expresión natural en un programa literario.

5.- SISTEMAS DE PROGRAMACIÓN LITERARIA.

Un sistema de programación literaria debe permitir lo siguiente:

- Generar conjuntamente documentación y código en un lenguaje de programación de alto nivel que residirán en un mismo fichero. La unión de documentación y código es un programa literario. Resulta mucho más fácil el ajustar la documentación a los cambios realizados en el código evitando su desfase e inutilidad.

- Especificar subdivisiones lógicas en el programa (módulos, secciones, etc.) que pueden interrelacionarse. Cada sección lógica consta de una parte de documentación y otra de código fuente.

- Escribir el programa en un orden lógico sin restricciones sintácticas. Las partes del programa deben presentarse en un orden que ayude a entender cómo funciona éste, que no es necesariamente el orden requerido por el compilador.

- Extraer automáticamente a partir del programa literario el código fuente del programa en el orden requerido por el compilador del lenguaje empleado.

- Extraer automáticamente la documentación a partir del programa literario. La documentación debe explicitar claramente lo que hace el programa y los medios de solución empleados e incluye el código fuente. Pueden usarse fórmulas matemáticas, tablas, gráficos y enlaces hipertextuales para ayudar a su comprensión. Esta documentación debe ser tan extensa como se precise y puede incluir detalles que habitualmente no están presentes en el código de un programa: consideraciones de diseño, discusión de soluciones alternativas, pruebas de ejecución, etc. que pueden resultar de interés.

- Generar un documento final (programa literario) de la máxima calidad visual para lo cual se utilizarán técnicas de *pretty printing* pues está demostrado que el resalte tipográfico ayuda a la comprensión de los programas [1].

El primer sistema de programación literaria fue "Web", desarrollado por Donald Knuth (considerado por muchos como el mejor programador del mundo) cuando tuvo que reescribir TeX. Knuth ha empleado extensivamente la programación literaria [9] con la que ha desarrollado programas muy portables y extensamente utilizados [7,8,10]. En la actualidad existen numerosos sistemas de programación literaria (CWeb, Noweb, etc.) aptos para una gran diversidad de lenguajes (Pascal, C, C++, Fortran, Modula 2, Ada, Lisp, Prolog, Maple, APL, etc) y que se diferencian en el grado de dependencia que presentan respecto al lenguaje de programación concreto y el formateador de texto empleado (usualmente LaTeX). Todos ellos tienen en común unas herramientas (procesadores) denominadas "Tangle" y "Weave". La primera de ellas reordena el código fuente presente en el programa literario de acuerdo con el orden lógico seguido y produce el fichero que será finalmente compilado. Internamente funciona mediante expansiones de macros. La herramienta Weave genera la documentación del programa literario (su "listado"). Estas herramientas pueden englobarse en un modelo general de documentación estructurada. En [3] y [4] pueden examinarse programas complejos y muy portables desarrollados en la comunidad académica utilizando programación literaria.

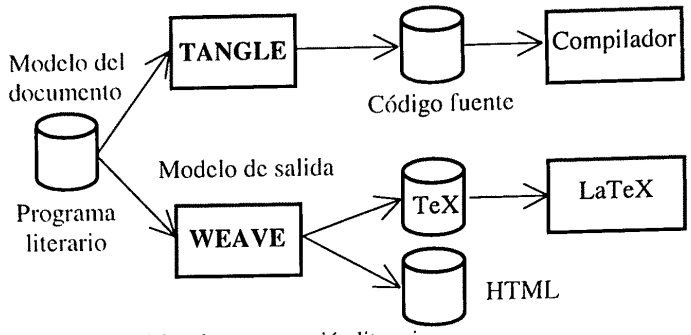


Figura 1: Sistema típico de programación literaria.

6.- PROBLEMAS.

La introducción de la programación literaria en un primer curso universitario de programación puede plantear una serie de dificultades importantes en la medida que el alumnado disponga de algunos conocimientos previos de programación por rudimentarios que sean. Estos conocimientos provienen del esfuerzo individual y de la lectura de revistas comerciales y libros de "bricolaje" informático. Este tipo de estudiantes, muy comunes en las escuelas de ingeniería, ingresan en la universidad con una serie de expectativas sobre lo que se les va a enseñar y tienen una fuerte preferencia hacia la enseñanza de los aspectos sintácticos del lenguaje de programación más en boga en ese momento (C hace unos diez años, luego C++, ahora Java, etc.) por lo que sin duda alguna no simpatizarán con el enfoque propugnado por la programación literaria. Además en el mundo laboral de la programación de ordenadores en la actualidad no se hace ningún tipo de hincapié en la documentación del software y desgraciadamente es una situación muy común el que no se requiera en absoluto. Por tanto aquellos estudiantes con experiencia laboral o que tengan contacto con programadores profesionales serán considerablemente escépticos acerca de las ventajas que pudiera aportarles una enseñanza literaria de la programación.

Muchos estudiantes creen firmemente en la "mística del programador solitario" que es aquel que trabaja aislado, sin haber recibido enseñanza formal y que demuestra su valía profesional realizando programas muy extensos y complejos, sin documentación y sin metodología alguna. Este tipo de programador era común en la industria hasta hace pocos años y su "leyenda" está firmemente asentada. Los aspirantes a "programadores solitarios" quedarán desagradablemente sorprendidos por el énfasis que la programación literaria realiza sobre la documentación del software y por el tiempo y disciplina de pensamiento que requiere la redacción de un buen programa literario.

Aunque resulte paradójico, el campo de las tecnologías de la información es mucho mas conservador en su práctica profesional de lo que pudiera intuirse y resulta ser muy poco permeable a las ideas provenientes de la comunidad universitaria. Es una actitud muy común y fácilmente contrastable el no desear aprender nada nuevo si ya se tiene algún conocimiento en el área. Lo general es el deseo de obtener resultados rápidos utilizando esencialmente los mismos métodos ya conocidos y que todos usan. Quizás esta actitud sea un importante factor explicativo de la salida laboral como programadores de muchos universitarios no formados en programación (matemáticos, físicos, etc.) Otros problemas de índole secundario son:

- La identificación de la programación literaria con el sistema operativo UNIX, resultado de su origen académico y que variará conforme se elaboren nuevas herramientas.

- El volumen de documentación a redactar por el profesor en un curso de programación literaria es muy superior al hoy día utilizado en la enseñanza de la programación. Sin embargo se tiene garantizada la publicación de unos "apuntes de clase" de calidad.

- Los alumnos ingresan en la universidad con una pobre capacidad de expresión escrita.

- La evaluación de un programa literario puede ser una tarea delicada pues es difícil el puntuar la exposición y claridad de estilo de un texto.

- Por supuesto vuelve a surgir la vieja cuestión de qué hacer con las faltas de ortografía.

7.- CONCLUSIONES.

Existe una importante separación entre la enseñanza universitaria de la programación de ordenadores y su desarrollo profesional. Es la industria informática la que realmente guía el cambio en esta área; basta con observar la abundante oferta de formación privada orientada exclusivamente al mundo laboral y que goza de un enorme éxito a pesar de su alto coste económico. La enseñanza universitaria de la programación no debe renunciar a intentar influir en la industria mediante la formación de sus futuros empleados. La programación literaria propone una enseñanza con énfasis en la redacción de documentación, la legibilidad de los programas y su mantenibilidad. Fomenta el trabajo cooperativo entre los alumnos y les obliga a mejorar su capacidad de expresión escrita y a comprender, estructurar y explicitar mejor los programas que realicen. Estas habilidades les serán sin duda muy útiles en sucesivos cursos universitarios y potenciarán su futuro desempeño profesional.

BIBLIOGRAFÍA

[1] BAECKER, R.M. y MARCUS, A. *Human Factors and Typography for More Readable Programs.* ACM Press/Addison-Wesley, 1990.

[2] DEIMEL, L.E. y NAVEDA, F. *Reading Computer Programs: Instructor's Guide and Exercises.* Carnegie Mellon University, 1990.

[3] FRASER, C.W. y HANSON, D.R. *A Retargetable C Compiler: Design and Implementation.* Addison-Wesley, 1995.

[4] HANSON, D.R. *C Interfaces and Implementations: Techniques for Creating Reusable Software.* Addison-Wesley, 1996.

[5] HOLUB, A.I. *Compiler Design in C.* Prentice Hall, 1990.

[6] KERNIGHAN, B.W. y RITCHIE, D.M. *El lenguaje de programación C, 2a. ed.* Prentice Hall, 1991.

[7] KNUTH, D.E. *TeX: The Program.* Vol. B of *Computers and Typesetting.* Addison-Wesley, 1986.

[8] KNUTH, D.E. *Metafont: The Program.* Vol. D of *Computers and Typesetting.* Addison-Wesley, 1986.

[9] KNUTH, D.E. *Literate Programming.* Center for the Study of Language and Information, Lecture Notes No. 27, Stanford University, 1992.

[10]KNUTH, D.E. *The Stanford Graphbase. A Platform for Combinatorial Computing.* ACM Press/Addison-Wesley, 1993.

[11]LAMMERS, S. *Programadores en acción.* Anaya Multimedia, 1988.

[12]TANENBAUM, A.S. *Sistemas operativos: diseño e implementación.* Prentice Hall, 1988.

[13]TONDO, C.L. y GIMPEL, S.E. *The C Answer Book.* Prentice Hall, 1989.

[14]WIRTH, N. *Algoritmos + estructuras de datos = programas.* Eds. Del Castillo, 1980.

ENSEÑANZA DE LA PROGRAMACION PARALELA EN LA UNIVERSIDAD POLITECNICA DE VALENCIA.

J.L. Pérez[1], A.M. Vidal[1], V. Hernández[1]

[1]*Facultad de Informática, Universidad Politécnica de Valencia*
e-mail: jlperez,avidal,vhernand,@dsic.upv.es

RESUMEN: En este artículo se presenta una visión de la enseñanza de las técnicas de programación paralela en la Escuela Universitaria de Informática y la Facultad de Informática de la Universidad de Politécnica de Valencia (UPV). Dicha visión se centra principalmente en la descripción de la asignatura, Lenguajes y Entornos de la Programación Paralela, impartida en el último curso de la titulación Ingeniero en Informática. En dicha asignatura se pretende que el alumno adquiera un conocimiento avanzado en el manejo de las técnicas de programación paralela utilizado el paradigma de paso de mensajes.

1.- INTRODUCCIÓN.

La posibilidad de ejecutar eficientemente aplicaciones paralelas en agrupaciones de PC's, con un coste muy reducido, unido a la existencia de múltiples estándares de programación paralela, aplicables a todo tipo de plataformas, hace asequible la utilización de técnicas de programación paralela en aplicaciones con un elevado coste computacional. Es por esta razón, y por la creciente demanda de este tipo de conocimientos por parte del mundo laboral, que consideramos fundamental la enseñanza de dichas técnicas dentro de la formación del futuro Ingeniero en Informática.

En las siguientes secciones se pretende esbozar la enseñanza de las asignaturas que describen las técnicas de Programación Paralela en la Escuela y la Facultad de Informática de la Universidad Politécnica de Valencia durante los últimos años. El estudio se centrará en la asignatura de quinto curso de la Facultad de Informática, Lenguajes y Entornos de la Programación Paralela. Dicha asignatura fue creada hace 3 años dentro del marco de la reforma del Plan de Estudios de la Facultad de Informática efectuada en 1994, y pretende hacer frente a los retos que esta planteaba.

2.-DESCRIPCIÓN DE LA ASIGNATURA LENGUAJES Y ENTORNOS DE LA PROGRAMACIÓN PARALELA

La asignatura de Lenguajes y Entornos de la Programación Paralela, tal y como se ha comentado, se ha venido impartiendo en el quinto curso de la titulación de Ingeniero Informático. El alumno de esta asignatura previamente ha cursado la asignatura, Arquitectura de Computadores, troncal de cuarto curso, donde se le han presentado los conceptos básicos de las arquitecturas paralela. También suele haber cursado la asignatura, Computación Paralela, como asignatura optativa de tercer curso, donde se aprenden los conceptos fundamentales de la computación en paralelo y de la programación paralela orientada a *paso de mensajes.*

a) Coordenadas principales

La posición de la asignatura en el último semestre del Plan de Estudio es un factor determinante de la organización que proponemos para ella. Tiene un carácter final muy acusado; es decir, el alumno que la elige completa con ella su formación en esta disciplina. Debe tenerse en cuenta que la asignatura coexiste, a veces, con el desarrollo del proyecto final de carrera, y que los alumnos están pensando ya en su próxima orientación profesional y en aplicar los conceptos adquiridos en el ejercicio de su profesión.

Todo ello nos lleva a proponer una organización de la asignatura que puede considerarse atípica respecto a la propuesta de otras asignaturas. En primer lugar, la asignatura debe tener un carácter muy práctico, imprescindible en una asignatura en la que priman aspectos de programación. Además, proponemos organizar la asignatura en distintos Casos de Estudio, elegidos de forma que ayuden al alumno a alcanzar los objetivos previstos, evitando en lo posible lecciones magistrales y provocando con ello una implicación real del alumno en el proceso de aprendizaje.

Es también importante destacar que la diversificación de especialidades en el último curso hace que el número de alumnos de cursan esta asignatura sea reducido (entre 20 y 30). Esto permite que la implantación de está metodología no suponga un excesivo consumo de recursos, tanto a nivel humano, como de recursos físicos.

b) Objetivos

De acuerdo con la filosofía de la asignatura, expresada en el subapartado anterior, se plantean los siguientes *objetivos generales*:

- Ser capaz de diseñar de forma eficiente una aplicación paralela.
- Adquirir habilidad en el manejo avanzado de, al menos, dos entornos de programación paralela.
- Ser capaz de evaluar el comportamiento, tanto a nivel teórico, como a nivel práctico, de los algoritmos implementados.

Los objetivos descritos pueden refinarse dando lugar a los siguientes *objetivos específicos*:

O1 Conocer los entornos de programación paralela más típicos.

O2 Ser capaz de determinar los parámetros básicos de una máquina paralela.

O3 Ser capaz de implementar rutinas paralelas sencillas.

O4 Ser capaz de implementar algoritmos paralelos robustos, portables, legibles y reutilizables.

O5 Saber convertir el planteamiento secuencial de un problema a su equivalente paralelo sin modificar su estructura sustancialmente.

O6 Aquirir habilidad en el manejo avanzado de dos entornos de programación paralela.

O7 Ser capaz de evaluar y comparar los algoritmos paralelos implementados, así como los entornos de programación utilizados.

O8 Ser capaz de implementar esquemas algorítmicos típicos utilizando el entorno de programación paralelo adecuado.

c) Metodología

La asignatura se divide una serie de *Casos de Estudio* que el alumno debe resolver, individualmente, como si fueran pequeños proyectos. Estos Casos de Estudio se eligen convenientemente, de forma que su desarrollo exija al alumno alcanzar los objetivos previstos. Cada Caso de Estudio se desarrolla en tres etapas

Información Teórica. En esta etapa se le proporciona al alumno la información necesaria para entender el problema que se aborda en cada Caso de Estudio, así como las especificaciones requeridas para su resolución. Esta información se suele presentar en sesiones teóricas. Como resultado de esta etapa el alumno debe disponer de un modelo teórico del problema a resolver.

Diseño e Implementación. A partir de la solución teórica al problema obtenida en la etapa anterior, el alumno debe escoger la implementación que consideré más eficiente, utilizando el entorno de programación que considere más adecuado. Así mismo, debe realizar todas aquellas pruebas experimentales que le permitan validar y evaluar su implementación.

Memoria. Por último, basándose en el trabajo realizado, el alumno confeccionará una memoria que describa el marco teórico, la aproximación algorítmica, los resultados y las conclusiones obtenidas.

Utilizando esta metodología se pretende obtener las siguientes ventajas:

Aprendizaje orientado a la resolución de problemas. El alumno se enfrenta a una serie de problemas prácticos (problemas tipo), similares a los que se presentan en la mayoría de aplicaciones reales.

Carácter práctico de la asignatura. El alumno piensa en problemas y no en contenidos. Desaparece la separación entre teoría y práctica; todo contenido teórico tiene sentido dentro de su ámbito de aplicación.

Estimular la responsabilidad del alumno en el aprendizaje. El alumno es el protagonista de las soluciones informáticas que obtiene. Se siente responsable ya que es él quien decide la estrategia a utilizar para resolver un problema determinado.

d) Casos de Estudio propuestos

Los casos de estudio propuestos (ver tabla 1) tiene como finalidad la consecución, de forma gradual, de los objetivos propuestos en la asignatura. Generalmente, los casos de estudio guardan relación con problemas resueltos en otras asignaturas utilizando un paradigma de programación secuencial, lo cual evita que el alumno deba prestar atención a detalles ajenos a la implementación. Además, existe la posibilidad de modificar el contenido de los Casos de Estudio atendiendo al perfil de los alumnos matriculados en un determinado año.

Caso	Título	Objetivos
CS1	El entorno PVM: Operaciones elementales y comunicaciones.	O1-3
CS2	Distribuciones de datos en entornos de programación paralela. Producto Matriz-Vector	O3-5
CS3	Algoritmo paralelos para el cálculo de las distancias mínimas en un grafo dirigido.	O3-5 y O8
CS4	El entorno MPI: Producto Matriz-Matriz	O4-6
CS5	Algoritmos paralelos para la resolución de Sistemas Triangulares.	O4-7
CS6	Algoritmo paralelos de tipo Divide y Vencerás.	O4-8

Tabla 1: Casos de Estudio de la asignatura de Lenguajes y Entornos de la Programación Paralela (curso 99/00) y relación con los objetivos.

La organización de la asignatura requiere una constante realimentación con el alumno con el fin de poder establecer el ritmo adecuado que permita, por una parte, la consecución de los objetivos, y por otra respeto a la filosofía del método que garantiza al alumno libertad para ir organizando su trabajo.

El contenido de los casos de estudio es flexible, ya que es susceptible de ser alterado en cursos sucesivos sin que se resientan los objetivos de la asignatura. Lo mismo sucede con los entornos de programación utilizados (en nuestro caso PVM y MPI), ya que podrían ser substituidos por

otros entornos de programación orientados a *paso de mensajes* sin que el contenido de los Casos de Estudio se alterase significativamente.

3.-EVALUACIÓN DE LA ASIGNATURA LENGUAJES Y ENTORNOS DE LA PROGRAMACIÓN PARALELA.

En este apartado se muestra una evaluación crítica de la asignatura, Lenguajes y Entornos de la Programación Paralela, durante los tres años que esta ha sido impartida en la Facultad de Informática. En primer lugar se describirán las deficiencias observadas durante el transcurso de los dos primeros años de impartición. Posteriormente se describirán las modificaciones introducidas en el presente curso (tercer año) con el fin de mejorar la asignatura y las mejoras que se han percibido una vez finalizado este. Finalmente, se muestra un estudio realizado sobre los alumnos de la asignatura.

a) Deficiencias observadas durante los 2 primeros cursos

Excesiva complejidad en la implementación. Algunos casos de estudio resultaron tener una complejidad excesivamente elevada para algunos de los alumnos, debido a carencias en la estrategia de diseño de los algoritmos paralelos y a la incapacidad de hacer una separación entre las diferentes partes que constituyen la aplicación paralela. Como resultado, los algoritmos paralelos implementados eran monolíticos y difíciles de depurar.

Poca adecuación del hardware. Las prácticas se realizaban en un laboratorio, que constaban de 20 PC's. Las máquinas de este laboratorio se hallan interconectadas mediante una red Ethernet junto con el resto de máquinas de la UPV. Este hecho afecta significativamente al coste de comunicaciones de los algoritmos paralelos implementados e impide una evaluación efectiva de los mismos.

Escasez de material y referencias bibliográficas. También se observó, en los dos primeros años de impartición de la asignatura, una falta de referencias adecuadas a las necesidades de los alumnos. Los manuales estándares de PVM y MPI son excesivamente poco amigables para usuarios no avanzados en su manejo, y por tanto, inadecuados para su uso docente.

b) Modificaciones introducidas en el tercer curso

Simplificación y reordenación de los casos de estudio. Se ha intercambiado el orden de algunos casos de estudio con el fin de alcanzar los objetivos de una forma más gradual. Se ha añadido el Caso de Estudio dedicado a la implementación de las distribuciones de datos más comunes (CS2) con el fin de facilitar el diseño de los siguientes Casos de Estudio. Además, en los primeros Casos de Estudio, se han proporcionado a los alumnos pequeñas librerías de funciones para la realización de determinadas tareas.

Utilización de una metodología de diseño de los algoritmos paralelos. Se ha enseñado la necesidad de estructurar adecuadamente las aplicaciones paralelas con el fin de facilitar la etapa de diseño e implementación. Se ha incidido especialmente en la conveniencia de encapsular cada una de las partes de la aplicación paralela, lo que permitirá la reutilización de gran parte del código de un Caso de Estudio en futuros Casos de Estudio.

Disponibilidad de un Agrupación de PC's en modo dedicado. Se adquirió una máquina paralela de coste reducido, que constaba de 5 PC's interconectados con una red Fast Ethernet aislada del resto de ordenadores de la UPV. Con esta máquina se espera que los alumnos puedan realizar una evaluación realista de las prestaciones de los algoritmos paralelos

Libros de apuntes y bibliografía en Internet. Se han proporcionado boletines explicativos para cada caso de estudio, junto con libros de apuntes destinados a la explicación de los entornos PVM y MPI. Además, se han proporcionado direcciones de Internet donde obtener información adicional para la elaboración de algunos de los Casos de Estudio.

c) Mejoras observadas durante el transcurso del tercer curso

Diseño de los algoritmos paralelos. El hecho de proporcionar funciones ya implementadas supone que el alumno realiza más rápidamente los primeros Casos de Estudio, lo cual le proporciona una motivación positiva, e impide, que alumnos con una base pobre en conocimientos de paralelismo abandonen la asignatura prematuramente. Además, el alumno comprende los beneficios que supone el uso de la programación estructurada y modular en el ámbito de la programación paralela, y la posibilidad, utilizando esta técnica, de reutilizar gran parte del código implementado.

Evaluación de los algoritmos paralelos. La posibilidad de disponer de una máquina paralela, pese a su reducido coste, de uso exclusivo para los alumnos de la asignatura, les ha permitido evaluar de forma cómoda los algoritmos paralelos implementados. Al tener la posibilidad de contrastar las prestaciones reales de los algoritmos paralelos con las teóricas, el alumno ha prestado especial atención al análisis a priori del coste de los algoritmos implementados.

d) Evaluación de la asignatura según los alumnos

En la Tabla 2 se muestra el resultado de una encuesta realizada por los alumnos que en cierta forma corrobora las mejoras efectuadas en el curso 99/00. Los resultados son meramente orientativos debido a que únicamente fueron contestadas por la mitad de los alumnos matriculados.

	Pregunta	Cont. por curso (0-10)		
		97/98	98/99	99/00
Interés	Interés general por la asignatura	7,4	7,6	8
	Utilidad futura de la asignatura	6	7,2	7,5
Dificultad	La asignatura se sigue con comodidad	5,9	6,25	7,7
	% de trabajo realizado en clase	50%	50%	63%
Asimilac.	Grado de asimilación de la asignatura	7,8	8,1	8,3
	Evaluación global de la asignatura	6,8	7,7	8,1
	Media de todas las preguntas	6,4	6,6	7,7

Tabla 2: Resultados de la evaluación de Lenguajes y Entornos de la Programación Paralela por los alumnos de la asignatura.

4.-CONCLUSIONES

La asignatura de Lenguajes y Entornos de la Programación Paralela pretende que los alumnos con la titulación de Ingeniero en Informática, tengan la posibilidad de tener un conocimiento avanzado en el manejo de las técnicas de programación paralela utilizando librerías de paso de mensajes. Podemos afirmar, tras la experiencia de los últimos tres años impartiendo esta asignatura, que la metodología utilizada permite conseguir los objetivos establecidos inicialmente. La consecución de estos objetivos no supone un esfuerzo excesivo al alumno de último curso de carrera. Por otra parte, la disponibilidad una máquina paralela muy económica, formada por una agrupación de solo cinco PC's, ha demostrado ser suficiente para permitir a los alumnos observar los beneficios prácticos de la programación en paralelo.

5.-BIBLIOGRAFÍA

[1] A.M. Vidal y J.L. Pérez, Introducción a la Programación en MPI. Ed. Servicio de Publicaciones de la Universidad Politécnica de Valencia. Dep. Legal: V-1280-2000, 2000.

[2] V.Kumar. Introduction to Parallell Computing: Design and Analysis of Algorithms. The Benjamin/Cummings Publishing Com. Inc. 1994

[3] A. Geist, PVM: Parallel Virtual Machine. The MIT Press, 1994

[4] I.Foster. Designing and Building Parallel Programs, (Online), Addison-Wesley.**http://www-unix.mcs.anl.gov/dbpp/**

ÁREA DE TELEMÁTICA

Responsable: Dr. Gregorio Fernández Fernández
(Universidad Politécnica de Madrid)
E-mail: gfer@dit.upm.es

EVALUACIÓN Y DISEÑO DE REDES: Una asignatura cercana al mundo empresarial

Ana Belén Lago[1], Jose Luis del Val[2]

[1]E.S.I.D.E. (Estudios Superiores de Ingeniería de Deusto)
e-mail: ablago@eside.deusto.es

[2]E.S.I.D.E. (Estudios Superiores de Ingeniería de Deusto)
e-mail: val@eside.deusto.es

Resumen: En el presente artículo se presenta la guía de la asignatura *Evaluación y Diseño de Redes*, dirigida a alumnos de quinto curso de los estudios de Ingeniería en Informática de E.S.I.D.E. Esta asignatura pretende completar los conocimientos ya impartidos en otras asignaturas troncales de las titulaciones de Ingeniero Técnico Industrial e Ingeniero en Informática con el fin de proporcionar los elementos básicos para el diseño de redes que van a soportar actividad de comercio electrónico, desde el diseño de la arquitectura física de la red y la configuración de sus servidores, hasta el diseño de las técnicas de seguridad de sistemas y aplicaciones y la aplicación de una política y herramienta de gestión de red. Se pretende inculcar a los alumnos los conceptos necesarios para que sean capaces de diseñar, configurar y administrar una red, pensada para dar soporte a una solución de comercio electrónico. Esta asignatura además del temario teórico, posee un amplio programa práctico que permitirá al alumno asentar sus conocimientos.

1.- INTRODUCCIÓN.

En 1996, se comenzó en la Facultad de Ingeniería de la Universidad de Deusto (ESIDE) la implantación de los planes de estudios de Ingeniería Técnico en Informática de Gestión e Ingeniero en Informática, siguiendo las directrices marcadas por el Ministerio de Educación y Ciencia. Los estudios siguen una organización basada en un primer ciclo de Ingeniería Técnica en Informática de Gestión, de tres años de duración y un segundo ciclo de Ingeniero en Informática de dos años. Este nuevo plan sustituyó progresivamente al plan de estudios anterior.

La implantación del nuevo plan de estudios supuso una reestructuración tanto en el número de asignaturas del Departamento de Telecomunicaciones como en sus contenidos, reflejo de la importancia que se ha concedido a esta área en la formación de los alumnos

La Distribución de asignaturas correspondientes al Departamento de Telecomunicaciones es la siguiente:

a) Fundamentos de Redes de Computadores

Se imparte durante el segundo cuatrimestre de segundo curso y cuenta con 6 créditos

En ella se proporciona una visión global de los fundamentos de las Redes de Computadores, describiendo los modelos de arquitectura OSI y TCP/IP, y explicando conceptos básicos tales como conmutación de circuitos y paquetes, multiplexación, control de flujo y de errores, y haciendo una descripción funcional de los distintos niveles del modelo TCP/IP que se exponen con detalle en asignaturas posteriores.

Con esta asignatura se persigue que todo alumno cuente al finalizar el primer ciclo conocimientos generales sobre este área.

b) Redes de Computadores

Se imparte durante el primer semestre de cuarto curso y cuenta con 6 créditos.

La asignatura posee tres partes: una dedicada a la tecnología LAN, otra a la tecnología ATM y la última, a la tecnología de interconexión de redes.

c) Sistemas de Transmisión de Datos

Se imparte durante el segundo semestre de cuarto curso y cuenta con 4,5 créditos

En esta asignatura se revisa con detalle los fundamentos y operación de los niveles de red, transporte y aplicación, con especial mención a los protocolos TCP/IP.

d) Evaluación y Diseño de Redes

Se imparte en el primer cuatrimestre de quinto curso y cuenta con 7,5 créditos. Es una asignatura optativa y es nueva tanto en su existencia como en contenidos respecto de los anteriores planes de estudios.
La orientación de la asignatura es novedosa en nuestra Facultad, puesto que hasta ahora, ninguna asignatura se había centrado en el diseño de redes.

El auge experimentado por las extranets y el comercio electrónico nos ha llevado a pensar en la importancia que tendrá para nuestros alumnos ser capaz de realizar un buen diseño de red, capaz de soportar las exigencias de esta nueva actividad, como son un buen rendimiento, seguridad tanto dentro de la empresa como en las transacciones comerciales,... todo ello orientado a proporcionar un buen servicio al cliente.

Esta reflexión, nos ha llevado a preparar un temario que sirva para que el alumno tenga una formación que resulte relevante para el mundo empresarial en el que va a desenvolverse próximamente.

Además, esta asignatura pretende llevar a cabo una labor integradora de todos los conocimientos impartidos en las distintas asignaturas a lo largo de las dos titulaciones y ver su aplicación en la empresa.

Entre los nuevos conceptos destacar:

- Una parte dedicada a las plataformas y aplicaciones de comercio electrónico existentes, explicando las necesidades de rendimiento, fiabilidad, seguridad,... que debe ser capaz de proporcionar la red sobre las que se implanta. Todo ello acompañado de ejemplos que ayudan al alumno a asimilar los conocimientos.

- Un capítulo centrado en explicar los diferentes tipos de servidores, destacando la necesidad de una buena configuración de éstos para lograr un buen rendimiento, así como de técnicas para reducir el retardo en la web.

- Un tema dedicado a un aspecto cada vez más acuciante, la seguridad. En él se mostrará la seguridad en dos vertientes: uno, el diseño de una red segura, explicando las diferentes técnicas y elementos utilizados para este fin y el otro, la seguridad en las transacciones comerciales, centrándose en temas como los certificados digitales, criptografía y la seguridad en el comercio electrónico. Este tema irá complementado con unas sesiones en el laboratorio donde se podrán poner en práctica estos conocimientos.

- Otro capítulo mostrará los conceptos de gestión de redes, sistemas y aplicaciones y cómo, gracias a la gestión, se puede mejorar el rendimiento y el funcionamiento de la red.

- Otra parte de la asignatura se centrará en explicar los aspectos a considerar a la hora de diseñar una red, los elementos que entran a formar parte, las diferentes tecnologías de red existentes, indicando cuándo se debe elegir una u otra. Esta parte irá complementada con ejemplos y ejercicios encaminados a diseñar una red partiendo de cero.

En resumen, se ha intentado crear una asignatura muy práctica, mostrando casos reales que aproximen al alumno al mundo empresarial y complementada con sesiones en el laboratorio que le permiten poner en práctica los conceptos teóricos.

Queda un aspecto que no se va a contemplar en la asignatura este primer año y es añadir un capítulo dedicado a fundamentos de teoría de colas y unas prácticas con una herramienta de simulación de redes. Sin embargo, esta modificación se va a incluir en próximas ediciones de la asignatura.

A continuación, se exponen más en profundidad los contenidos de la asignatura.

2.-OBJETIVOS DE LA ASIGNATURA.

a) Objetivos generales.

- Conocer los conceptos básicos del comercio electrónico.
- Conocer y comprender los pasos a seguir en una empresa a la hora de introducirse en el comercio electrónico.
- Conocer los problemas y soluciones para instalar una intranet/extranet.
- Conocer y comprender los distintos aspectos a considerar a la hora de diseñar una red.

b) Objetivos específicos.

- Conocer las distintas plataformas y aplicaciones de comercio electrónico.
- Conocer y ser capaces de instalar y configurar los diferentes tipos de servidores para que realicen su función de la manera más óptima.
- Conocer y prever los problemas relacionados con la seguridad tanto interna como en el comercio electrónico.
- Comprender la importancia de la gestión de redes, sistemas y aplicaciones y conocer el modo de aplicar una herramienta concreta.
- Ser capaz de diseñar y configurar una red, asignando direcciones IP en casos concretos, trabajando con subredes, superredes y máscaras de subred de longitud variable y configurando los distintos dispositivos, routers, switches,... para alcanzar el mayor rendimiento.
- Conocer y seleccionar las tecnologías de red más adecuadas en cada situación.

3.-TEMARIO DE LA ASIGNATURA.

a) Contenido teórico.

1.- Introducción
2.- Plataformas y aplicaciones de comercio electrónico
Arquitectura e-business, componentes e-business, plataformas para el comercio electrónico, aplicaciones e-business.
3.- Tecnologías del nivel de aplicación
Correo electrónico, servidores:servidor web, ftp,...., técnicas para reducir el retardo en la web.
4.- Seguridad
Certificados digitales, criptografía, comercio electrónico y seguridad.
5.- Gestión de redes, sistemas y aplicaciones.
Administración de sistemas y aplicaciones, gestión de red: clasificación de la gestión de red, monitorización de red, control de red, protocolos y arquitectura de gestión de red.
6.- Redes TCP/IP
Política de direcciones IP, subredes, máscaras de subred de longitud variable, superredes. Asignación dinámica de direcciones, direcciones ficticias. Configuración de routers. Protocolos de encaminamiento dinámico.
7.- Infraestructuras de comunicaciones

Aspectos a considerar a la hora de diseñar una red. Bloques constituyentes. Introducción a las tecnologías de red, LAN, VLAN, WAN, ADSL, VPN, redes de acceso inalámbrico.

b) Contenido práctico.

En una asignatura de este tipo es necesario preparar un plan de prácticas que ayude al alumno a asimilar los conocimientos, pero hablar de realización de prácticas de instalación y configuración de redes es complicado, no sólo por el número de alumnos sino también por el material necesario.

A pesar de todas las limitaciones, se ha elaborado una asignatura de carácter eminentemente práctico que permita alcanzar los objetivos planteados.

Las prácticas están divididas en dos partes:

En la primera, que está relacionada con la instalación y administración de una red, se ha seleccionado Microsoft como entorno de trabajo. Consiste en varias sesiones en el laboratorio donde el alumno instalará Windows NT Server y además, se le mostrará cómo realizar la administración de usuarios, archivos, configuración de TCP/IP junto con sus servicios y servicio de impresión. También deberá instalar y configurar un servidor de correo, un servidor web, un servidor FTP,... De forma adicional se le enseñará a instalar un servidor de certificaciones y a operar con él.

En la segunda parte, que está relacionada con la necesidad de comunicaciones seguras en el comercio electrónico, el alumno deberá realizar un proyecto en el que se apliquen conceptos de seguridad como SSL, clave pública, clave privada, firma digital,...

4.-METODOLOGÍA DE LA ASIGNATURA.

La metodología que se seguirá en la asignatura será la de clases magistrales con contenido teórico, complementadas con resolución de ejercicios prácticos y ejemplos que ayudan a asimilar los conceptos teóricos adquiridos.

En la parte práctica, se realizarán una serie de sesiones en el laboratorio donde se impartirán los conocimientos necesarios para que el alumno sea capaz de construir una extranet basada en Windows NT Server. Además, el alumno deberá desarrollar una práctica que aplique los conceptos aprendidos sobre seguridad.

De manera voluntaria, el alumno podrá realizar trabajos para profundizar en ciertos aspectos. Aquellos trabajos más interesantes serán expuestos en clase con lo que se logra fomentar la participación.

5.- EVALUACIÓN.

a) Teoría.

La nota de teoría corresponde con un 70% de la nota final.

b) Prácticas.

La nota de prácticas corresponde con un 30% de la nota final, se evaluará mediante unos ejercicios que comprueban los conocimientos adquiridos en el laboratorio (20%) y el desarrollo de un proyecto que aplique los conceptos aprendidos sobre seguridad en las clases teóricas (10%).

c) Trabajos.

Como alternativa a la realización de los ejercicios de la parte práctica (20%), el alumno podrá presentar un trabajo previamente autorizado por el profesor.

6.- BIBLIOGRAFIA.

a) Bibliografía básica.

Apuntes de la asignatura.
Artículos de prensa seleccionados.

b) Bibliografía de consulta.

"The e-commerce book: building the e-empire". *Koper, S., Ellis, J. Academic Press. 2000.*
"Seguridad y Comercio en el Web". Garfinkel, S., Spafford, G. McGraw-Hill. 1999.
"SNMP, SNMPv2, SNMPv3 and RMON 1 and 2". *Stallings, W. Addison-Wesley. 1999.*
"Client-Server information systems: a business-oriented approach". Goldman, James E., Rawles, Philip T.,Mariga, Julie R. John Wiley & Sons. 1999.
"Introduction to client-server systems: a practical guide for systems professionals". *Reanud, Paul E. John Wiley & Sons. 1996*
"Los secretos de la seguridad en Internet" *Vacca, John R. Anaya Multimedia. 1997.*

7.- CONCLUSIÓN.

En este artículo hemos descrito el programa teórico y práctico de la asignatura Evaluación y Diseño de Redes que se va a impartir el próximo curso 2000/2001 a los alumnos de 5° de Ingeniería en Informática.

Esta asignatura pretende completar los conocimientos ya impartidos en otras asignaturas troncales de las titulaciones de Ingeniero Técnico Industrial e Ingeniero en Informática. El objetivo es proporcionar los elementos básicos para el diseño de redes que van a soportar actividad de comercio electrónico, los cuales van desde el diseño de la arquitectura física de la red junto con la configuración de sus servidores, hasta el diseño de las técnicas de seguridad de sistemas, aplicaciones y de comercio electrónico y la aplicación de una política y herramienta de gestión de red.

PERSONALIZACIÓN DE PRÁCTICAS PARA EL DESARROLLO DE SISTEMAS TELEMÁTICOS AVANZADOS

Carlos A. Iglesias Fernández[1], Mercedes Garijo Ayestarán[1]

[1] *Departamento de Ingeniería de Sistemas Telemáticos, E.T.S. I. Telecomunicación, Universidad Politécnica de Madrid*
cif@gsi.dit.upm.es, mga@gsi.dit.upm.es

RESUMEN: El artículo presenta la experiencia del desarrollo de un curso práctico para desarrollar Sistemas Telemáticos Avanzados, introduciendo técnicas de inteligencia artificial. El curso se ha estructurado con el fin de que los alumnos se motiven y puedan decidir el contenido y la carga de trabajo que quieren dedicarle.

1.- INTRODUCCIÓN.

Este artículo resume la experiencia docente en la asignatura optativa "Laboratorio de Sistemas Inteligentes" de la especialización en Telemática en la E.T.S.I. Telecomunicación de la Universidad Politécnica de Madrid.

En el diseño de la asignatura, se ha intentado conseguir los siguientes objetivos:

1. Complementar de forma práctica los conocimientos teóricos aprendidos en la asignatura "Sistemas Inteligentes".
2. Relacionar esta asignatura con otras asignaturas de la especialidad, como "Gestión de Redes de Telecomunicación" e "Ingeniería del Software".
3. Motivar a los alumnos, permitiéndoles escoger o proponer las prácticas que realizan.
4. Permitir la realización del laboratorio en casa.
5. Utilizar herramientas de dominio público y lenguajes de programación de propósito general.

2.- ESTRUCTURA DEL CURSO.

El curso se ha estructurado en tres prácticas: introductoria, práctica breve y proyecto final.

PRÁCTICA INTRODUCTORIA

La primera práctica tiene como objetivo comprobar el funcionamiento de sistemas inteligentes disponibles en la red y evaluarlos, simulando que se está realizando un test de Turing. Tiene una gran componente lúdica, y a la vez permite que quien la realiza se pregunte por las técnicas que

hay detrás de estos demostradores, así como que se reflexione sobre temas como qué hace a un programa inteligente.

PRÁCTICA BREVE

La segunda práctica es una práctica breve sobre uno de los siguientes temas: búsqueda, lógica borrosa, sistemas expertos, agentes móviles o aprendizaje. Para todos los temas se proponen varias prácticas, de diferente dificultad, y se permite que los alumnos propongan otras, o utilicen herramientas diferentes. En la evaluación se ha hecho énfasis en la interpretación de los resultados obtenidos y en la metodología seguida en las pruebas.

Las prácticas de búsqueda se han realizado con una biblioteca de búsqueda en C++ [TB97] o en Java [RN96]. Se han propuesto diversas prácticas de problemas clásicos: puzzle-8, los misioneros, las ocho reinas, las torres de Hanoi, etc.

En las prácticas de lógica borrosa se ha utilizado principalmente FuzzyCLIPS [FZ94], y se han propuesto prácticas de control borroso de un ventilador, un péndulo invertido, una bomba de calor o un camión.

Para las prácticas de aprendizaje, el objetivo es que el alumno aplique los algoritmos aprendidos en teoría o estudie otros nuevos y, sobre todo, que vea todo el proceso de minería de datos, cómo optimizar los algoritmos y cómo interpretar los resultados. Se ha utilizado fundamentalmente la biblioteca Java WEKA [WF99]. En el caso de redes neuronales, se ha optado por utilizar SNNS [SNNS95]. También se han empleado programas Java para visualizar algoritmos genéticos, optimizando funciones.

PROYECTO FINAL DE LA ASIGNATURA

El proyecto final pretende que el alumno aborde un problema que le interesa, y desarrolle una aplicación de una complejidad razonable. En esta práctica, se fomenta que los alumnos se organicen en grupos. El único requisito que hemos puesto es que se escoja un tema diferente del escogido en la práctica breve.

Para estas prácticas, se ha hecho uso principalmente de dos plataformas desarrolladas en nuestro grupo: JAEN (Java Agent Environment) [PHA99] y Pfc-Robocup [MAR00].

JAEN proporciona a los usuarios una biblioteca de clases Java para construir agentes. Los agentes se distinguen por poder asumir objetivos y ofrecer o solicitar servicios a otros agentes. La comunicación entre agentes se hace con enchufes TCP o con CORBA, empleando en ambos casos mensajes FIPA. Para el laboratorio, se ha hecho énfasis en la utilización de agentes con capacidad de razonamiento, que incluyen un motor de inferencias Jess [FRI99]. El uso de JAEN facilita la comprensión de la programación basada en agentes. Se ha definido un lenguaje en Jess para programar servicios, objetivos, planes para los objetivos, etc. Entre las prácticas desarrolladas en esta asignatura, cabe destacar:

- Sistema multiagente para la gestión de una impresora, complementando a la asignatura "Gestión de Redes de Telecomunicación".

- Sistema multiagente para planificación de viajes (hoteles y vuelos).

- Sistema multiagente para modelar el flujo de trabajo en un equipo de desarrollo software, complementando a la asignatura "Laboratorio de Ingeniería Software"

- Sistema multiagente para modelar el flujo de trabajo en una empresa de traducción, modelando la distribución de trabajo, selección de traductores y planificación de presupuestos.

- Sistema multiagente para jugar a "los chinos", modelando creencias de los oponentes.

- Sistema multiagente para planificación de obras públicas, modelando distribución de tareas.

- Sistema multiagente para un servicio de televisión a la carta, modelando la cooperación, una interfaz gráfica, e integrando un módulo de aprendizaje de preferencias desarrollado con Weka.

- Sistema multiagente para modelar una casa inteligente, modelando la cooperación y distribución de carga entre electrodomésticos.

Pfc-Robocup es una biblioteca Java que permite programar jugadores para competir en la Robocup. Esta biblioteca emplea JavaBeans para almacenar y procesar la información sensorial recibida del servidor de fútbol, y un motor de inferencias para determinar el comportamiento de los jugadores. La base de conocimiento de los futbolistas se ha estructurado en varios módulos: módulo de acciones, que permite especificar las acciones posibles (elementales o compuestas) de cada agente, módulo de conductas, que contiene reglas para determinar qué acción realizar según la información, y módulo de estrategia, que define la estrategia de un jugador, ataque o defensa, según su estado y el del resto de agentes.

Entre otras, también se ha propuesto:

- uso de Gatt [COR94], una aplicación de algoritmos genéticos a la planificación de clases, profesores y exámenes de nuestra escuela. Se ha coordinado esta práctica con el jefe de estudio para analizar la viabilidad de su aplicación en nuestra escuela.

- personalización de páginas Web, realizando modelado de usuarios. En esta práctica se ha integrado Jess en un sistema con servlets, bases de datos, y recuperación de información para modificar la apariencia de las páginas según el perfil del usuario.

- realización de "mascotas inteligentes", integrando Jess y Java. En estas prácticas, al igual que en las que tienen interfaz gráfica en Jaen, se ha hecho énfasis en la aplicacion del patrón de diseño MVC (Modelo-Vista-Controlador) y JavaBeans para aprender a integrar un sistema de reglas en una aplicación gráfica.

3.- SOPORTE TELEMÁTICO DOCENTE DEL LABORATORIO

El laboratorio se ha coordinado a través de una página Web, donde se accede al resto de servicios:

- Una lista de correo de la asignatura para dar avisos importantes. Ademas se anotan en la lista de avisos de la página.

- Un servidor ftp del software asociado a la asignatura.
- Una herramienta de generación de faqs (*Frequently Asked Questions list*), realizado con la herramienta de libre distribución FAQ-O-MATIC desarrollada por Jon Howell en la Universidad de Darmouth, que permite que tanto los profesores como los alumnos añadan preguntas y respuestas.
- Servicio de tutoría por correo electrónico.
- Página web de progreso de cada alumno.
- Sistema de envío de notas y comentarios por correo electrónico, desarrollado en Perl.

4.- CONCLUSIONES

Frente al esquema tradicional de definir un conjunto de prácticas obligatorias para todos, hemos optado por definir un conjunto abierto de prácticas en diferentes temas. Como resultado, cada alumno puede determinar en qué temas explicados en la teoría desea profundizar, y qué desea hacer en cada uno. Con esto hemos conseguido que los alumnos estén motivados y realicen prácticas con gran calidad.

Este esquema puede ser aplicado con grupos no muy numerosos de alumnos, ya que necesita una atención personalizada tanto para la elección y definición de las mismas, como para su seguimiento.

Las herramienta de FAQ no ha sido empleada apenas por los alumnos para introducir respuestas como esperábamos, pero sí ha servido para centralizar las respuestas de los profesores.
El dominio de la Robocup ha gozado de gran aceptación, pero es un dominio muy complejo. Para futuros años nos proponemos extender la biblioteca actual para que ofrezca más acciones y estrategias, y los alumnos se puedan centrar en modelar la coordinación de los agentes, y podamos realizar una liga interna.

5.- BIBLIOGRAFÍA

[COR94] Evolutionary Timetabling: Practice, Prospects and Work in
Progress, Dave Corne, Peter Ross, Hsiao-Lan Fang. Presented at the UK Planning and Scheduling SIG Workshop, Strathclyde, September '94, organised by Patrick Prosser

[FRI99], Jess, The Java Expert System Shell, Ernest J. Friedman-Hill, Distributed Computing Systems, Sandia National Laboratories, Livermore, CA, Version 5.0, 1999.

[FZ94] FuzzyCLIPS 6.02-A User's Guide. Knowledge Systems Laboratory, Institute for Information Technology, National Research Council, Canada, 1994.

[HAY99] Pablo Haya, Raúl García y Javier Torres, Manual de Jaen, Universidad Politécnica de Madrid, 1999.

[MAR00] Álvaro Martínez Reol, Desarrollo de un Sistema Multiagente integrando Jess y Java para la Robocup, Universidad de Valladolid, 2000.

[MOL98] Mole 3.0 User Manual, v1.0, Universidad de Stuttgart, 1998.

[RN96] Russell, S. y Norvig, P. Artificial Intelligence: A Modern Approach. Prentice Hall, 1995.

[SNNS95] SNNS. Stuttgart Neural Network Simulator, User Manual Version 4.1. University of Stuttgart, 1995.

[TB97] Kim W. Tracy and Peter Bouthoorn. Object-Oriented Artificial Intelligence using C++. Computer Science Press, 1997.

[WF99] Ian H. Witten y Eibe Frank. Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, 1999.

APROXIMACIÓN A LA SIMULACIÓN DE UN SISTEMA DE TELEFONÍA MÓVIL CELULAR

Gustavo H. Rodríguez Rodríguez, Pablo V. Hérnández Morera, Carmen N. Ojeda Guerra, Ernestina A. Martel Jordán, Elsa M. Macías López, Carlos Ramírez Casañas

Departamento de Ingeniería Telemática
Universidad de Las Palmas de Gran Canaria (ULPGC)
e-mail: grodri@cma.ulpgc.es

RESUMEN: En este artículo se describe un módulo de prácticas diseñado para los alumnos de primer curso de Ingeniería Técnica de Telecomunicación de la ULPGC en el que se simulan ciertos aspectos de un sistema de telefonía móvil celular (STMC). Los principales objetivos que se han perseguido son presentar a los alumnos la programación de computadores como una disciplina claramente relevante dentro del ámbito de las telecomunicaciones, y al mismo tiempo favorecer que los alumnos puedan ir descubriendo las conexiones que existen entre las distintas materias del plan de estudios desde los primeros cursos.

1.- INTRODUCCIÓN

En la Escuela Universitaria de Ingeniería Técnica de Telecomunicación de la ULPGC se imparten las asignaturas Algorítmica y Programación. Se trata de dos asignaturas ubicadas en el primer y segundo cuatrimestre, respectivamente, del primer curso de las titulaciones: Sistemas de Telecomunicación, Sonido e Imagen, Sistemas Electrónicos y Telemática.

Desde hace algunos años, los profesores que impartimos docencia en las asignaturas anteriores hemos adoptado un modelo de prácticas basado en la resolución de un problema global de carácter práctico que orienta o actúa como hilo conductor de todas las prácticas que se realizan a lo largo del curso. De este modo, el alumno utiliza las estructuras de datos y de control adecuadas para cada subproblema concreto, incrementa modularmente los programas utilizando la comunicación a través de parámetros y se da cuenta de que cada uno de los subproblemas resueltos forman parte de un todo, es decir, están relacionados de alguna forma y proporcionan la solución de un problema más complejo. Así, por ejemplo, en [Ramírez96] se presenta la realización de un simulador de un lenguaje ensamblador para una máquina virtual, en la que son abarcados todos los bloques de contenidos que configuran las asignaturas [Martel96], [Ojeda97].

En cada curso académico se ha intentado que el problema vaya enfocado a algún tema en particular de los currículos de alguna de las cuatro titulaciones que se imparten en la E.U.I.T.T. de la ULPGC. Así, por ejemplo, el simulador de un lenguaje ensamblador para una máquina virtual presentado en [Ramírez96], está especialmente relacionado con el currículo de la titulación de Sistemas Electrónicos, aunque dichas prácticas se imparten por igual para las cuatro titulaciones. Con ello se pretende que en la formación del alumno no solamente se incluyan temas específicos de su especialidad.

En este artículo se describe un módulo de prácticas donde se propone la simulación de ciertos aspectos de un STMC, tema que está especialmente relacionado con el plan de estudios de las titulaciones de Telemática y Sistemas de Telecomunicación. Uno de los principales objetivos propuestos ha sido conseguir que los alumnos no consideren las asignaturas Algorítmica y Programación como materias "ajenas" o poco útiles para su titulación, empleando temas de actualidad y que conecten con sus intereses. Esta dificultad es comprensible, debido a que en primer curso los alumnos no han cursado asignaturas específicas de su titulación, y con sus conocimientos previos es difícil reconocer la enorme importancia y utilidad que el software y la programación tienen en todos los ámbitos del mundo científico y tecnológico hoy en día.

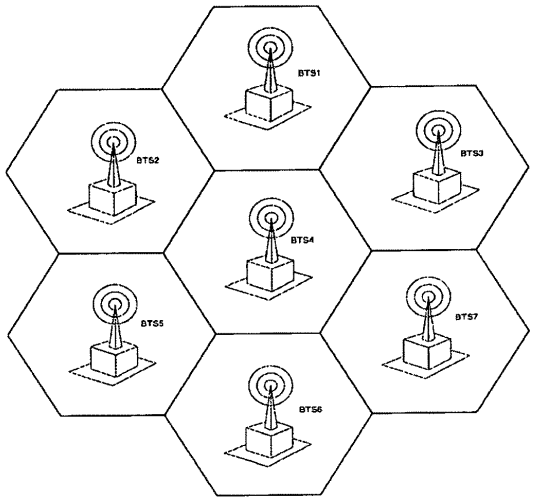
Por otro lado, el contenido de la práctica es suficientemente significativo y funcional para los alumnos, dado el enorme auge de la telefonía móvil, que hoy en día es accesible y familiar a todo el mundo. De esta forma se ha pretendido aumentar la motivación de los alumnos, y que al mismo tiempo puedan ir descubriendo las conexiones que existen entre las distintas materias del plan de estudios. Debido a que las comunicaciones móviles no están incluidas en la currícula del primer curso de la titulación, ha sido necesario proporcionar a los alumnos una pequeña introducción sobre los conceptos fundamentales que se tratan en la práctica sobre los sistemas de telefonía móvil celular, parte de la cual se expone a continuación (una descripción detallada del funcionamiento de un sistema de telefonía móvil celular se puede encontrar por ejemplo en [Hernando98]).

2.- ELEMENTOS BÁSICOS DE UN STMC

En un sistema de telefonía móvil celular se considera que el territorio está dividido en zonas de cobertura o *células*, y en el centro de cada una de ellas existe una *estación base* o BTS (*Base Transceiver Station*), que es una estación transmisora/receptora que controla a todos los teléfonos móviles que se encuentran dentro de esa célula. Las estaciones base suelen estar situadas en lo alto de los edificios, colinas, etc., y a través de ellas, los móviles se comunican con el resto del sistema telefónico.

Una célula puede verse como un mapa topográfico de potencias. En el centro de la célula, donde se encuentra la estación base o BTS, el nivel de potencia medido (radiado por la BTS) será máximo, mientras que a medida que nos alejamos del centro y nos acercamos a las fronteras de la célula o zona de cobertura, la potencia irá disminuyendo. Por simplicidad, las células suelen dibujarse con forma hexagonal. Es necesario tener en cuenta que todas las células no tienen por qué tener el mismo tamaño ni la misma geometría.

Cada BTS mantiene una *base de datos* con todos los móviles que en ese momento se encuentran bajo su jurisdicción, es decir, dentro de su *zona de cobertura*. La BTS está continuamente analizando el nivel de potencia que recibe de los móviles. De la misma forma, los móviles analizan el nivel de potencia que reciben de la BTS.



Cuando un móvil se desplaza (por ejemplo, una persona habla por su móvil mientras viaja en coche) y se acerca al borde de su célula o zona de cobertura, al caer la potencia recibida por debajo de un cierto límite o *valor umbral*, esto será indicativo de que el móvil está saliendo de su célula y está entrando en una nueva; por tanto, el móvil se desconecta de la que hasta ahora era su BTS y pasa a estar bajo el control de la BTS situada en la célula en la que ha entrado. Cuando un móvil abandona una célula, su número se borra de la base de datos de la BTS de esa célula. De la misma forma, cuando un móvil entra en una célula, su número se añade a la base de datos de la BTS de esa célula. A este proceso de paso de un móvil de una célula a otra se le denomina *handover* o *handoff*

Asimismo, cuando un móvil llama a otro móvil, su BTS local comprueba si el teléfono destino también se encuentra bajo su jurisdicción, y si es así le conecta directamente. En caso contrario, la BTS "pregunta" a sus BTS vecinas si el móvil destino se encuentra bajo el control de alguna de ellas. Una vez localizado éste, se establece la conexión entre los móviles a través de las estaciones base.

3.- SIMULACIÓN DE UN STMC

A continuación se describe la secuencia de prácticas planteadas, haciendo hincapié en cómo se ha integrado en cada uno de los bloques de contenidos los aspectos del sistema de telefonía móvil celular que se pretenden simular.

a) Estructuras de datos estáticas

La primera práctica con la que se ha introducido el sistema de telefonía móvil celular que se pretende simular es la correspondiente a estructuras de datos estáticas. El objetivo de esta práctica es que el alumno comprenda la utilización de los registros, matrices y vectores a través del problema real que queremos simular. Para representar el mapa de cobertura se emplea una matriz rectangular de valores reales (a la que se ha llamado TERRITORIO). Como su nombre indica, esta matriz representa un cierto territorio, y cada una de sus casillas una determinada

subzona. El número real almacenado en cada una de las casillas indica el nivel de potencia medio (en vatios) radiado por la BTS en esa subzona. Los elementos de la matriz con un valor máximo corresponderán a una casilla o subzona donde se encuentra una BTS, es decir, al centro de una célula o zona de cobertura (para una mayor simplicidad del problema se ha hecho la suposición de que todas las células tienen forma cuadrada y que los límites de cada célula se alcanzan cuando el nivel de potencia es inferior a 1.0 vatio). Para evitar a los alumnos la tediosa tarea de tener que asignar esos valores a la matriz elemento a elemento, se proporciona una biblioteca externa realizada a tal efecto que contiene una serie de subrutinas que inician las estructuras de datos a los valores en cuestión.

Además de la matriz anterior, para almacenar la información relativa a cada estación base o BTS se emplea un vector de registros (al que se ha llamado ESTACIONES_BASE) donde cada registro almacena la siguiente información de cada BTS: nombre (cadena) que se asigna a la BTS, coordenadas de su ubicación en la matriz TERRITORIO, coordenadas de la esquina superior izquierda y de la esquina inferior derecha de la zona de cobertura correspondiente en la matriz TERRITORIO, y un vector de cadenas, que contiene la información de todos los teléfonos que en un momento dado están bajo la jurisdicción de esa BTS.

Sobre las dos estructuras anteriores (matriz TERRITORIO y vector ESTACIONES_BASE) se propone al alumno realizar una serie de actividades que implican el manejo de estructuras de control para recorrer, acceder a los elementos y realizar búsquedas secuenciales en dichas estructuras de datos (por ejemplo, dado un cierto camino o trayecto de un móvil a lo largo de la matriz TERRITORIO, indicar el nombre de las BTS de las células por las que pasa).

b) Conjuntos dimensionados

Para que el alumno trabaje los métodos de búsqueda, ordenación, inserción ordenada, desplazamientos y recorridos en matrices y vectores se proponen distintas tareas a realizar sobre las estructuras de datos anteriores (matriz TERRITORIO y vector ESTACIONES_BASE). Por ejemplo, se propone calcular el nivel de potencia medio que existe dentro de cada zona de cobertura del sistema de telefonía, indicando cuál es la BTS correspondiente a la zona de cobertura con mayor potencia media. Otra tarea propuesta consiste en, dado un cierto número de teléfono, localizar la BTS de la zona de cobertura donde se encuentra. Para simular el proceso que tiene lugar cuando un móvil se desplaza de una zona de cobertura a otra diferente, es necesario realizar una eliminación del teléfono en cuestión en el vector de teléfonos de la BTS origen, e insertarlo en el vector de teléfonos de la BTS destino.

c) Cadenas de caracteres

Hasta esta práctica se han utilizado cadenas de caracteres como un tipo de datos que almacena una ristra de caracteres, pero no se utilizan las funciones específicas que se definen sobre ellas. Las cadenas de caracteres son empleadas en este caso para representar los números de teléfono origen y destino entre los cuales se va a realizar una llamada. También se hace uso de las cadenas para representar los mensajes cortos que van a ser transmitidos a través de los teléfonos móviles del sistema, o bien las secuencias de teclas que presionadas sobre el teclado del móvil dan lugar a dichos mensajes. Así, uno de los ejercicios iniciales que se proponen consiste en, dado un número de teléfono móvil, detectar si el formato de dicho número es correcto y al mismo tiempo indicar el nombre del operador de telefonía móvil al que corresponde (proporcionando al alumno cuáles son los prefijos que actualmente emplean cada uno de los operadores de telefonía móvil existentes en España).

Para aproximarnos de forma progresiva a la situación real con la que nos encontramos al transmitir mensajes cortos con un teléfono móvil, inicialmente se plantea representar las secuencias de pulsaciones separando los grupos de pulsaciones correspondientes a caracteres diferentes mediante el carácter '_' (así, por ejemplo, la cadena '8_33_1_33_7777_7_33_777_666' es la secuencia que es necesario teclear en el móvil para transmitir el mensaje 'TE ESPERO'). Posteriormente se proponen distintas modificaciones a esta representación, como por ejemplo eliminar el carácter separador '_' para delimitar cada grupo de pulsaciones.

d) Archivos de texto y con tipo

Las siguientes estructuras de datos que se contemplan en el módulo de prácticas son los archivos de texto y con tipo. Para poder usar y tratar archivos de texto, se opta por almacenar en cada una de las líneas de un archivo de texto la información básica referente a una llamada realizada en el sistema de telefonía móvil, siendo su formato: '*número_teléfono_origen mensaje_transmitido número_teléfono_destino*'. De esta forma se combina el uso de archivos de texto con el tratamiento de cadenas de caracteres, proponiendo tareas como la obtención de las secuencias de pulsaciones correspondientes a los mensajes transmitidos, etc.

Para introducir el uso de archivos con tipo o binarios, se ha optado por almacenar la información referente a las llamadas realizadas en forma de registros, cuyos campos almacenan el número de teléfono origen, el número de teléfono destino y el número de caracteres del mensaje transmitido. Una vez creado el archivo binario, se proponen tareas como generar la factura correspondiente a las llamadas realizadas por un cierto teléfono origen.

e) Estructuras dinámicas: listas enlazadas y grafos

El uso de las listas enlazadas y grafos es enfocado haciendo énfasis en su propiedad de poder 'crecer' o 'menguar' conforme se necesite a lo largo de la ejecución de un programa. Para ello, en lugar de usar la estructura estática inicial del vector ESTACIONES_BASE, donde el número de teléfonos que una estación base podía controlar estaba limitado a un cierto valor máximo, se propone la utilización de una lista enlazada para almacenar la información de todos los teléfonos que en un momento dado están bajo la jurisdicción de cada BTS. Se introducen entonces aspectos como la eliminación e inserción en listas enlazadas cuando se intenta simular el paso de un móvil de una zona de cobertura a otra.

4.- CONCLUSIONES

El artículo presenta un modelo de prácticas para las asignaturas Algorítmica y Programación encaminado a incorporar temas específicos del ámbito de las telecomunicaciones, al mismo tiempo que se desarrolla el contenido disciplinar propio de las asignaturas. Con esto se pretende aumentar la motivación del alumnado y hacer evidente la importancia de las asignaturas en estos estudios de ingeniería. Al mismo tiempo, el problema que se plantea, además de su carácter práctico, intenta tener un carácter global y continuo que permite hacer ver al alumno el uso del software como un conjunto de módulos que posibilitan la resolución de problemas complejos o de grandes dimensiones.

5.- REFERENCIAS

[Martel96] E.A. Martel Jordán, C.N. Ojeda Guerra, C.M. Ramírez Casañas, M.A. Quintana Suárez. *Algorítmica. Fundamentos de los Computadores. Algoritmos y Estructuras de Datos, 4ª Edición.* Servicio de Reprografía de la Universidad de Las Palmas de G.C., 1996.

[Ojeda97] C.N. Ojeda Guerra, E.A. Martel Jordán, C.M. Ramírez Casañas, M.A. Quintana Suárez. *Programación. Estructuras de Datos Dinámicas, 3ª Edición.* Servicio de Reprografía de la Universidad de Las Palmas de G.C., 1996.

[Hernando98] J.M. Hernando Rábanos. *Transmisión por Radio, 3ª Edición.* Departamento de Publicaciones de la ETSIT de Madrid, 1998.

[Ramírez96] C. Ramírez Casañas, M.A. Quintana Suárez, C.N. Ojeda Guerra, E.A. Martel Jordán. *Modelo Alternativo de Prácticas para las Asignaturas de Algorítmica y Programación.* II Jornadas de Innovación Docente ICE 96. Madrid, 1996.

LA ASIGNATURA REDES DE ÁREA LOCAL EN LA INGENIERÍA INFORMÁTICA DE LA UPV: TEORÍA Y PRÁCTICAS

Juan Carlos Cano Escriba[1], Julio Sahuquillo Borrás[1], José Flich Cardo[1] y José Antonio Gil Salinas[1]

[1]*Departamento de Informática de Sistemas y Computadores*
E. U. I.–Universidad Politécnica de Valencia–
e_mail: {jucano, jsahuqui, jflich, jagil} @ disca.upv.es

RESUMEN: La asignatura de Redes de Área Local (RAL) se imparte con carácter optativo en el noveno semestre de la titulación de Ingeniero en Informática en la Universidad Politécnica de Valencia.

En el presente artículo se esbozan los contenidos temáticos de esta asignatura, así como las prácticas y trabajos adicionales que se proponen al alumno con el fin consolidar su formación ingenieril.

Asimismo, se describe la metodología docente que se utiliza para incentivar y mejorar el proceso de enseñanza-aprendizaje del estudiante. Por último, se describe el proceso ineludible de evaluación del alumno.

1.- INTRODUCCIÓN.

Los estudios de Ingeniero en Informática (II) de la Universidad Politécnica de Valencia (UPV), se estructuran de forma cíclica, con un total de 375 créditos organizados en dos ciclos, de 6 y 4 semestres donde el alumno debe cursar 225 y 150 créditos respectivamente. Las asignaturas cursadas se clasifican en obligatorias, optativas y de libre configuración. Las asignaturas obligatorias, incluyen tanto materias troncales de los títulos, como materias, que cada universidad, haciendo uso de su autonomía establece en el momento de la elaboración del plan de estudios. En cuanto a las asignaturas optativas, su principal objetivo es permitir una formación específica en determinadas áreas. Los alumnos pueden optar a las especialidades de (a) Gestión, (b) Ingeniería de los computadores y (c) Software. Finalmente, los créditos pertenecientes a las asignaturas de libre configuración, incluyen asignaturas y actividades orientadas a la formación integral del alumno, ofertadas por los centros y departamentos, y reconocidas por la Universidad.

Para obtener el título, el alumno debe superar todas las asignaturas obligatorias, además de un número de créditos correspondientes a materias optativas y de libre configuración.
La estructura de la propuesta curricular establecida en la titulación, oferta la mayoría de asignaturas obligatorias en los dos primeros y cuarto curso, mientras que en tercero y quinto se ofertan la mayor parte de asignaturas optativas. Las asignaturas optativas ofertadas en el tercer

curso, orientan la formación del alumno hacia la especialidad que deseen cursar en el segundo ciclo. Por su parte, la oferta de asignaturas optativas de quinto curso consolidan la formación específica en función de la especialidad elegida. Las asignaturas optativas del último curso se agrupan en bloques temáticos o intensificaciones con el fin de orientar la elección del alumno.

El área de conocimiento de la Ingeniería Telemática se estructura alrededor de dos asignaturas troncales impartidas en cuarto curso: (i) Redes de Computadores I (RCI) y (ii) Redes de Computadores II (RC2), impartidas en el séptimo y octavo semestre respectivamente. La asignatura RC1 (3 créditos de teoría + 1,5 créditos de prácticas), introduce los contenidos básicos de un sistema telemático completo, presentando los niveles de referencia ISO, y haciendo especial énfasis en técnicas de transmisión y protocolos de enlace. Por otra parte, debido al crecimiento y explosión del uso de Internet en los últimos años, la asignatura RC2 (3 créditos de teoría + 1,5 créditos de prácticas), presenta detalladamente el conjunto de protocolos y conceptos que sustentan Internet, así como las aplicaciones utilizadas sobre dicha red.

El amplio campo y la creciente evolución que constituyen las comunicaciones y las redes de computadores, conforman cada vez más un pilar fundamental en la formación de los futuros Ingenieros en Informática. En este sentido, de cara a consolidar la formación integral del alumno en el área de conocimientos telemáticos, se ofertan un conjunto de asignaturas optativas en quinto curso, que complementar y extienden aspectos teórico/prácticos impartidos en las asignaturas troncales RC1 y RC2.

En este trabajo, se presenta la guía y metodología utilizada por los autores en la asignatura Redes de Área Local (RAL), que con un total de 6 créditos asignados (3 créditos de teoría y 3 créditos de prácticas), se oferta como asignatura optativa en noveno semestre de la titulación. La asignatura RAL, profundiza en los aspectos más significativos de las cada vez más utilizadas comunicaciones en entornos locales, haciendo especial énfasis en los estándares más recientes sobre redes locales, así como en aspectos teórico-prácticos relacionados con la instalación, la administración y el mantenimiento de dichas redes.

2.- OBJETIVOS.

a) Objetivos generales

- Introducir los conceptos fundamentales sobre la estructura; la organización y el funcionamiento de las redes de área local, haciendo especial énfasis en las tecnologías emergentes y de alta velocidad más recientes.

- Adquirir experiencia en la instalación, utilización, gestión y diseño de las redes locales utilizando sistemas operativos en red comerciales.

b) Objetivos específicos

- Analizar las ventajas e inconvenientes de los diferentes protocolos de acceso al medio utilizados en redes locales, y su evolución hacia soluciones de alta velocidad.

- Analizar la necesidad y utilización de normas de cableado estructurado locales, presentando detalladamente la normativa 568 de la Asociación de Industrias Electrónicas/Asociación de Industrias de Telecomunicaciones (EIA/TIA).

- Conocer las arquitecturas más significativas utilizadas en las redes de área local.
- Conocer los estándares sobre redes de área local, haciendo especial énfasis en las últimas tendencias por ejemplo Fast Ethernet, Gigabit Ethernet y redes inalámbricas.
- Analizar la problemática y necesidad de interconectar redes, presentando posibles soluciones tanto a nivel arquitectónico como de dispositivos.
- Introducir al alumno en los mecanismos y técnicas de gestión de redes.
- Estudiar técnicas de conectividad entre sistemas operativos en red.
- Utilizar y configurar herramientas que permiten la interoperabilidad de sistemas en entornos locales.
- Introducir los sistemas operativos en red Novell, Windows NT y Linux.
- Estudiar diferentes Interfaces de Programación de Aplicaciones (API's) como herramientas de acceso a los servicios de red y utilizadas para desarrollar aplicaciones distribuidas cliente/servidor.
- Explorar y analizar las posibilidades de las Intranets, como aplicación directa de la TCP/IP a entornos privados locales.

Como objetivo adicional, y debido principalmente al carácter terminal de la asignatura, se pretende potenciar las habilidades escritas y orales del alumno como herramientas fundamentales en su perfil profesional.

3.- TEMARIO.

a) Contenido teórico.

Para conseguir los objetivos teóricos anteriores, se proponen los contenidos temáticos especificados en tabla 1.

Programa teórico de Redes de Área Local	Duración (clases 2h)
Tema 1. Introducción • Características de las redes locales: Medios y Técnicas de transmisión, Topologías y Control de acceso al Medio. • Proyecto 802.x: Estandares IEEE 802.x • Normativa EIA/TIA 568 para redes locales	1,5
Tema 2. Redes de Área Local tradicionales • Estándar 802.3 CSMA/CD (Ethernet) • Estándar 802.5 Token-Ring	2,5
Tema 3. Redes de Área Local Inalámbricas • Introducción • Aplicaciones y requisitos de las redes inalambricas • Estándar 802.11, para redes inalambricas	1,5

Tema 4. Redes de Área Local de alta velocidad • Estándar 802.3u Fast-Ethernet • Gigabit Ethernet • Estándar 802.12 100VG-AnyLAN • Estándar ANSI X3T9.5 FDDI	3,5
Tema 5. Capa de Enlace Lógico de datos (LLC) • Estandar 802.2: Funciones y especificaciones de servicio • Interfaces de Enlace de datos NDIS y ODI	1
Tema 6. Interconexión de redes locales • Supuestos prácticos	1
Tema 7. Gestión de red • Protocolo SNMP	1
Presentación trabajos de asignatura	2
Total	14

Tabla 1. Distribución de los contenidos teóricos de RAL

Aunque para la elaboración del temario se suele considerar un curso académico de 15 semanas, en la práctica, y debido a días festivos así como problemas de sincronización, se ha propuesto un temario en base a un curso de 14 semanas. Como se puede observar, las dos últimas semanas del curso se reservan para la presentación de trabajos de asignatura cuyos objetivos principales se comentarán posteriormente.

b) Contenido Práctico.

Debido al número de créditos asignados (3 créditos) a la parte teórica de la asignatura, así como a la utilización de parte de los mismos (0,4 créditos) a la exposición de los trabajos realizados por parte de los alumnos, se ha considerado adecuado, trasladar a las prácticas aquellos conceptos susceptibles de ser tratados y asimilados de forma experimental. En este sentido las prácticas realizadas intentan cubrir objetivos relacionados con (i) utilización de las API's para diseñar aplicaciones en red (ii) herramientas de interoperabilidad de sistemas operativos en red y (iii) administración y gestión de redes locales.

La tabla 2 muestra las prácticas planificadas de cara a cubrir los objetivos anteriores. Como se puede observar, se han planificado un total de 5 prácticas que utilizan 6 sesiones de prácticas, reservando otras 6 sesiones para que los alumnos realicen el trabajo de asignatura con la ayuda y orientación del profesor. Las sesiones de trabajo y las prácticas planificadas, se realizan en sesiones alternas. Al igual que sucede con la planificación de teoría, se han reservado un total de tres sesiones de prácticas para la presentación de los trabajos realizados.

Prácticas de Redes de Área Local	Duración (clases 2h)
Práctica 1: Introducción a Novell Netware	1
Práctica 2: Utilidades de red en sistemas Windows (98, NT)	1
Práctica 3: Arquitectura servidor X: Herramienta Xwin32	1
Práctica 4: Sistemas de ficheros en red: NFS, Samba	1
Práctica 5: Programación cliente/servidor en RAL's: API's	2
Desarrollo del trabajo de asignatura	6
Presentación trabajos de asignatura	2
Total	14

Tabla 2. Distribución de los contenidos prácticos de RAL

Con el principal objetivo de evitar la carga desmedida del alumno, se realiza una planificación de las prácticas de 2 horas de duración cada una, basadas cada una en un cuadernillo detallado y con intervalos cerrados, de forma que tanto el alumno como el profesor conozcan en todo momento el inicio y final de las mismas. La metodología de evaluación utilizada se detalla en el punto 5.

b) Trabajos Propuestos.

Cada día que pasa acrecienta la necesidad de acercar la universidad al mercado laboral. Además, gran parte de los alumnos que cursa una asignatura de estas características sueña con encontrar un trabajo de estas dimensiones. Por tanto, y en la medida de nuestras posibilidades nos encontramos con la obligación de reducir esta distancia existente; más aún si se considera el carácter terminal de la asignatura. Para reducir esta distancia, es de vital importancia que la orientación de las prácticas y trabajos se encuentren relacionadas con la demanda de trabajo actual de forma que despierten y aviven el sueño del estudiante. Por otra parte, es conveniente fomentar un entorno de trabajo que no se encuentre excesivamente distante del entorno profesional, donde el trabajo en grupo y la presentación tanto escrita como oral de proyectos de ciertas dimensiones es la norma habitual.

Debido a los motivos citados, una parte importante de la asignatura se destina a la realización de un trabajo, directamente relacionado con demandas laborales actuales o bien con tecnologías emergentes. Los trabajos se realizar en grupos de 3 o 4 alumnos.

Trabajos ofertados	Trabajos realizados
Instalación y configuración de sistemas operativos en red: • Novell 5.0 y Windows 2000	3
Análisis teórico de tecnologías de red: • ADSL, Myrinet, GSM, 100VGAnyLAN	4
Configuración y administración de Intranets • IIS4.0 sobre NT Server, Apache sobre Linux y NT Server	4
Software de Gestión de red • SNMP, Analizador Observer	3
Seguridad e Interconexión • Sistemas Proxy: Proxy server NT, Squid Linux, Winproxy Win98	4
Administración de redes locales: supuestos prácticos • NT Server, Linux	3

Tabla 3. Relación de trabajos ofertados y realizados e el curso académico 1999/2000

La tabla 3 muestra los trabajos ofertados y realizados en el año académico 1999/2000. Como se puede observar, con miras a fomentar la comunicación no solamente en grupo, sino además entre-grupos, no hay inconveniente en que un trabajo se realice por varios grupos, incluso puede ser de interés el intercambio de ideas y planteamientos de cara a mejorar la calidad final de los mismos. Cada uno de los trabajos realizados, se presenta en clase al resto de alumnos y profesores de la asignatura, y se evalúa en base tanto a contenidos del mismo como a la presentación oral individual y escrita que se realice del mismo. Finalmente se desea comentar, que al final del curso, y de cara a que todos los alumnos conserven una fuente bibliográfica de los diferentes trabajos realizados, se realiza una copia en CD ROM de la página Web de la asignatura, que contiene toda la información de la misma, además de las memorias y presentaciones en formato digital (Power Point) de cada uno de los trabajos realizados.

4.- METODOLOGÍA.

En cuanto a las clases de teoría, la metodología se basa principalmente en clases magistrales, apoyadas por transparencias desarrolladas con la herramienta PowerPoint y utilizando el cañón de proyección. Esta herramienta, en opinión de los autores refuerza la clase magistral, captando la atención del alumno mediante la exposición de imágenes, y permitiendo una fácil secuenciación y exposición ordenada de los contenidos. En cuanto a las prácticas, se basan en la utilización de cuadernillos didácticos. Al igual que sucede con las clases teóricas, el cañón de proyección permite exponer de forma dinámica tanto conceptos teóricos como aspectos prácticos relacionados con la práctica. Todo el material utilizado, tanto en clases teóricas como en prácticas se encuentra disponible para el alumno en formato electrónico en la dirección Web de la asignatura: http://www.redes.upv.es/ralfi/.

Para las presentaciones de trabajos, cada grupo dispone de unos 15 minutos de exposición repartidos entre los miembros del grupo, en los cuales se debe hacer especial énfasis en el trabajo realizado y conclusiones obtenidas a lo largo del trabajo, seguidos de un turno de discusión. Debido a la naturaleza que presenta la asignatura, siempre existen temas prácticos de considerable importancia e interés candente que suelen salir varias veces a debate, ya sea de motu propio del alumno o bien introducidos intencionadamente por el profesor.

Con el fin de motivar la asistencia del alumno a la presentación de los trabajos de otros compañeros, se les informa que aquellos conceptos que salen a relucir en varias exposiciones serán exigibles en el examen teórico final.

5.- EVALUACIÓN.

La evaluación de la asignatura, se realiza a partir de una prueba escrita que tiene un peso del 60 % sobre la nota final, y que resume los diferentes aspectos tratados a lo largo del curso. Esta prueba incluye cuestiones relacionadas con: (i) contenidos teóricos tratados en clase, (ii) contenidos de prácticas y (iii) trabajos expuestos en clase. El 40 % restante, se obtiene a partir del trabajo de asignatura realizado. Como es sabido, la evaluación de un trabajo que se ha realizado en grupo siempre es difícil. Si sólo se evalúa el trabajo en base a la memoria, puede darse el caso de que algunos alumnos avispados o aprovechados, según como se mire, se puedan beneficiar. Para evitar esta situación, el porcentaje asignado al trabajo se obtiene a partir de la media aritmética de los siguientes valores: (i) contenido del trabajo, (ii) presentación oral (individual) (iii) presentación escrita. De esta forma, la presentación individual y el debate posterior, aunque no evita, ayuda a reducir el número de situaciones injustas.

6.- BIBLIOGRAFÍA PROPUESTA.

La bibliografía utilizada para la organización de la asignatura es:

- Local and Metropolitan Area Networks, Ed. Macmillan, W. Stallings
- Redes para Proceso Distribuido, Ed. RA-MA, Jesús García Tomás, Santiago Ferrando, Mario Piattini
- Redes de Computadores, Ed.Engelwood Cliffs, A.S.Tanenbaum
- Interconections: Bridges & Routers., Ed: MacMillan, 1995, R. Perlman
- Estandares IEEE 802.x. sobre redes de área local

7.- CONCLUSIONES.

En este trabajo se han presentado los objetivos que persigue la asignatura Redes de Área Local, como parte de la formación específica de los futuros Ingenieros Informáticos de la Universidad Politécnica de Valencia. Se han remarcado tanto los contenidos como la metodología docente, así como los métodos de evaluación utilizados.

Los autores, en base a su experiencia, fomentan la utilización de trabajos prácticos en la asignatura, ya que ésta evoluciona y cambia año tras año. De esta manera se completa la formación del alumno, potenciando su iniciativa, razonamiento y capacidad de síntesis al tiempo que fomenta sus capacidades de trabajo en grupo y ejercita sus aptitudes orales y escritas.

UNA ALTERNATIVA PARA MOTIVAR A LOS ALUMNOS EN LA ENSEÑANZA Y APLICACIÓN DE LAS REDES DE ORDENADORES

Lenin Lemus, Esperanza Esteso

E.U.I (Escuela Universitaria de Informática de la Universidad Politécnica de Valencia)
e-mail: lemus@disca.upv.es, eesteso@disca.upv.es

RESUMEN: La asignatura de Laboratorio de Redes (***LAR***) fue impartida por primera vez en el segundo semestre del curso escolar 97-98 en la Escuela Universitaria de Informática (EUI) de la Universidad Politécnica de Valencia (UPV), desde sus inicios, el objetivo principal ha sido que los alumnos del último curso de la titulación Ingeniero Técnico en Informática tanto de gestión como (ITIG) de sistemas (ITIS) desarrollen aplicaciones que ejerciten los conocimientos aprendidos en los dos primeros años de la titulación.

En esta ponencia queremos presentar un método de mejorar la enseñanza en la asignatura *LAR* en base a la experiencia ya adquirida, la organización de la ponencia es la siguiente: primero se muestra la organización actual de la asignatura y las deficiencias encontradas.

A continuación se proponen cambios en la organización para solventar las deficiencias. Finalmente se presentan las conclusiones.

1.- ORGANIZACIÓN ACTUAL DE LA ASIGNATURA

La asignatura *LAR*, es una asignatura cuatrimestral optativa del tercer curso de las titulaciones Ingeniero Técnico en Informática de Sistemas (ITIS), e Ingeniero Técnico en Informática de Gestión (ITIG), de la Univesidad Politécnica de Valencia (UPV). Se imparte en el segundo semestre del año escolar.

Su principal objetivo es que los alumnos implementen aplicaciones basadas en redes, la elección de la aplicación a desarrollar es libre, la única condición es que esté relacionado con las redes de computadores.

Está repartida en seis créditos, tres de los cuales son dedicados a laboratorio, 1.5 a atender las necesidades de los alumnos para desarrollar la aplicación que eligen para implementar y 1.5 créditos son dedicados a la parte teórica.

En lo que respecta a las clases teóricas, éstas se imparten mediante lección magistral, con la particularidad de que el contenido de las clases son solicitadas por el alumnado. Estas peticiones

sirven para profundizar en conceptos que los alumnos no tienen lo suficientemente claros y que necesitan para llevar a cabo la aplicación que están desarrollando.

La parte de créditos correspondiente a la resolución de problemas, se utiliza para asesorar a los alumnos para la planificación, selección de herramientas e implementación de su trabajo.

Los alumnos se agrupan en equipos de cuatro a ocho personas, dependiendo de la aplicación a desarrollar., el semestre consta de trece semanas, las cuatro primeras se dedican a tutoriales sobre las herramientas y lenguajes a utilizar, así como la implementación de las partes más básicas de sus aplicaciones (comunicaciones, interfaces de usuario...).

Las dos semanas siguientes es responsabilidad de los alumnos formar equipos y elegir una aplicación a desarrollar, una vez hecho esto, los alumnos deben presentar al profesorado el análisis de requerimientos, la planificación de tareas y el responsable de cada una de ellas.

La fecha de entrega de trabajos es la que fija la EUI para presentar el examen de *LAR*, al finalizar el semestre.

La asignatura se califica en base a los criterios mostrados en la tabla siguiente:

	Nota Máxima
Aplicación Operativa	5
Documentación	2
Extensiones	2
Presentación Oral	1

Tabla 1 Criterios para obtener la nota final.

El tipo de aplicaciones que se realizan son:

- Comercio Electrónico: Tiendas Virtuales.
- Protocolos de Comunicaciones: Servidores FTP, Servidores POP3, Servidores SMTP
- Aplicaciones cliente/servidor: Utilizando RMI, CORBA o Enchufes
- Gestión de redes: SNMP

Los trabajos realizados en los dos últimos años se presentan en las tablas 2 y 3.

Tipo de Aplicación	Nombre	S.O.
Tienda Virtual	LAR's COMP	Win98
Tienda Virtual	FORTUNA	Win98
Chat	MEGACHAT	Win98
Chat	TURBOCHAT	Win98
Visor de la bolsa	BOLSA NET	Win98
Visor de la bolsa	Acciones / Java	Win98
Buscador WEB	COYOTE	Win98
Servidor WEB para adq. De datos	HS-WEB	Win98
Servidor de Correo	Servidor de correo	Win98
Buscador Inteligente	BUSKA	Win98
Ejecución remota de scripts	SULTAN	Win98
Ejecución remota de scripts	control remoto PC	Win98
Aplicación para subastas Inmobiliarias	Subastas inmobiliarias	Win98
Servidor FTP	FTP inteligente	Win98
Navegador fuera de Línea	CERCANET	Win98
Navegador WEB	Netscape	Linux
Navegador fuera de Línea	Browser off -line	Win98

Tabla 2 Trabajos realizados en el curso 1998-1999

Tipo de Aplicación	Nombre	S.O.
Tienda Virtual	EShop	Win98
Tienda Virtual	Carbury	Win98
Tienda Virtual		Win98
Tienda Virtual		Win98
Tienda Virtual		Win98
Tienda Virtual		Win98
Chat	Chat World	Win98
Chat		Win98
Servidor de Correo		Win98
Servidor gestor de Correo		Win98
Gestión SNMP		Win98
FTP Inteligente		Win98
Cliente de Correo		Win98
Browser Off Line		Win98
Generador de Tablas		Win98

Tabla 3 Trabajos realizados en el curso 1999-2000

En el curso lectivo 1999-2000 se han desarrollado:

- Seis tiendas Virtuales.
- Dos servidores de correo.
- Dos chat.
- Una aplicaciones FTP Inteligente
- Un agente SNMP
- Un Generador de Tablas (HTML)
- Un Browser off-line

- Un cliente de correo

Las notas obtenidas en los últimos dos años se muestran en las tablas 4 y 5, la tabla 6 muestra la nota media de los dos últimos cursos.

No a	Nú nero de Ali mnos	% Presentados	% / Total
Sobresalientes	3	4.35	2.73
Notables	60	86.96	54.55
Aprobados	5	7.25	4.55
Total de Aptos	68	98.55	61.82
No Aptos	1	1.45	0.91

Tabla 4 Notas obtenidas en el curso 1998-1999

No a	Nú nero de Ali mnos	% Presentados	% / Total
Sobresalientes	8	14.81	6.72
Notables	46	85.19	38.6
Aprobados	0	0	0
Total de Aptos	54	100	45.38
No Aptos	0	0	0

Tabla 5 Notas obtenidas en la convocatoria de junio del curso 1999-2000

Nota: Falta agregar las estadísticas de la segunda convocatoria del curso 1999-2000

Curso	Nota Media	Desvi ación estándar
1998-1999	7.0	1
1999-2000	7.87	0.79

Tabla 6 Nota media de los dos últimos cursos.

2.- PROBLEMÁTICA DE LA ORGANIZACIÓN ACTUAL.

La experiencia obtenida en los tres cursos en los que se ha impartido la asignatura ha resultado muy positiva, tanto para los alumnos como para los profesores.

En *LAR* los alumnos se enfrentan por primera vez en la carrera, al desarrollo de un proyecto con una complejidad y extensión superiores a lo que están acostumbrados, y mucho más cercanas a lo que se van a encontrar en el mundo laboral. El objetivo es que les permita enfrentarse a

problemas reales de planificación, trabajo en grupo y selección de los métodos y herramientas más adecuados para su proyecto.

Por la propia definición de la asignatura los profesores que la impartimos tenemos la enriquecedora experiencia de poder acercar al alumno a lo que va a ser el desarrollo de proyectos que muy pronto se van a encontrar en el mundo laboral.

Es por tanto, una asignatura con una valoración muy positiva tanto por parte de los profesores como de los alumnos y, es por esto, hemos decidido recoger la experiencia de sus dos cursos de vida y ver de qué forma podemos mejorarla para sacar el máximo partido de ella.

Entre las cosas que hemos observado hay que destacar que el hecho de dedicar cuatro semanas a tutoriales más dos para la selección del trabajo antes de empezar con el mismo parece que hace que el alumno no preste tanto interés a esos tutoriales iniciales. Pensamos que sacarían mucho más provecho de ellos si ya tuviesen el trabajo decidido e intentasen ver en que parte del mismo les son de utilidad.

El hecho, también, de empezar el trabajo seis semanas después de comienzo de curso hace que aunque al principio todos van bastante relajados, conforme se acerca el final del curso a casi todos los grupos se les echa el tiempo encima.

Debido, probablemente, a que no están habituados a hacerlo, les cuesta mucho desarrollar las fases iniciales del proyecto (planificación, requerimientos...), y luego se nota esta carencia en el proceso de implementación y depuración de su trabajo. Esto no implica que, al final, se obtengan trabajos de calidad, pero que les ha costado mucho más de desarrollar y los que, a falta de la documentación necesaria, serían mucho más difíciles de retomar y modificar pasado un cierto tiempo, tarea que resultará sumamente importante en su vida laboral.

En general, todos los grupos que han trabajado con Visual Basic se quejan de los problemas que han tenido a la hora de desarrollar la parte de comunicaciones, que es a la que mayor importancia se le da en esta asigatura debido a la filosofía de la misma. Se intenta que los alumnos no trabajen directamente con Active-X sino que ellos mismos sean capaces de realizarlos y programar directamente las funciones de acceso a la red. Es por esto, que el hecho de utilizar Visual Basic para tales desarrollos resulta más complicado.

Parece indicativo, también, que la relación entre el número de alumnos matriculados y los que se presentan y aprueban la asignatura es muy similar en todos sus años de vida. Este año el número de alumnos matriculados ha sido superior a los anteriores, lo que nos indica un interés creciente de los alumnos por esta asignatura. Es por la diferencia entre el número de alumnos matriculados y los que van realizando su trabajo día a día y se presentan en junio, por lo que es importante cuidar que los alumnos que se dejan la asignatura para septiembre no aprueben copiando trabajos de compañeros que la han pasado en junio.

3.- METODOLOGÍA PROPUESTA

En base a la experiencia obtenida en la impartición de la asignatura *LAR* y teniendo en cuenta las carencias observadas en los alumnos, pensamos que para el próximo curso escolar (2000-2001), la elección del trabajo a realizar se hará antes de comenzar los tutoriales, es decir, durante las dos primeras semanas del curso y después de introducir y presentar los objetivos de la asignatura adecuadamente.

Es por esto que las primeras clases serán muy importantes y habrá que planificarlas con sumo cuidado para que los alumnos reciban toda la información que puedan necesitar y se les asesore adecuadamente para que seleccionen su trabajo. Durante estas clases los alumnos deberán, también, ir creando sus grupos y consultar a los profesores en todo lo que necesiten para ello.

También para el próximo curso se dará la posibilidad de que los alumnos que trabajen con Visual Basic puedan utilizar componentes Active-X ya desarrollado, pero con un prejuicio en la nota respecto a los que utilicen componentes o funciones realizadas por ellos mismos utilizando las *.dlls* de Winsockets.

Se propone que el alumno tenga la posibilidad de acceder a todos los tutoriales *On-line*. Se intentará que esta información se pueda acceder de forma personalizada y puntúe a cada alumno dependiendo de su correcto aprovechamiento. Es por esto que se intentará establecer un procediento de evaluación que también sea *On-line* y las calificaciones serán, en este caso, individuales para cada alumno, suponiendo un porcentaje sobre la nota final que obtenga el grupo.

Se propone también definir la figura del profesor como la del "*cliente*" al que hay que vender el proyecto realizado. Los alumnos deben procurar que la relación con el "*cliente*" sea la más adecuada durante todas las fases de desarrollo, para ello:

- Deberán definir adecuadamente los objetivos de su aplicación y "*venderselos*" a su cliente con el que deberán "*negociar*" los requerimientos finales.
- Definir todas las fases de desarrollo de su proyecto planificando fechas de validación para cada una de ellas.
- El "*cliente*" deberá "*evaluar*" y "*aceptar*" cada fase en las fechas acordadas. Esto supondrá que el alumno va obteniendo su calificación de forma progresiva, destacando así la importancia de cada una de las fases y repartiendo más adecuadamente el tiempo disponible.

Para organizar todo esto de forma adecuada hemos decidido emplear una herramienta de planificación de proyectos como es el *Microsoft Project*. Esto introducirá a los alumnos a un nuevo tipo de herramientas que en el futuro les será de mucha utilidad. También facilitará la tarea de los profesores que tendremos que planificar adecuadamente el control de todos los proyectos los cuales habrá que seguir más de cerca en nuestra calidad de "*clientes*".

4.- COMENTARIOS FINALES

En base a la experiencia adquirida durante los tres años de vida de la asignatura y el creciente interés de los alumnos por la misma pensamos que mediante la puesta en práctica de la metodología propuesta en el punto anterior podremos mejorar y sacar mucho más provecho de ella.

Es por esto, que pensamos presentar un proyecto PIE que nos permita recoger todos los tutoriales que tengan algún interés para la asignatura (protocolos de comunicaciones, lenguajes de programación en entornos de redes, configuración de sistemas de red ...), desarrollarlos en forma de tutoriales lo más didácticos posible y hacerlos accesibles al alumno a través de la página web de la asignatura, el objetivo es que en el menor tiempo posible recuerden los detalles que interesan (Protocolo SMTP, FTP, POP3, SNMP, etcétera).

Este proyecto agrupará varias asignaturas, entre las que destacan "Sistemas Operativos", "Modelado y Evaluación de Redes", la meta es poder recoger información que consideramos de gran utilidad no solo para los ingenieros que salen al mercado laboral, sino también para los alumnos que van a cursar estas asignaturas por primera vez, ya que tendrán reunida la información detallada de temas importantes del área de conocimientos de "Arquitectura y Tecnología de Computadores".

Como simple comentario en el proyecto PIE se intentará desarrollar un sistema de evaluación por WEB, con el fin de obtener estadísticas de la calidad de los tutoriales, de su utilización y del impacto que pueden tener en las asignaturas que participan.

HERRAMIENTA PARA LA EXPERIMENTACIÓN DE LA CODIFICACIÓN, COMPRESIÓN Y TRANSMISIÓN DE VÍDEO

Jose Oliver[1], M.P. Malumbres[2]

[1]Becario F.P.I. [2]Titular de Universidad de la U.P.V.
Departamento de Informática de Sistemas y Computadores
e-mail: {joliver, mperez}@gap.upv.es

RESUMEN: Esta ponencia muestra una herramienta desarrollada para la experimentación en la codificación y transmisión de secuencias de vídeo, temática abordada en la asignatura Redes Avanzadas de la Facultad de Informática de la UPV. Además de realizar una breve descripción de la asignatura, se mostrarán las características de la herramienta así como una serie de ejercicios propuestos que van desde la propia codificación hasta los aspectos más complejos de la compresión de vídeo.

1.- INTRODUCCIÓN.

Redes Avanzadas es una asignatura que se encuentra ubicada en el último semestre de la titulación de Ingeniero en Informática de la Facultad de Informática de la Universidad Politécnica de Valencia. Su carga docente es de 6 créditos (3 de teoría y 3 de laboratorio) y su carácter es optativo. Sus contenidos están diseñados con el fin de ofrecer materias avanzadas y complementarias a las estudiadas en las asignaturas troncales de Redes de Computadores de la titulación. Por ello, es una asignatura que cada año necesita actualizar sus contenidos para mostrar las tecnologías que están saliendo al mercado o que en un futuro inmediato estarán disponibles.

La asignatura se divide en dos grandes bloques temáticos:

- El primero trata sobre aspectos avanzados de las tecnologías LAN y WAN y sus perspectivas de futuro. Aproximadamente requiere el 40% de las sesiones de teoría.

- Y el segundo, aborda el tema de redes multimedia (*Multimedia Networking*), con objeto de estudiar las peculiaridades que deben tener las redes para poder transportar adecuadamente el tráfico multimedia. Para ello, se introducen los medios continuos más habituales en las aplicaciones multimedia (audio y vídeo), la necesidad de utilizar compresores describiendo los estándares más habituales y por último los protocolos y arquitecturas propuestas para poder transportar el tráfico multimedia en Internet con una cierta calidad de servicio.

En cuanto a las sesiones de laboratorio (véase la Tabla 1), están centradas en el segundo bloque temático, ya que el primero tiene un carácter muy teórico y son tecnologías que, en algunos casos, no están disponibles o bien las herramientas de simulación son muy complejas y/o costosas.

P1: API multimedia de Windows: Audio.	4 h.	14.29%
P2: API multimedia de Windows: Vídeo.	4 h.	14.29%
P3: Algoritmos de compresión de audio.	4 h.	14.29%
P4: Algoritmos de compresión de vídeo.	10 h.	35.71%
P5: Transporte de audio y vídeo en una LAN	6 h.	21.43%

Tabla 1: Programa de prácticas de Redes Avanzadas.

Para la realización de las prácticas se requieren el software y hardware siguiente: El entorno de programación Borland Builder Visual C++, una tarjeta de sonido, auriculares, micrófono y cámara de vídeo. También se dispone de un servidor [1] en el que se encuentra el material de prácticas (fuentes incompletos, las soluciones en forma de ejecutable, RFCs, artículos, FAQs, fuentes de compresores, estándares, presentaciones, etc.) de esta asignatura.

Las dos primeras prácticas son introductorias. En ellas se muestra la forma de acceder al API multimedia de Windows, con objeto de capturar y reproducir tanto audio como vídeo. Esto es indispensable para poder llevar a cabo el resto de prácticas, que se basan en ésta. Las dos siguientes versan sobre las técnicas de codificación y compresión de audio y vídeo respectivamente y por último se realiza una práctica en donde se experimentan transmisiones de audio y vídeo en una LAN utilizando un sencillo protocolo.

La herramienta que se presenta aquí se utiliza en la práctica número cuatro[2], a la que se dedica más de la tercera parte de las sesiones de laboratorio y, junto con la anterior, es la que más posibilidades tiene para la experimentación, ya que las técnicas recogidas en ellas se pueden "oir" y "ver". Esto suele ser muy estimulante para los alumnos ya que su motivación se ve incrementada con el único propósito de que sus algoritmos sean capaces de realizar codificaciones y/o compresiones que ofrezcan resultados palpables, y no simples resultados numéricos.

2.- CARACTERÍSTICAS DE LA HERRAMIENTA.

Para el análisis de la codificación, compresión y transmisión de vídeo hemos desarrollado una completa plataforma, de manera que el alumno puede experimentar fácilmente sobre varios de los algoritmos de codificación y compresión de vídeo, variando los parámetros de configuración de los mismos, y observando inmediatamente los resultados obtenidos.

Esta herramienta está basada en los protocolos TCP/IP y UDP/IP, siguiendo la clásica arquitectura cliente-servidor y usando el API multimedia que proporciona Windows 9x para la captura de imágenes y procesado de ficheros de intercambio AVI. Los codificadores de vídeo empleados son una modificación de la implementación oficial del estándar MPEG-2 [2] [3], usando las recomendaciones H.261 [4] para obtener un enfoque orientado hacia la videoconferencia.

[2] Aunque dispone de módulos que permiten la transmisión de vídeo en una LAN, pudiendo utilizarse también en la última práctica de la asignatura.

A la hora de realizar la transmisión, uno de los parámetros que se puede seleccionar es la fuente de vídeo, que podrá ser tanto una vídeo cámara de sobremesa como un fichero AVI. Trabajar con una misma secuencia fuente en distintas pruebas es fundamental para que el alumno pueda obtener resultados que únicamente dependan de los algoritmos y parámetros de compresión seleccionados, y no de la secuencia que registre en ese momento la cámara. De esta forma, para asegurarse de que distintas pruebas se puedan realizar con una misma secuencia fuente usaremos vídeos almacenados en disco.

Otro de los parámetros de transmisión que se puede configurar es el tipo de protocolo que se usará para realizar la transmisión de las imágenes, eligiendo entre el protocolo UDP y el TCP. De esta manera, el alumno puede observar como, cuando se transmite un gran volumen de información, el protocolo UDP pierde más paquetes (y por tanto imágenes) que TCP, ya que éste último realizá más control sobre la transmisión, lo que también se traduce en una mayor sobrecarga computacional.

Sobre el tipo de codificación que el alumno puede elegir y sus parámetros, tenemos:

- *Codificación RGB.* La imagen se transmite sin compresión (24 bpp).
- *Codificación YUV.* Aplicamos una codificación YUV con una reducción de color 4:1:1, y con la posibilidad de transmitir sólo información de luminancia (imágenes en B/N)
- *DCT trucando altas frecuencias.* Se calcula la DCT sobre cada uno de los bloques de 8x8 (de luminancia o crominancia), pudiendo seleccionar el alumno el tamaño del bloque final. Cuanto menor es el tamaño final, más rápida se efectúa la DCT pero de peor calidad es la imagen resultante, ya que perderemos los componentes de más alta frecuencia.
- *MJPEG.* Consiste en aplicar JPEG [5] de forma sucesiva a cada cuadro de la secuencia, de manera que únicamente se consigue compresión espacial. Para este algoritmo podremos seleccionar el tipo de recorrido que se realiza para almacenar los coeficientes cuantizados (ZIG-ZAG o scan alternativo) y el propio parámetro de cuantización. El alumno observa fácilmente que con mayores cuantizaciones se alcanza un mayor nivel de compresión, a costa de una menor calidad de las imágenes.
- *H.261.* En este caso se puede seleccionar el tamaño de la ventana de búsqueda de bloques similares. Cuanto mayor es la ventana de búsqueda, mayor será el coste computacional, pero también la probabilidad de encontrar un bloque de referencia que permita minimizar el error residual aumenta. Otro parámetro que se puede seleccionar la es frecuencia con la que se codifica un cuadro I que, al ser totalmente autocontenidos, permitirán recuperarse frente a errores.

El alumno, para realizar las pruebas, deberá fijar en el servidor los parámetros deseados, de forma que éste tome las imágenes de la fuente especificada, las comprima y las transmita al cliente, que las recogerá y las descomprimirá para finalmente imprimirlas en una ventana gráfica. Después de una sesión de transmisión de vídeo, el alumno puede obtener información sobre cómo ha resultado la compresión y la transmisión, visualizando una pantalla de estadísticas similar a la que se muestra en la figura 1. Las estadísticas que se facilitan las podemos clasificar en:

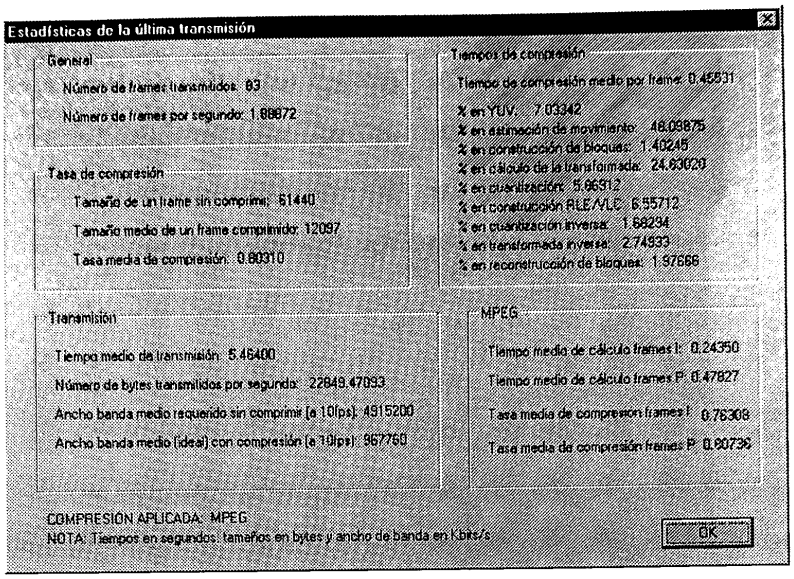


Figura 1: Estadísticas obtenidas

- Estadísticas generales. Incluye el número de cuadros transmitidos y los cuadros por segundo.

- Tasa de compresión. En este apartado se muestra cuantos bytes ocupa una imagen original, y el tamaño medio de una imagen comprimida junto con la tasa media de compresión alcanzada.

- En el apartado de transmisión se indica el tiempo medio necesario para transmitir una imagen, el número de bytes que se han transmitido por segundo y el ancho de banda que se requeriría para obtener una tasa de inyección de 10 cuadros/segundo.

- Tiempos de compresión. En esta sección se muestra el tiempo medio necesario para comprimir un cuadro y se realiza un desglose porcentual de la proporción del tiempo total que se ha empleado en cada fase del algoritmo.

- En el caso de H.261, se indica la tasa media de compresión obtenida en imágenes codificadas como I frente a las P, y el tiempo medio usado en cada caso. El alumno observará como el nivel de compresión obtenido en los cuadros I es inferior pero también el tiempo de cálculo es menor, sobre todo con grandes ventanas de búsqueda.

- Uno de los mejores métodos para determinar la calidad de las imágenes es la observación de las mismas por parte de los alumnos. Sin embargo, a veces resulta interesante tener algún valor numérico que muestre la similitud entre la imagen original frente a la comprimida, para esto se facilita el Error Cuadrático Medio (MSE, *Mean Square Error*), la Relación Señal Ruido (SNR, *Signal to Noise Ratio*) y la Relación Señal Ruido Pico (PSNR, *Peak SNR*).

3.- ALGUNAS EXPERIENCIAS PROPUESTAS A LOS ALUMNOS.

Una vez el alumno ha estudiado los principales algoritmos de compresión de imagen, y se ha familiarizado con la herramienta descrita en el punto anterior, se proponen diversas experiencias prácticas.

Uno de los parámetros que determina de forma más clara la calidad de la imagen y la tasa de compresión obtenida es el parámetro de cuantización. Esta es la primera variable que estudia el alumno, al que se le facilita una típica secuencia AVI de videoconferencia (160x120 píxels, 24 bpp, 10 cuadros/seg. y 8.4 seg.) que codifica y transmite en una red local Ethernet 10BaseT. Se pide al alumno que rellene una tabla semejante a la tabla 2, donde se observa el nivel de compresión obtenido con los distintos niveles de cuantización, el tiempo necesario para realizar la compresión y el ancho de banda requerido usando H.261. También se pide al alumno que contemple la calidad visual de las imágenes (ver figura 2) y observe su relación con el PSNR.

Parámetro de cuantización	Tasa de Compresión	Tiempo de compresión (seg)	PSNR (dB)	Ancho de banda requerido (Kbps)
1	80.209 %	0.24386	42.21	313
8	97.527 %	0.21212	35.13	54
15	98.668 %	0.20796	31.57	30
22	99.051 %	0.20658	29.66	21
30	99.238 %	0.20588	28.51	17

Tabla 2: Estadísticas según la cuantización aplicada

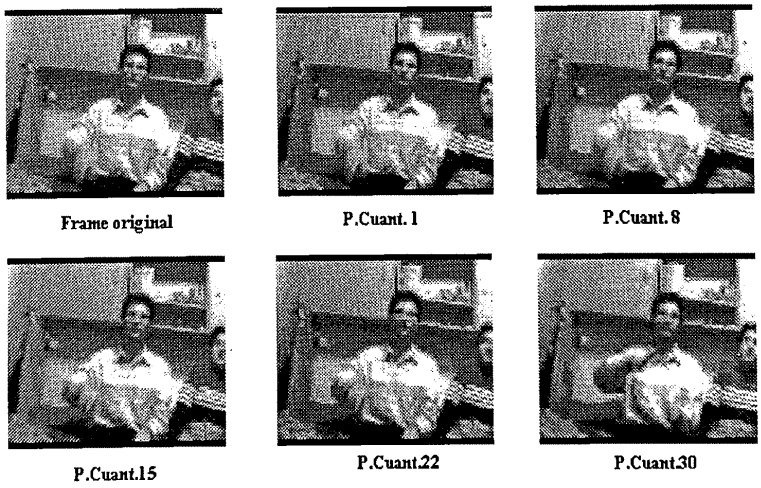


Figura 2: Cuadros obtenidos con distintos parámetros de cuantización

Una segunda experiencia práctica se realiza con el tamaño de los bloques con los que se opera al calcular la DCT, de manera que el alumno, a parte de rellenar una tabla similar a la anterior, debe observar como, con bloques más pequeños, el cálculo de la DCT pasa de ser una de las fases más costosas a tener unos costes computacionales mucho más equilibrados.

Otra experiencia propuesta es variar el tamaño de la ventana de búsqueda de H.261. En las anteriores pruebas no se realizaba una búsqueda de macrobloques en el cuadro de referencia (la ventana era de 0x0). Cuando se amplia el tamaño de la ventana de búsqueda, la fase de estimación de movimiento se convierte en una de las más costosas, lo que el alumno deberá notar en sus experiencias.

4.- ACTIVIDADES ADICIONALES.

El alumno tiene disponible un libro [6] que servirá como soporte teórico y guión práctico para la parte de la asignatura que aborda la codificación y compresión de vídeo. Una vez el alumno ha afianzado los conocimientos teóricos con las experiencias prácticas descritas en el punto anterior, deberá resolver, durante sucesivas sesiones prácticas, una serie de ejercicios propuestos en el libro, que pueden tener carácter obligatorio o voluntario.

Algunos de los ejercicios propuestos requieren la implementación de varios algoritmos de compresión, como es la programación de un codificador RGB/YUV, la implementación de diversas versiones de la DCT optimizando su coste computacional, o la construcción de un codec simplificado JPEG y H.261. Otros ejercicios deberán ser resueltos analíticamente, como es la codificación de un bloque en zig-zag usando las tablas para RLE/VLC o el cálculo del diagrama inverso de Vetterli y Ligtenberg para acelerar el procesamiento de la IDCT.

5.- CONCLUSIONES.

En esta ponencia hemos presentado una herramienta que sirve de apoyo para las prácticas de laboratorio de la asignatura Redes Avanzadas. Una de las características más sobresalientes de ésta es que ofrece a los alumnos un entorno donde se pueden experimentar de forma sencilla y cómoda las distintas técnicas de codificación y compresión de vídeo. Además es una herramienta que permite añadir nuevos algoritmos y técnicas para su evaluación y estudio, por lo que no sólo permite experimentar con los algoritmos que lleva integrados, sino que es posible diseñar y evaluar nuevos algoritmos.

REFERENCIAS.

[1] **http://www.redes.upv.es/~mperez/rav**. Servidor WWW de la asignatura Redes Avanzadas.
[2] MPEG Software Simulation Group. *MPEG-2 Codec v1.2*, 1996.
[3] Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s. ISO/IEC 11172-2: Video, 1993.
[4] *Video Codec for Audiovisual Services at px64 Kbit/s. CCITT Recommendation H.261.* International Telegraph and Telephone Consultative Committee (CCITT), 1990.
[5] Digital Compression and Coding of Continuous Tone Images (Part 1: Requirements and Guidelines). ISO/IEC 10918-1, 1992.
[6] J. Oliver, M.P. Malumbres. *Compresión de imágenes: fundamentos teóricos y aspectos prácticos.* Pendiente de publicación por la U.P.V.

UNIVERSIDAD DE ALCALÁ

SERVICIO DE PUBLICACIONES