DIRECTRICES PARA LA ACTUALIZACIÓN DE PROGRAMAS DE INFORMÁTICA EN ESPAÑA

Edmundo Tovar[i], Jesús Cardeñosa[ii]

Abstract *— Los Planes de Estudio de Ingeniería Informática se diferencian de los planes de otras disciplinas en que requieren una actualización frecuente debido al dinamismo de los conocimientos que incorpora y a distintas directrices propuestas por diversas instituciones u organismos. La efectividad de estos cambios puede controlarse a través de la valoración de indicadores de calidad de la titulación que normalmente son aplicados con fines de acreditación.*
En este artículo se analizan y organizan diferentes clases de directrices aplicables en la evolución que sufren los Planes de Estudio de titulaciones de Informática. Este problema se ilustra con un caso real de aplicación del uso de indicadores de calidad como una de los factores que deciden la necesidad de cambios en un Plan, y la aplicación de algunas de las directrices identificadas para la realización de una propuesta de actualización de un nuevo currículo de la titulación de Ingeniero en Informática de la Facultad de Informática de la Universidad Politécnica de Madrid.

Index Terms — Computer Science Education, Evaluation of Engineering Programs´ Quality, Models for Higher Education in various countries.

INTRODUCCIÓN

La Informática ha sufrido cambios drásticos durante la década pasada. Hay nuevas tecnologías que se introducen continuamente, y otras se quedan obsoletas. Los avances técnicos, como el de las aplicaciones para World Wide Web y las tecnologías de redes han incrementado la importancia de muchos temas curriculares.

Pero también ha habido cambios culturales. La informática se ha expandido enormemente gracias al desarrollo y las facilidades para el acceso a aplicaciones de software a través de Internet. Las TI tienen una creciente importancia en la economía han surgido las empresas tecnológicas, y la alta demanda de la industria para a atraer personal cualificado. Ello ha conducido a una mayor aceptación de la Informática como una disciplina académica.

[1] Edmundo Tovar, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo s/n, Boadilla del Monte 28660, Madrid, Spain, etovar@fi.upm.es
[1] Jesús Cardeñosa, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo s/n, Boadilla del Monte 28660, Madrid, Spain, carde@fi.upm.es

La entrada de la tecnología de la información en los ámbitos culturales y económicos han consolidado el reconocimiento de la disciplina.

En definitiva, la Informática es una disciplina especialmente viva. En los últimos 10 años los cambios habidos técnicos y culturales han cambiado el diseño de un

Plan de Estudios en Informática, así como la forma de enseñarlo. En la actualidad el alcance es tan amplio que es difícil establecer los límites y considerarlo como una única disciplina.

Esta es la razón por la que en la última revisión del "Computing Curricula", el CC2001 [1] se ha desarrollado en diferentes volúmenes. Sólo uno de ellos ha sido publicado, el de Computer Science, mientras que aún no lo han hecho los de Computer Engineering, Software Engineering e Information Systems. La conjunción de todas estas especialidades, según el criterio de ACM y de IEEE, se complementan perfectamente y la educación de la Informática debe cubrir por lo tanto, el nuevo alcance de la Disciplina.

En este artículo se tratan dos de los aspectos fundamentales que pueden ayudar en la elaboración de nuevos planes de estudio en informática: los indicadores de control referidos a planes de estudio, y las directrices que pueden ser tenidas en cuenta. Finalmente se analizará el uso de esta información en la propuesta de un nuevo plan de estudios en informática.

LOS INDICADORES DE CONTROL

Las Universidades españolas usan un sistema de indicadores que hace posible que éstas cuenten con una información de calidad y con potencial estratégico en la que se fundamentan sus decisiones y con la que los implicados pueden tener un certero conocimiento y un diagnóstico preciso de su realidad.

Cuando una universidad quiere informar a sus miembros, a los responsables políticos o a la sociedad en general sobre su calidad se pueden utilizar diversos tipos de indicadores, con significado diferente desde distintas perspectivas, pues la calidad es un concepto multidimensional. Se puede informar de la calidad a través de distintos tipos de indicadores [2]:

- **Indicadores directos**: son los resultados que obtiene la institución cumpliendo sus funciones.
- **Indicadores de impacto**: medidas de valoración o satisfacción de la institución por

© 2003 ICECE

parte de las distintas audiencias con los colectivos implicados.

- **Indicadores indirectos**: medidas con información de sus entradas, de sus características organizativas, de sus recursos y de sus procesos de funcionamiento.

Bajo esta clasificación los indicadores que aquí nos interesan son un conjunto de indicadores indirectos que afectan al proceso de enseñanza aprendizaje y que viene representado por el Plan de Estudios. Así, los indicadores utilizados por las universidades españolas que guardan una relación directa con el Plan de Estudios están recogidos en la tabla I, todos ellos indicadores indirectos:

TABLA I
INDICADORES REFERIDOS AL PLAN DE ESTUDIOS DEL SISTEMA UNIVERSITARIO PÚBLICO ESPAÑOL

Denominación	Explicación
Dedicación lectiva del alumnado	Media de créditos en los que los alumnos de una titulación se han matriculado. Este indicador se puede usar para analizar si la carga docente está por encima o por debajo de otros posibles valores de referencia.
Prácticas requeridas	Es la relación entre el número total de créditos prácticos requeridos y el total de créditos aprobados en el plan de estudios, dando así la importancia que tiene la oferta de docencia práctica.
Optatividad requerida de la titulación	Es la relación entre el número total de créditos optativos y de libre configuración que ha de cursar el alumno a lo largo de la carrera y el total de créditos a cursar para obtener el título. Es una manera de indicar el peso específico de la optatividad.
Oferta de optatividad	Relación entre el número de créditos optativos diferentes ofertados y el número total de créditos que ha de cursar el alumnado. Valores altos reflejan una gran oferta de optatividad.
Prácticas en empresa	Relación de créditos que el alumno debe cursar en empresas u organismos públicos, y el número de créditos prácticos que debe cursar.

Estos indicadores forman parte de un catálogo de indicadores que son aplicables al conjunto de las universidades españolas, independientemente de la naturaleza de sus disciplinas, humanas sociales o técnicas. Pero no basta con ofrecer información, sino que hay que contar con referentes para su contraste que permita decidir si el Pan de Estudios es mejor o peor. Por eso aquí se comenta también un sistema de indicadores específicos para titulaciones técnicas que forman parte del método de evaluación SECAI [3].

Los indicadores correspondientes al Plan de Estudios, según este método, se encuentran organizados en tres apartados:

- aquellos referidos al **proceso de elaboración**: indican en qué medida los elementos representados por el indicador, aparentemente, a elaborar un plan de estudios de calidad.
- Respecto del **contenido** del Plan de Estudios: señalan en qué medida los elementos representados por el indicador se manifiestan de la manera esperada en un plan de estudios de calidad.
- Sobre la **puesta en marcha**: representan en qué medida los elementos representados por el indicador se manifiestan de la esperada si el proceso y seguimiento del Plan de estudios es o fue de calidad.

TABLA II
INDICADORES REFERIDOS AL PLAN DE ESTUDIOS DEL SISTEMA DE EVALUACIÓN SECAI

Categoría	Indicadores
Proceso de elaboración	- Participación de agentes externos en la carrera - Participación de los miembros de la comunidad académica - Definición de ámbitos y perfiles profesionales de los titulados en función de la demanda social. - Análisis y valoración de los conocimientos, capacidad e intereses de los estudiantes que acceden a la carrera. - Análisis de la congruencia entre contenidos, métodos y recursos en función de los objetivos. - Posibilidades de logro en función del tiempo disponible para cursar el Plan.
Contenido	- Definición expresa del modelo de formación y de los objetivos educativos de la carrera - Estructura del Plan de estudios - Orientaciones metodológicas y su adecuación a los objetivos - Orientaciones relativas a la evaluación de conocimientos y su adecuación a los objetivos - Plan académico de cada asignatura - Contenidos de las enseñanzas. Actualización y adecuación a los objetivos - Previsión de actividades de nivelación y refuerzo
Puesta en marcha	- Previsión de las necesidades de acuerdo a los contenidos del plan - Nivel de conocimientos y aceptación del Plan por parte de los miembros de la comunidad educativa - Programa de seguimiento, evaluación y actualización del Plan

SECAI, por tanto, es más exhaustivo en la relación de indicadores que utiliza con respecto del Plan de Estudios, pues no sólo trata aspectos relacionados con sus contenidos sino con el proceso de elaboración, y su seguimiento. Su orientación a la evaluación de estudios técnicos se manifiesta al dar especial relevancia a la participación de agentes sociales en su elaboración y en la definición de perfiles profesionales, así como en el seguimiento en la actualización de contenidos. Además la descripción de cada indicador es más formal respondiendo mejor a un espíritu ingenieril, incluyendo información como criterios orientativos que deben utilizarse para evaluar el indicador, o la lista de informaciones (institucionales o resultados de encuestas) que son necesarias y están disponibles. Esto hace posible prácticamente la aplicación y evaluación sistemática de los indicadores.

Con el apoyo de la Comisión Europea, el consorcio Career Space [4], que está constituido por once importantes empresas de Tecnologías de Información y Comunicación (TIC) y por la Asociación Tecnológica Europea de Industrias de la Electrónica, la Información y las Comunicaciones, ha estado abordando la escasez de profesionales en este sector. Aparte de las iniciativas que tomaron, y que serán referidas más adelante, recomiendan que las universidades deben introducir un proceso de control

de la calidad con resultados documentados. Y que la información obtenida debe aplicarse para seguir perfeccionando el programa. Según este consorcio, el proceso de control de calidad debe obtener retroalimentación del sector empresarial respecto a la evaluación de competencias de los antiguos estudiantes en áreas técnicas y de comportamiento después de su contratación. También requiere, partir de los comentarios de los estudiantes, conocer en qué medida el curso consigue los resultados esperados, y si el estudiante piensa que durante el curso ha adquirido los conocimientos y capacidades adecuados de cara al mercado laboral.

ORGANIZACIÓN DE DIRECTRICES

Hay distintas formas de clasificar y organizar las directrices que pueden considerarse en la renovación de planes de estudio de titulaciones de Informática. Dependiendo de quienes son las entidades que las elaboran pueden distinguirse directrices propias de las Asociaciones profesionales de reconocido prestigio, instituciones gubernamentales que tienen a su cargo la política educativa de un país o de una región que intenta armonizar las correspondientes a distintos países, las promovidas por las propias universidades, o las propuestas por el tejido empresarial a través de estudios de demanda de empleo.

En adelante se ilustrarán estas diferentes directrices con ejemplos en el contexto europeo y español, a los que pertenece el centro universitario que se tomará como caso real de actualización de plan de estudio.

Directrices de Asociaciones profesionales

Los estudios universitarios de la titulación de Ingeniería Informática en España se han basado tradicionalmente en los curriculums elaborados conjuntamente por los esfuerzos conjuntos de las asociaciones IEEE y ACM. En 1998 estas organizaciones formaron un grupo de trabajo para diseñar el Comuting Curricula 2001 [1], con el fin de llevar a cabo una importante revisión y mejora del anterior, fechado en 1991 [5]. Los principales cambios residieron en considerar los cambios tecnológicos, y cambios culturales en los últimos diez años. Por otra parte, el término computación se ha ampliado en estos últimos años hasta tal punto que es difícil definirlo como una única disciplina. Por ello decidieron desglosar el currículo en cuatro bloques: Computer Science, Computer Engineering, Software Engineering e Information Systems.

Directrices Gubernamentales

La universidad española inició un camino hacia un marco de autonomía e independencia con la Ley de Reforma Universitaria, en 1983. Los aspectos más novedosos fueron el régimen estatutario de las universidades, la organización departamental y la reforma universitaria mediante la ordenación académica de las enseñanzas. Al Consejo de Universidades, creado en 1985 con tal fin, le fue atribuido la competencia de proponer al Gobierno los títulos con carácter oficial y validez en todo el territorio nacional, como las directrices generales comunes y de homologación.

En 1987 se publican las Directrices Generales comunes de todos los planes de estudio de ámbito nacional y de carácter oficial [6], que establece que éstos deben ser elaborados por las universidades y posteriormente homologados por el Consejo de Universidades. Este marco legislativo presenta como aspectos fundamentales:

- **Flexibilización**: se distinguen tres bloques de materias. Troncales (contenidos mínimos comunes que deben incluir los planes conducentes al mismo título), Obligatorios (representan el carácter distintivo de la universidad), Optativas (representan una orientación elegida por el alumno para su formación), Libre configuración (con contenidos formativos que el alumno desea adquirir no directamente relacionadas con el título al que opta).
- **Estructura cíclica**: el primer ciclo comprende enseñanzas básicas y de formación general, y el segundo ciclo se dedica a la profundización y especialización en las enseñanzas. Los estudios de primer ciclo tienen duración de dos o tres años, y el segundo ciclo de dos años. La carga lectiva oscila entre 60 y 90 créditos por año académico, teniendo un crédito un equivalente de 10 horas de enseñanza.

Las directrices propias de cada titulación establecieron posteriormente el marco general que hizo compatible un mínimo de homogeneidad entre las distintas titulaciones oficiales, pues incluían la estructura y duración de las correspondientes enseñanzas, los objetivos formativos y el perfil profesional, indicando su número de créditos, su adscripción a áreas de conocimiento y una breve descripción de sus contenidos temáticos. De licenciatura y diplomatura en Informática se pasa a titulaciones de Ingeniero en Informática [7]. Según estas directrices, la carga lectiva no podía ser inferior a 300 créditos ni superior a 450. La carga lectiva semanal debía oscilar entre 20 y 30 horas, incluidas las prácticas (la carga teórica lectiva no podía superar las 15 horas semanales). Las enseñanzas prácticas consumen entre el 40 y el 50% del total de los créditos. Las materias troncales debían consumir un mínimo de 69 créditos y las de libre configuración el 10% del total.

Por otro lado, en loso últimos años se está creando el espacio europeo de la enseñanza superior, nacida en la Declaración de Sorbonne el 25 de Mayo de 1998 [8], donde se enfatiza el papel central que tienen las universidades para el desarrollo de las dimensiones culturales europeas. Posteriormente en la Declaración de Bolonia [9] los ministros europeos a cargo de la educación superior de 32 países firmaron una declaración conjunta en donde aceptan el desafío de construir el área europea de educación superior, frente a la tradicional independencia y autonomía de las universidades. Los objetivos propuestos se reagrupan en torno al pilar esencial de la valoración de la calidad para

dotar al espacio europeo de confianza, pertenencia, movilidad y compatibilidad.

Directrices de Universidades

Los estudios universitarios de la titulación de Ingeniería Informática en España están organizados de distinta múltiples maneras, con universidades que sólo tienen un título o con otras que tienen una oferta en Informática con titulaciones de primer y segundo ciclo. En este último caso, incluso se dan casos en que el primer ciclo de la carrera superior es realizado por la Ingeniería técnica, y en otros o bien coexisten en el mismo o distintos centros de la titulación técnica y superior con variantes que van desde un determinado cupo de alumnos para acceder al segundo ciclo hasta planes de estudio que no facilitan el acceso de los ingenieros al segundo ciclo. Además, ha aparecido una enseñanza superior no universitaria tanto pública como privada que obliga a reconsiderara los estudios universitarios de informática. Ante esta heterogeneidad comienzan a existir unas directrices comunes, aunque no de obligado cumplimiento, que son discutidas y acordadas a través de la "Conferencia de Decanos y Directores de Informática" (CODDI), Conferencia que reúne a los responsables de la titulación de todas las universidades públicas y privadas españolas. En la actualidad el tema que ha centrado los debates se ha referido a determinar unas directrices comunes que permitan la adaptación de los estudios de las Ingenierías en Informática a la Declaración de Bolonia.

Los objetivos propuestos por la CODDI [10] han partido de la base en que la evolución de los planes de estudio basados en Bolonia debe cambiar la estructura emanada de la Ley de Reforma Universitaria en algunos aspectos importantes. En cualquier caso, se deben considerar tres elementos importantes:

- La informática forma una unidad que no permite diferenciar desde el principio de los estudios de informática de primer y segundo ciclo.
- Hay que incorporar otros conocimientos no estrictamente técnicos, como capacidades de negocio, sociales e individuales.
- Se deben mantener de la LRU dos conceptos básicos: las materias troncales y el catálogo de titulaciones.

Una vez constatados los inconvenientes de la especialización en el primer ciclo y la generalización en el segundo, para implantar las recomendaciones de Bolonia la opinión de la CODDI es que hay que adoptar una estructura en forma de un único primer ciclo, de 4 años, con una fuerte troncalidad que permitiera el acceso un segundo ciclo cuyo énfasis sería la especialización y la profesionalización, con muy poca o ninguna troncalidad, o bien el acceso al doctorado.

Las directrices proporcionadas se refieren a:

- Tamaño del Plan , utilizando como unidad de medida el Crédito europeo (ECTS). En concreto, entre 224 y 260 ECTS, siendo un crédito ETCS equivalente a 25 horas de trabajo para los estudiantes, incluyendo asistencia a clase, desarrollo de actividades y prácticas, estudio personal, asistencia a exámenes. Si se tiene en cuenta que un curso académico tiene una duración de 40 semanas, se puede aconsejar que la carga por semana para el alumno esté entre 35 y 40 horas.
- Distribución de materias de corte genérico, de arquitectura y tecnología de computadoras, de software y de proyectos para el primer ciclo.
- Para el segundo ciclo se dan sugerencias sobre materias de Arquitectura y Diseño Software, Desarrollo de Aplicaciones Software, Especialista de Sistemas de Información, Diseño de Sistemas Multimedia y Interacción Persona-Máquina, Telemática, Integración y Prueba o de Implementación y Prueba, Soporte Técnico, Diseño de Productos Digitales, Gestión de la Tecnología y de Proyectos, Informática Industrial y Sistemas Empotrados.

Directrices de empresas

En España la profesión informática no está bien definida, pues se convive con una gran variedad de títulos, tanto públicos como privados, a los que hay que añadir los diplomas públicos de enseñanza no reglada. Ante este problema de diferenciación profesional esta titulación está en permanente competencia con las Ingenierías de Telecomunicaciones, la Ingeniería Industrial y las licenciaturas en Físicas y Matemáticas, así como con titulaciones revestidas generalmente como "certificaciones" procedentes del sector empresarial, como Microsoft, Sun, Oracle o Cisco.

Frente a este problema de diferenciación el consorcio Career Space [4] cree que no existe una forma única de diseñar el currículo ideal en esta materia. Primero ha definido una serie de perfiles de capacidades genéricas básicas que se ofrecen como punto de referencia para las universidades. Pero aunque cada universidad tiene que encontrar por su cuenta la mejor solución, este consorcio ha elaborado un conjunto de directrices útiles cuya aplicación puede ayudar a las universidades a encontrar su propio camino hacia el éxito. Después de hacer un análisis de las tareas específicas de un trabajo en particular se llegó a la conclusión de que aunque las demandas de las empresas pueden ser distintas según la tarea que tienen que realizar, la estructura básica de los conocimientos necesarios es la misma. Esta es la razón por la que se pueden dar directrices comunes. Éstas fueron:

- Las calificaciones técnicas necesarias tienen un amplio espectro de conocimientos en matemáticas, ciencia y tecnología. Esa base es un requisito importante para que los graduados se puedan comunicar con colegas de otras áreas

por medio de un lenguaje técnico común. Esa base se puede distribuir en un 30% de base científica, para comprender los métodos científicos para el análisis y el diseño, otro 30% de base tecnológica, que proporciona una visión general de las distintas tecnologías disponibles, y un 25% de conocimiento básico de sus campos de aplicación, de sus aplicaciones particulares según las demandas del lugar de trabajo para el perfil de un puesto de trabajo particular.

- El profesional de las TIC requiere la aplicación y el desarrollo continuos de las capacidades personales y empresariales por medio de proyectos en equipo, negociaciones, o presentaciones. Por eso se debe prestar una atención especial a la integración de la enseñanza de estas capacidades personales y empresariales esenciales en áreas temáticas más técnicas, que deben ocupar al menos el 15% del currículo.
- No basta con aprender cuestiones técnicas, sino que además hay que utilizarlos en situaciones reales, y conocer los problemas relacionados con los derechos de propiedad intelectual y el secreto comercial. Para conocer estas cuestiones, el consorcio recomienda realizar prácticas empresariales durante un período mínimo de tres meses. De esta manera se ayudará al alumno a elegir el tipo de trabajo que le gustaría encontrar después de graduarse.
- Para adquirir las capacidades profesionales es importante dedicar al menos tres meses a un trabajo en proyecto, aunque sea difícil evaluar el trabajo de un alumno individual.

El informe PACET [11] (Propuesta de acciones para la formación de profesionales de electrónica, informática y telecomunicación), promovido por ANIEL (Asociación Nacional de Industrial electrónicas y de telecomunicaciones), y el Colegio Oficial de Ingenieros de Telecomunicación, ofrece un análisis sobre el déficit de profesionales TIC en España, pero adaptando aquellos perfiles propuestos por Career Space a las demandas profesionales, identificando un total de 20 perfiles. Entre las recomendaciones recogidas en este informe destacan las siguientes:

- Recoger información sobre los perfiles profesionales, las habilidades requeridas, su oferta y la evolución en el mercado.
- Participación en los diseños curriculares de las empresas y las administraciones.
- Dar más peso a la formación de las tareas de gestión.

Este informe no ha recibido una aprobación unánime entre los colectivos implicados. Las principales críticas han sido realizadas por los Colegios de Ingenieros en Informática existentes en España. No se han tratado las acciones formativas que se precisan para el reciclaje profesional, ni se ha contemplado la formación precisa de perfiles de máxima responsabilidad de las áreas de Dirección de las TIC. Además se llega a recomendar un perfil de ingeniero superior para perfiles de programado de aplicaciones lo que demostraría un desconocimiento de los autores del informe de las titulaciones universitarias de Ingeniería en Informática y Telecomunicación.

En cualquier caso parece que existe un desfase entre la definición de perfiles obtenidos con respecto de las competencias y habilidades definidos en los planes de estudio de informática, por lo que la aplicación de estas últimas recomendaciones suponen asumir un riesgo.

UN CASO REAL: LA FACULTAD DE INFORMÁTICA DE LA UPM

En la Facultad de Informática de la Universidad Politécnica de Madrid, se llevó a cabo por primera vez un proceso de evaluación de la calidad en el año 2000.

Una de las tareas previas consistió en definir con precisión los indicadores a evaluar. Se tomaron como referencia los siguientes métodos, además de la Guía de Evaluación del Consejo de Universidades: el método SECAI (Sistema de Evaluación de la Calidad de Ingenierías), y el modelo europeo para la Gestión de la Calidad Total (EFQM). Se incorporaron nuevos indicadores que no fueron contemplados por la Guía del Consejo de Universidades. Como resultado se seleccionaron y describieron 66 indicadores relacionados con el área de enseñanza, 37 del área de Investigación y 34 correspondientes al área de tercer ciclo.

En la tabla III se resumen los resultados obtenidos referidos al Plan de Estudios:

TABLA III
RESUMEN DE CARENCIAS DETECTADAS EN EL PROCESO DE EVALUACIÓN REFERIDOS AL PLAN DE ESTUDIOS EN LA FI DE LA UPM

Categorías de indicadores	Puntos débiles
Estructura del Plan de Estudios	- Falta de definición de perfiles en el Plan de Estudios - No existe documentación registrada en la elaboración del Plan de Estudios - No hay especializaciones - Desequilibrio teoría-práctica en algunos casos. La carga real práctica es excesiva
Programas de Asignaturas	- Falta de datos para poder valorar la adecuación y coherencia de los programas de asignaturas - Falta de coordinación en la planificación de las prácticas
Organización de la Enseñanza	- Escasa participación del profesorado en la dirección de TFC - Tamaño excesivo en número de alumnos de los grupos

En paralelo, pero una vez elaborado el Informe final de evaluación de la titulación [12], un grupo de trabajo en el centro recibió el encargo de realizar una propuesta de un nuevo Plan de Estudios para que fuera debatido por los

colectivos de la FI [13]. En concreto se tomaron en cuenta las siguientes directrices:

- Planes de Estudio en informática de otras universidades españolas (por ser de nuestro entorno más cercano), europeas (por la necesidad de converger con sus enseñanzas de referencia), y universidades de EEUU (por ser un país de referencia en temas informáticos). La información ha sido de utilidad para confirmar el rango de posibilidades y alternativas en la enseñanza de la informática.
- Computing currículum de ACM y de IEEE de 2001. En la actualidad es el único marco genérico internacional existente sobre curricula. Pero se ha encontrado con dificultades en aplicarlo pues la unidad de crédito no es equivalente, y el alumnado entre en la universidad con diferentes conocimientos.
- Boletín Oficial del Estado 20-11-1990, donde se establece la troncalidad mínima. Es de obligado cumplimiento para su homologación.
- Encuesta a profesionales informáticos, para conocer más de cerca la demanda profesional en nuestro entorno más cercano.
- Encuesta sobre los coordinadores de asignaturas con respecto de los contenidos de asignaturas, para asegurar el escalonamiento de las asignaturas relacionadas con el plan actual.

CONCLUSIONES

La actualización de Planes de Estudio en Informática es una tarea habitual para las universidades dada la actualización continua que requieren los contenidos técnicos que hay que transmitir, como la indefinición, quizás debida a su juventud, de la profesión informática. En este artículo se han recogido, descrito y organizado las directrices propias para este fin aplicables en el contexto español y europeo.

En la experiencia de una propuesta de un nuevo Plan de Estudios que se ha seguido se han utilizado la mayor parte de las directrices mencionadas. Algunas de ellas porque forman parte del marco jurídico español. Otras porque son directrices de reconocido prestigio. Pero las ha habido que no han sido utilizadas porque los proponentes han considerado que esas recomendaciones sólo cubren parte del problema o porque han aparecido tan recientemente que no ha sido posible aplicarlas. Este es el caso del efecto de la Declaración de Bolonia, y de los esfuerzos de armonización conjunta entre todos los directores de titulaciones en Informática. En cualquier caso es necesario tener en cuenta los nuevos perfiles profesionales, y tener en cuenta las habilidades básicas que requieren los nuevos profesionales en el sector.

Es decir, afortunadamente cada vez hay más directrices adecuadas para la elaboración de un plan de estudios en informática, aunque no todas las necesarias. Pero simultáneamente hay un esfuerzo cada vez mayor en la definición precisa de indicadores que controlen la calidad de los planes implantados. El conocimiento de estos indicadores ayudan a su vez como guías para recopilar las directrices necesarias.

REFERENCES

[1] The Joint Task Force on Computing Curricula, ACM CS-IEEE, "Computing Curricula 2001", Final Report, December 2001.

[2] Consejo de Coordinación Universitaria, Borrador del Catálogo de Indicadores del Sistema Universitario español, Ministerio de Educación, http://www.mec.es, 2001.

[3] Instituto de Ciencias de la Educación, UPM, "Sistema de Evaluación de la Calidad de las Enseñanzas de Ingeniería SECAI. Documentos de Aplicación", Programa Columbus, julio de 1999.

[4] Career Space, "Directrices para el desarrollo curricular", http://www.career-space.com, Oficina de Publicaciones oficiales de las Comunidades Europeas, 2001.

[5] The Joint Task Force on Computing Curricula, ACM CS-IEEE, "Computing Curricula 1991", Report nº 201910, 1991.

[6] Real Decreto 1497/1987, Ministerio de Educación y Ciencia, 1987.

[7] Real Decreto 1459-60-61/1990 del Consejo de Universidades, Ministerio de Educación y Ciencia, 1990.

[8] Joint declaration of the european ministres of Education, "Sorbonne Declaration", Sorbonne, 25 mayo de 1998

[9] Joint declaration of the european ministers of Education, "The European Higher Education Area", convened, Bologna, 19 de junio de 1999.

[10] Ramón Puigjaner, Universitat de les Illes Balears, Conferencia impartida en las jenui, *Jornadas de Enseñanza Universitaria de Informática*, Cáceres julio de 2002.

[11] Proyecto PAFET, Propuesta de acciones para la formación de profesionales en empresas de electrónica, informática y telecomunicación, http://www.getec.etsit.upm.es/investigacion/pafet/pafet.htm, 2001.

[12] Comité de Evaluación Interna, "Informe final de Evaluación de la titulación de Ingeniero en Informática", Informe Técnico FIM/114.2/Decanato/2002, ed. E. Tovar, 2002.

[13] Equipo de trabajo de Plan Estudios, "Propuesta de Plan de Estudios", v5.0, Faultad de Informática, UPM, Septiembre de 2002.

USING PAIR PROGRAMMING TECHNIQUES IN CLASSROOM ENVIRONMENT

Carlos Alberto Ynoguti[1], Afonso Celso Soares [2]

Abstract *— For several years, software development companies have been using pair programming with great success. Pair programming is a technique that consists of two programmers working together on the same computer to develop a computer program: one of them, the driver, operates the computer, and the other, the observer, examines the work of the driver, looking for errors and thinking of possible alternatives that can improve the solution. This procedure is closely related to Collaborative Learning philosophy, extensively used in elementary and high school teaching, with very good results. Unfortunately, its use in undergraduate courses, especially in the engineering area, is almost absent. Therefore, we decided to adopt pair programming in a first semester computer-programming course, both in theoretical classes and in laboratory practical classes. This paper reports the results of a one-semester course using pair programming.*

Index Terms — pair programming, collaborative learning, computer science learning, extreme programming.

INTRODUCTION

Traditionally, students find introductory computer science courses very frustrating (in our institution, about 30% of them fail the subjects at each semester).

With pair programming learning, two students work simultaneously to solve a task (an algorithm or a computer program). In this technique, one of the students is the "driver" and has control on the pencil/mouse/keyboard and writes the algorithm or the program. The other, called the "observer", continuously and actively examines the work of the driver, watching for defects, thinking of alternatives, looking up resources, and considering strategic implications of the work at hand. Examples of things noted by the observer are erroneous syntax, misspelling, and smaller logic mistakes, among others.

The student pairs apply a positive form of "pair-pressure" on each other, which has proven beneficial to the quality of their work products. At the end of the semester, the students were given a questionnaire about the pair-programming experience and most of them reported good impressions about this technique.

Also, this technique has proven to be beneficial for the teachers too. Some minor questions are answered inside the pairs. The number of exercises to correct is divided by a factor of two, enabling the teacher to give more exercises, and consequently, making a better evaluation of what issue is or is not being absorbed by the students. One important thing to note is that the number of cheating cases is reduced because collaboration is legitimized.

Cognitive theory can help explain why pair programming might be more effective than solo programming. In 1991 Nick Flor, a master's student of Cognitive Science at U.C. San Diego, reported on distributed cognition in a collaborative programming pair he studied. Flor recorded via video and audiotape the exchanges of two experienced programmers working together on a software maintenance task. In [3], he correlated specific verbal and non-verbal behaviors of the two programmers to known distributed cognition theories. One of these theories is "Searching Through Larger Spaces of Alternatives":

"A system with multiple actors possesses greater potential for the generation of more diverse plans for at least three reasons: (1) the actors bring different prior experiences to the task; (2) they may have different access to task relevant information; (3) they stand in different relationships to the problem by virtue of their functional roles. . . An important consequence of the attempt to share goals and plans is that when they are in conflict, the programmers must overtly negotiate a shared course of action. In doing so, they explore a larger number of alternatives than a single programmer alone might do. This reduces the chances of selecting a bad plan."

In this article, an experience involving pair programming learning technique in a first semester undergraduate computation course is presented. Advantages and disadvantages of this method are also presented and discussed.

Pair programming learning strategy is based on collaborative learning theory, which has been widely researched and advocated throughout the professional literature, mainly at the primary and secondary levels. For higher level courses, it's been adopted recently in some institutions with good results. This theory is further discussed in the next section.

[1] Carlos Alberto Ynoguti, INATEL – National Institute of Telecommunications, Av. João de Camargo, 510, 37540-000, Santa Rita do Sapucaí, MG, Brazil, ynoguti@inatel.br

[2] Afonso Celso Soares, INATEL – National Institute of Telecommunications, Av. João de Camargo, 510, 37540-000, Santa Rita do Sapucaí, MG, Brazil, acsoares@inatel.br .

© 2003 ICECE

COLLABORATIVE LEARNING

The term "collaborative learning" refers to an instruction method in which students at various performance levels work together in small groups toward a common goal. The students are responsible for one another's learning as well as for their own. Thus, the success of one student helps other students to be successful [5].

The collaborative learning points out the active participation and interaction, either between the students and the teacher or among the students.

Basic elements of collaborative learning

The basic elements of the collaborative learning method can be summarized into the following items[6]:

1. Group interdependency: the students, as a group, have a common goal and should work as an efficient team to reach it. The students are responsible for their own apprenticeship. This procedure helps in the apprenticeship of every member of the group.
2. Interaction: one of the goals of the collaborative learning is to develop the student's competence in working in groups. Each member of the group should accomplish his part of the task and spare some time to share his knowledge with his partner(s) and, on the other hand, receive the contributions of his partner(s).
3. Diverging thoughts: there shouldn't be a member that claims himself as the leader or the smart guy, but instead, a conscience that both of the members of the group can explain their own points of view, competence and perspectives. The activities should be created in order to demand collaboration instead of competition (complex tasks that require creativity and has several possible solutions).

Vygotsky's socio-cultural theory

The socio-cultural theory by Vigotsky about the learning process emphasizes that the human intelligence comes from our society and culture, and happens at first time because of the interaction with the social environment.

Another aspect of Vygotsky's theory is the idea that the potential for cognitive development is limited to a determined zone that he called "proximal development zone" (PDZ). He defines this concept as "the distance between the real development level, determined by the independent problem resolution, and the potential development level, determined by the problem resolution under a supervisor advising or in collaboration with more capable partners" [1].

It's important to consider that the PDZ varies with the culture, the society and the experience of each individual.

For a PDZ to be created, there should exist a joint activity that enables the interaction between teacher and students. The group work allows the confront and integration of different points of view, making the learning process richer and more interesting.

Of course, people learn by themselves naturally provided that there exist adequate and minimally stimulant contexts. However, if a teacher helps a student to analyze and reflect about his/her actions, the learning process is accelerated.

EXPERIMENTS

Class characterization

Before describing the pair programming tests results, it's instructive to characterize the classes they were performed. The studies were carried out in an undergraduate, first semester computing class. In our institution, classes have typically 70-100 students. The course is divided into two parts: a theoretical part (60 hours) and a practical one (30 hours).

In the theoretical part, the students learn how to construct algorithms to solve simple problems, and use a pseudo-code language to construct these algorithms. All the algorithms are written with pencil and paper and the students are continuously invited to debug their algorithms simulating the behavior of the compilers.

Now, in the practical part, the students are divided into smaller groups (20 to 25 students) and use the Delphi® compiler to create their programs.

Grades are distributed in the following manner: 2 theoretical tests and 3 practical tests, taken individually. Also, during the semester, 10 to 15 exercises are assigned and corrected, and add a bonus up to 10 points in the final grade (grades range from 0 to 100 points). Part of these exercises was done individually, and part with a partner, using the pair-programming technique.

In other words, the pair-programming technique was used as a learning method, not as an evaluating one, although part of the grading had been obtained using this technique.

Solo and pair programming tests

As reported earlier in this article, the pair programming technique was used in the classroom and laboratory exercises. In the first part of the semester, the students developed their activities working alone. Because of the class size (74 students) and the class dynamics (teacher was always solving students' doubts during the exercises), cheating was difficult to control, and it was common to see 4, 5 or even 6 exercises with identical (and wrong) solutions. It was clear from these results that only a few students did the exercises. The others just cheated. It was a frustrating result, because it's clear from these facts that students were more interested in the bonus points than in learning how to program.

To try to change this behavior, the students were told that if the teacher found more than two exercises with identical solutions, the grade would be divided by the

number of identical exercises. After this, the number of cheating cases dropped, and so did the grades.

Of course, as a result of this scenario, the students' grades in the first tests were not good.

In the second part of the semester, they were given an article about pair-programming [2] to read and had a brief explanation about the new working method. After this, they were invited to test on the new working methodology. None of them was obligated to work in pairs, but the teacher encouraged them to try the method before deciding how they would like to work. These activities were done both in classroom, with algorithm design problems, and in the laboratory environment, with code design problems.

Pairs were formed in a free way. No constraints were made in this sense and, in general, the students chose their friends or the students that was seated nearby, and later, it was noticed that this procedure was an error, as shown in the next section.

At the end of the semester, the students were invited to answer some questions concerning the pair programming experience. The questions were extracted from a work by Williams & Kessler [2] with some minor modifications. The answers the students gave will be commented later in this article.

Results

Cheating cases dropped dramatically, and the number of different solutions raised enormously. Also, the solutions varied from very sophisticated ones to very complicated and inefficient ones, but the great majority of the designs met the specifications of the problems, a result quite different from the first part of the semester.

The classes became very noisy, but this was because the students were really discussing solutions and alternatives to solve the problems and actively participating in the class. Also, when students asked the teacher to clear doubts, they came with more elaborated questions, not trivial ones, so the easy answering doubts were cleared independently by the students.

Specifically in the algorithms made in class, syntax errors are very common because of the lack of a compiler that reports them to the programmer. With pair programming this kind of error dropped dramatically.

Now in the laboratory environment, it was noted that the students waste less time doing other activities (such as talking, surfing on the internet, etc.) because of the partner's pressure. Also, they learn not only the theoretical aspects of programming, but also get some tips from their classmates: hot keys, typing tricks, help usage and other things were learned just by observing the partner working.

A final feature of this method is that, as students work in pairs the task of correcting the exercises is approximately divided by a factor of two, and then it can be possible to give more exercises during the semester and keep a closer look on how the students are assimilating each part of the subject and reinforce the weak areas.

Questionnaire analysis

The questions taken from [2], and the most common answers, were:

1) *It has been said among teachers, "You do not know it unless you can teach it." Do you find any value to yourself in explaining your work to your partner?*

Many students reported that when explaining some subject to his/her partner, they had to elaborate it in a more detailed fashion, so they learned a little bit more and noticed several aspects of the subject during this process.

2) *Do you feel you have learned anything just by reading your partner's code?*

The great majority of the students answered "no" to this question. They reported that they learned almost nothing by observing their partners working.

3) *What was the biggest hurdle you have had to overcome as a collaborative programmer?*

Accepting another strategy, different from what they had traced in the principle, was the most common problem reported.

4) *What kinds of things does the non-driver do as he/she observes?*

Some syntax errors, erroneous indentation, and minor logic mistakes were the most frequently answers found.

5) *What do you think is the biggest advantage of collaborative programming?*

Most of the students reported that the big advantage of pair-programming is that they could perform better algorithms, with fewer errors and in less time.

One of the students reported an interesting fact: during one development section, he didn't know how to solve the first part of the problem, and his partner helped him. In the second part, the opposite happened, he found the answer that his partner couldn't. So, working separately, both of them would fail the exercise, but working together, they could accomplish their goal.

6) *What do you think is the biggest problem with collaborative programming?*

The main problem the students found in pair-programming practice was when their partners didn't accept different ideas or suggestions. Some of the students also reported that their partner simply did nothing, not cooperating for the solution of the problem.

Drawbacks

As any other learning methodology, pair-programming has its drawbacks too. Analyzing the questionnaire answers and based on our own observations, we can list the following drawbacks in using pair-programming methodology in a classroom environment:

Some students really dislike working in pairs, and prefer to work alone. Some of them not even tried to work in pairs, preferring to work alone, even scoring only poor grades.

Pair choosing is another aspect that must be addressed carefully. They must be formed with one student that has a higher knowledge/skill level than the other, so he/she can help his/her partner. Pairs with two low knowledge/skill level students didn't worked also, because none of them could help each other. A student with very low knowledge/skill level together with a very high knowledge/skill level is another kind of pair that doesn't work, because the "expert" student quickly becomes bored with his partner and resolves the problem alone, without explaining the solution to his/her partner.

Another thing that must be taken into account for the pair programming to work properly is the personality of the partners. In our classes, we observed that students with a dominant profile tend to not accept suggestions and critics and try to resolve things by themselves. On the other hand, passive students tend to accept their partners solutions and avoid giving opinions, even when they notice something that is clearly wrong.

The last thing we observed in our classes is that there are some students that became addicted to pair programming and could not develop solo programming anymore. Probably this is the case of a passive student that agrees with everything even not understanding what the partner is doing.

CONCLUSIONS AND FUTURE WORK

In this article we described the pair-programming method when used as a learning tool and showed the relationship between this method and the collaborative learning theory and the Vygotsky's socio-cultural theory.

In general good results were achieved, and most of the students were satisfied with their own performance in the course. We noticed problems in some pairs due to great difference in knowledge, when one of the partners was too dominant or too passive, and when partners had personal differences.

Also, some students could not maintain the performance of pair programming when working alone. This fact may indicate that not all the problems would be solved in pairs; students have to have some problems to solve alone to identify their strong and weak points, and have a more realistic view of themselves.

In our institution, a psychological profile evaluation is made for all the students by a specialist, and for the next semester, this information will be used to try to avoid problems with dominant/passive partners in the working pairs. Also, the knowledge/skill levels should be used to form the pairs, and the first test of the semester can be used for this purpose.

REFERENCES

[1] VYGOTSKY, L. S. "A formação social da mente", *Martins Fontes*, São Paulo, 1994.

[2] Williams, Laurie A. & Kessler, Robert R. "Experimenting with industry's Pair Programming model in the computer science classroom", *Journal on Computer Science Education*, March 2001.

[3] Flor, N. V., & Hutchins, E. L. "Analyzing Distributed Cognition in Software Teams: A Case Study of Team Programming During Perfective Software Maintenance". *Paper presented at the Empirical Studies of Programmers: Fourth Workshop*, 1991.

[4] Williams, Laurie and Kessler, Robert R. "The Effects of Pair-Pressure and Pair-Learning on Software Engineering Education." *Conference of Software Engineering Education and Training*, 2000.

[5] Gokhale, Anuradha A. "Collaborative Learning Enhances Critical Thinking", *Journal of Technology Education.* Volume 7, Number 1 Fall 1995.(http://scholar.lib.vt.edu/ejournals/JTE/jte-v7n1/gokhale.jte-v7n1.html, (11/22/2002))

[6] http://www.minerva.uevora.pt/cscl.