

HERRAMIENTAS AUTOMÁTICAS DE APOYO AL APRENDIZAJE Y EVALUACIÓN EN ASIGNATURAS BASADAS EN PROYECTOS

Juan M. Montero, Javier Macías-Guarasa, Rubén San-Segundo, Ricardo de Córdoba, Javier Ferreiros

Dpto. Ingeniería Electrónica – ETSIT- UPM

{juancho, macias, lapiz, cordoba, jfl}@die.upm.es

Resumen

El aprendizaje basado en proyectos (PBL) es una de las técnicas docentes más interesantes en el campo de las enseñanzas técnicas. Sin embargo, para que la formación tenga éxito puede ser necesaria una gran carga de trabajo tanto de profesores como de alumnos, y puede llevar a estos a centrarse excesivamente en el objetivo de conseguir terminar el proyecto, sin prestar suficiente atención al modo en el que se debe realizar.

Para poder realizar un buen seguimiento y evaluación de una asignatura de proyectos software en Ingeniería de Telecomunicación, hemos estudiado cuantitativamente cuáles son los parámetros que contribuyen a dar calidad a un programa desarrollado en ensamblador, partiendo de las evaluaciones que llevan a cabo los profesores en cada convocatoria y aplicando técnicas estadísticas. Con los resultados obtenidos hemos construido un conjunto de herramientas automáticas que nos ayudan en el seguimiento, la orientación y la evaluación de los alumnos, tanto a lo largo del desarrollo del proyecto como a la hora de calificar. Finalmente hemos aplicado con éxito el sistema a la evaluación en una convocatoria.

1. INTRODUCCIÓN

En este trabajo se abordan algunas de las líneas de trabajo en el campo de la innovación educativa que fueron apuntadas en las conclusiones extraídas de la edición anterior de este congreso (XI CUIEET 2003). Por un lado se apostaba por la incorporación de nuevas tecnologías en el ámbito de la docencia, y por otro lado, se presentaba la técnica de aprendizaje basado en problemas o proyectos (PBL: Project Based Learning) como una solución muy interesante para fomentar la iniciativa de los alumnos. En este artículo se describen un conjunto de herramientas automáticas de análisis software que sirven de apoyo al profesor para realizar el seguimiento y evaluación de los alumnos en una asignatura basada en proyectos de ingeniería.

La técnica de aprendizaje basada en proyectos ha sido utilizada con un éxito relevante tanto en la docencia universitaria [1][2], como en cursos preuniversitarios [3][4]. En relación a la docencia universitaria se ha aplicado a una gran variedad de disciplinas como el derecho, la medicina [5][6]; pero donde mayor aplicación ha tenido ha sido en las enseñanzas técnicas [7][8][9][10][11]. Las comparaciones realizadas con la docencia tradicional revelan un mayor grado de aprendizaje en el caso de la técnica basada en proyectos [12][13], más aún cuando la aplicación de esta técnica se apoya en nuevas tecnologías [14][15].

El aprendizaje basado en proyectos permite fomentar la participación del alumno en el proceso de aprendizaje consiguiendo unos resultados mejores, tanto por los conocimientos como por los hábitos adquiridos por el alumno. Con esta técnica, el estudiante desarrolla nuevas capacidades que completan su formación y le preparan mejor de cara al mundo laboral. Entre estas capacidades destaca el trabajo e interacción en grupo, aprendizaje autónomo, asunción de responsabilidades y planificación del tiempo.

La aplicación de esta técnica de aprendizaje no está exenta de dificultades: mayor carga docente y trabajo para profesores y alumnos, mayor esfuerzo de organización y coordinación, y cierta inercia al cambio de algunos profesores. Por otro lado el proceso de calificación de los alumnos también supone una tarea de gran coste que puede hacer que el proceso de evaluación se centre en el funcionamiento del proyecto y no en el proceso seguido, por el grupo de alumnos, para llevar a cabo el proyecto. Una correcta evaluación requiere de un seguimiento de la evolución de los alumnos, seguimiento que supone un esfuerzo muy elevado, especialmente en asignaturas masivas como la nuestra (más de 175 grupos de trabajo).

En esta comunicación se describen un conjunto de herramientas que apoyan al profesor en el proceso de formación y evaluación de los grupos de alumnos. Estas medidas realizan tareas mecánicas cuantitativas, llevando a cabo mediciones objetivas de los principales parámetros de calidad de software. De esta forma, el profesor puede detectar fácilmente errores o malos hábitos de programación con un esfuerzo reducido.

2. LA ASIGNATURA LSED: LABORATORIO DE SISTEMAS ELECTRÓNICOS DIGITALES

Las herramientas desarrolladas se han aplicado en la asignatura Laboratorio de Sistemas Electrónicos Digitales (LSED). Esta asignatura es impartida por el Departamento de Ingeniería Electrónica [16] de la Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT) [17], de la Universidad Politécnica de Madrid (UPM) [18]. Este departamento imparte un conjunto amplio de asignaturas dedicadas al diseño de sistemas electrónicos basadas en microprocesadores. Este conjunto de asignaturas comprende una troncal de carácter teórico dentro del tercer curso de Ingeniería de Telecomunicación (SEDG: Sistemas Electrónicos Digitales), una obligatoria de carácter práctico dentro del mismo curso (LSED: Laboratorio de Sistemas Electrónicos Digitales), dos optativas de quinto curso de la especialidad de Electrónica (ISEL: Ingeniería de Sistemas Electrónicos y LSEL: Laboratorio de Sistemas Electrónicos) y también una asignatura de Libre Elección orientada a los alumnos instructores del LSED, todas ellas semestrales.

La asignatura LSED enlaza con la asignatura teórica previa (SEDG) que se centra en el mismo microprocesador y los mismos periféricos. En este laboratorio se busca un cierto equilibrio entre un alto contenido formativo con una carga de trabajo y aprendizaje que resulten abordables. Por otro lado, esta asignatura es la base del LSEL, asignatura de 5º curso de la especialidad de ingeniería electrónica.

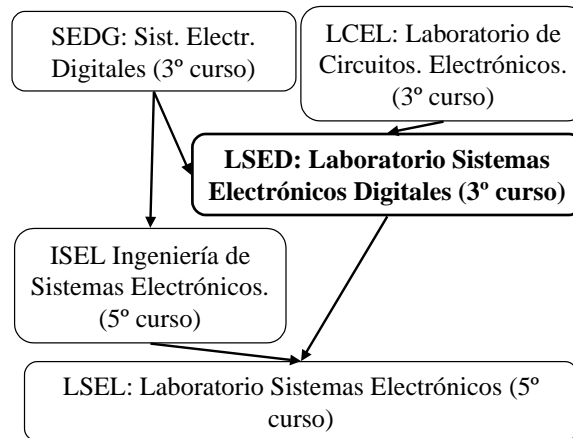


Figura 1: Relaciones entre asignaturas relacionadas con sistemas electrónicos. Se remarca en negrita la asignatura objeto de nuestro trabajo: LSED.

El LSED es una asignatura obligatoria que cursan cada año unos 350 alumnos. En ella, los alumnos deben diseñar, construir y probar un sistema completo compuesto por una parte hardware y otra software, ambas desarrolladas por ellos mismos agrupados en grupos de dos alumnos. Parten de un enunciado de cierta extensión (30-40 páginas) que incluye las especificaciones y requisitos del sistema (esto es, el ámbito, la descripción general y los casos de uso), así como parte del análisis (en forma de un conjunto de objetos o subsistemas de análisis con sus propiedades, métodos y relaciones principales) y parte del diseño (concretamente una arquitectura SW modular y obligatoria en cuanto a cómo distribuir las tareas entre el proceso principal y el proceso por interrupciones). A partir de esta información, el alumno debe acabar de analizar el sistema (toda especificación es siempre incompleta y no está totalmente determinada) y debe acabar de diseñarlo, implementarlo y probarlo.

El sistema solicitado cambia anualmente y cada grupo de alumnos debe desarrollar un prototipo funcional completo que es evaluado mediante:

- a) entregas electrónicas intermedias que ayudan a profesores y coordinador a comprobar la evolución y originalidad del trabajo,
- b) un informe escrito, donde se explique el análisis final, el diseño, la implementación y las pruebas realizadas y,
- c) un examen oral, donde el profesor comprueba que el prototipo cumple las especificaciones planteadas y donde formula preguntas individualizadas encaminadas a determinar la capacidad de cada alumno para explicar los resultados obtenidos.

El sistema tiene un carácter multidisciplinar puesto que la electrónica es un medio para construir sistemas de procesamiento de señal, de comunicaciones o de control de procesos cuya base matemática conoce el alumno de otras asignaturas no electrónicas. El sistema a desarrollar incluye siempre una componente de tiempo real (una parte importante de la funcionalidad se concentra en rutinas de atención a interrupciones periódicas), lo cual hace más compleja la depuración del sistema por parte de los alumnos y más exigente el desarrollo completo del prototipo, aunque se proporcionan recomendaciones

orientativas para que el alumno se enfrente específicamente al problema del tiempo real, la concurrencia y la compartición de recursos [8].

Se trata por tanto de una asignatura de laboratorio orientada a un diseño único (abierto a que los alumnos alcancen sus propias soluciones), de carácter sistémico y realista (aunque simplificado en la medida de lo posible, tanto económicamente como desde un punto de vista formativo). Aunque no es un laboratorio guiado, sí que se estructura el desarrollo del proyecto en varias sesiones y se establece una planificación temporal orientativa. Esta estructuración es importante para realizar un seguimiento de los alumnos y analizar su adecuación a la planificación prevista.

En la evaluación de la asignatura se prima la creatividad y la profesionalidad de cada grupo de alumnos: para alcanzar la calificación final máxima, el alumno debe realizar mejoras opcionales sobre el sistema básico propuesto (ampliaciones de la funcionalidad básica propuesta) que pueden suponer más de un 15 por ciento de la nota, o bien realizar una práctica especial con unas especificaciones o un microprocesador distinto a los propuestos en la práctica estándar. Se valoran además factores como la calidad de la escritura técnica (memoria técnica) o las capacidades para la comunicación oral (examen oral).

3. DESCRIPCIÓN DE LAS HERRAMIENTAS AUTOMÁTICAS

El desarrollo de herramientas automáticas para el seguimiento en tiempo real de la evolución de los alumnos, constituye un campo de innovación docente en el que se está invirtiendo un esfuerzo importante [19][20]. La principal razón es que el desarrollo de estas herramientas reduce enormemente la carga de los profesores, garantizando un mejor seguimiento de los alumnos. Estas herramientas son especialmente importantes en asignaturas masivas basadas en proyectos [8][9] y en aquellas en las que se quiere evaluar, no sólo el desarrollo del proyecto sino la colaboración entre los miembros del grupo de trabajo [21][22][23].

3.1. Herramientas de Gestión

Las herramientas de gestión constituyen una nueva versión de las herramientas descritas en [20] y que permiten las siguientes acciones:

- Gestión de las inscripciones de los alumnos y asignación de turnos (día de la semana y hora) y puestos del laboratorio.
- Generación de listas de los alumnos.
- Planificación de los exámenes de los profesores según la carga docente asignada.
- Gestión de calificaciones: introducción por parte de los profesores y cálculo de estadísticas generales y por profesor.
- Listas de correo basadas en el sistema de gestión mailman.
- Mantenimiento de páginas de consulta sobre preguntas y dudas frecuentes.
- Servicios para los alumnos: información y reserva de puestos libres, y acceso a las calificaciones obtenidas.

Esta nueva versión incorpora, además, la posibilidad de que los alumnos realicen la encuesta sobre la asignatura vía web, lo que permite el procesado automático posterior: para cada uno de los profesores y en cada uno de los aspectos de la asignatura evaluados.

3.2. Herramientas de Adquisición de Datos

Unidas a las herramientas de gestión comentadas anteriormente, se ha desarrollado un nuevo conjunto de herramientas (también vía web) con el fin de capturar la información necesaria para realizar el seguimiento de los alumnos. Estas herramientas permiten realizar las siguientes acciones:

- Controlar la asistencia de los alumnos al laboratorio tanto en sus turnos asignados como en los turnos libres o extra.
- Conocer la última sesión alcanzada (dentro de la estructuración propuesta por los profesores) en cada asistencia al laboratorio.
- Realizar las entregas electrónicas parciales del software desarrollado hasta la fecha. Generalmente se realizan entre 4-5 entregas a lo largo del semestre. Como veremos más adelante, estas entregas parciales, permiten realizar evaluaciones del software para detectar posibles errores o malos hábitos de programación durante el proceso de desarrollo del proyecto, así como detectar posibles intentos de copia por parte de algunos alumnos.
- Realizar la entrega electrónica final del proyecto, tanto del software desarrollado como de la memoria técnica descrita. La aplicación de las herramientas de análisis de software sobre la última entrega realizada permitirá obtener una gran cantidad de medidas que esta vez no se utilizarán (por parte de los profesores) para orientar o reconducir al grupo de alumnos, sino para realizar su evaluación final. En esta evaluación no sólo se tiene en cuenta el análisis de la versión final sino también la evolución del grupo de alumnos a lo largo del semestre (entregas intermedias).

Mediante estas herramientas es posible disponer de los datos necesarios para realizar un seguimiento, en tiempo real, de la evolución de los alumnos.

3.3 Datos adquiridos

En relación con los datos recopilados hasta la actualidad, disponemos de la siguiente información organizada por cursos.

En el curso 2002-03 se desarrolló una calculadora parlante basada en el MC68000. Esta calculadora permitía sumar y multiplicar números introducidos mediante de un teclado matricial, y pronunciarlos conforme se van tecleando por medio de voz pregrabada y un DAC externo. En este caso se dispone de las entregas intermedias y final de software, y la calificación final obtenida.

En el presente curso 2003-04 se ha implementado un chat con enlace de infrarrojos basado en el MC68000. Este sistema permite la introducción de una cadena de caracteres mediante un teclado matricial con la función de multisímbolo por tecla (similar a los teclados de los teléfonos móviles), la transmisión de dichos caracteres a través de un enlace de infrarrojos y su recepción. El protocolo de comunicación utilizado es muy sencillo y contiene un

único bit de arranque, un bit de paridad (paridad par) y un bit de parada para cada uno de los caracteres transmitidos. En este caso disponemos de toda la información posible: asistencia al laboratorio, evolución del desarrollo del proyecto de cada grupo, análisis de las entregas intermedias, análisis de la entrega final del software y la calificación obtenida.

3.4. Tecnologías de soporte de las herramientas y requisitos del sistema

En la versión actual, el sistema está soportado por las siguientes tecnologías:

- Generación dinámica de contenidos con PHP.
- Base de datos MySQL.
- Protocolos HTTP y HTTPS, usando un servidor apache.
- Software de soporte escrito en C, bash, yacc, lex y perl.

Dada la relativamente baja carga computacional que implica, no es necesaria su implantación sobre una plataforma hardware de última generación. Los requisitos del sistema necesarios para implantar las herramientas desarrolladas son las siguientes:

- Ordenador PC compatible Pentium III (con requisitos adicionales para poder ejecutar e instalar los paquetes que se describen a continuación).
- Sistema operativo Linux con kernel 2.4.18 o superior [13].
- Servidor apache versión 1.3.26 o superior, con soporte SSL y módulos PHP y MySQL.
- Intérprete PHP3 versión 3.0.18 o superior.
- MySQL versión 3.23.49 o superior, aplicaciones cliente y servidor.
- PERL v5.8.0.

4. ANÁLISIS DE LA CALIDAD DEL SOFTWARE

No resulta fácil definir con precisión en qué consiste la calidad de un programa, aunque los profesionales con experiencia sí que la pueden estimar de una manera fiable empleando su conocimiento de experto. Así para soslayar el problema de la definición formal explícita de la calidad, podemos aprovechar de una manera directa el conocimiento experto de los profesores de nuestro laboratorio, expresado a través de las evaluaciones que realizan cada curso cuando los alumnos concluyen su trabajo. De esta manera el análisis de la calidad de un programa puede verse como un caso particular del problema general de análisis y clasificación de patrones (*pattern-matching*). Un clasificador consta de:

- una etapa de extracción de características, que debe reducir el programa a un conjunto de características mensurables cuyos valores contribuyan a diferenciar un programa de calidad de un programa que no la tenga. Estos valores forman un vector que caracteriza el programa y que permite compararlo con otros programas cuya calidad sea conocida;
- un conjunto de programas de referencia convenientemente evaluados por un experto, y cuyos vectores de características servirán de patrones para el proceso de comparación. Este conjunto, usualmente denominado base de

datos, sirve para estimar los parámetros que definen la distancia entre patrones, de tal manera que aquellas características más relevantes desde el punto de vista de la evaluación se vean reforzadas (tengan un mayor peso en la comparación) respecto a las menos relevantes (que no irrelevantes). Este proceso se conoce como entrenamiento del clasificador;

- una etapa de evaluación que, en función de las características obtenidas, de las referencias disponibles y de la distancia previamente entrenada, estima la calidad del programa por medio de un proceso de comparación de patrones.

4.1. Características relacionadas con la calidad de software

Muchas son las características cuantificables que podemos extraer de un programa y que pueden estar relacionadas con la calidad del software. En la fase inicial de entrenamiento debemos reunir el mayor conjunto posible de características y estimar su relevancia según lo evaluado por los profesores de la asignatura. En nuestro trabajo hemos analizado hasta 48 características básicas de un programa en ensamblador:

- el aprovechamiento de los recursos de la CPU: esto es, de los registros de datos y de direcciones, de la variedad de instrucciones disponibles, el uso de los distintos modos de direccionamiento;
- las estructuras de datos empleadas: el número de variables declaradas y usadas, el número de constantes, de tablas o de mensajes;
- características estructurales como el número y longitud media de las subrutinas o de la rutina de atención a la interrupción, el número medio de puntos de terminación de las rutinas, la longitud media y máxima de los saltos o el grado de discontinuidad de las subrutinas;
- los comentarios incluidos en el código, tanto comentarios de bloque como comentarios de línea.

A partir de los datos adquiridos en la convocatoria de junio del curso 2002-2003 y estudiando la correlación de Pearson entre los valores de las características y la evaluación asignada por los profesores a cada programa, hemos descubierto las siguientes relaciones:

- **Uso de los modos de direccionamiento:** intuitivamente considerábamos que el uso de los más complejos (como el indirecto o el indirecto indexado que permiten acceder a tablas o recorrer listas de datos) revelaría una mayor calidad. Lo más interesante es que el modo indirecto prácticamente no correla con la nota final ($<0,05$) y lo hace negativamente, mientras que el modo indexado correla positivamente ($>0,20$): los que más lo usan obtienen mejores notas en promedio. Los modos con predecremento y postincremento correlan sólo marginalmente y de forma negativa ($<-0,05$). Los modos directo a registro o inmediato no correlan de manera positiva ni relevante. En cambio, el número de modos de direccionamiento diferentes que haya empleado el alumno sí que es un indicio de calidad ($>0,20$).
- **Aprovechamiento de otros recursos de la CPU:** el número de registros diferentes empleados por el alumno correla bastante positivamente con su

nota final ($>0,15$). El número de instrucciones diferentes incluidas, por el contrario, tiene una importancia marginal ($<0,05$).

- **Estructuras de datos:** el número de variables o el modo en que se declaran, apenas correla con la nota obtenida ($<0,05$). Un mayor empleo de constantes, sin embargo, es un indicio de mejor calidad global ($>0,05$). El elemento más relevante en este apartado es el uso de estructuras de datos más complejas como listas, tablas o mensajes, que alcanzan una correlación superior al 0,20 con la nota final obtenida.
- **Características estructurales:** el número de subrutinas incluidas es una característica fundamental a la hora de predecir la calidad de un programa ($>0,35$). Sorprendentemente, el número de líneas de un programa también correla muy positivamente con la nota (0,35). No tienen mucha importancia elementos tales como el tamaño de la rutina de atención a la interrupción o el tamaño medio de las subrutinas, aunque resultan negativos. El tamaño de la subrutina más larga correla negativamente con la nota final ($<-0,15$): la existencia de una subrutina muy larga se traduce en una peor calificación en media; también correlan negativamente la distancia del salto condicional o incondicional más largo ($<-0,05$) o la distancia media de los saltos ($<-0,30$). El número de puntos de salida que tiene en promedio cada rutina correla positivamente ($>0,05$), así como el número de saltos incluidos ($>0,20$). La presencia de subrutinas discontinuas o con saltos hacia atrás apenas tiene influencia.
- **Comentarios:** su relevancia se revela como tan sólo marginal ($<0,05$)

4.2. Herramientas de valoración automática

En una primera aproximación, hemos diseñado un clasificador lineal sencillo y efectivo, basado en un conjunto de pesos que, al multiplicarse escalarmente por el vector de características, dan como resultado una estimación de la calificación que merecería el alumno de acuerdo con el análisis anteriormente expuesto.

Un primer uso de esta herramienta tuvo lugar en la revisión de calificaciones de la convocatoria analizada: a los alumnos que estaban disconformes con su nota se les mostraba las características objetivas medidas en su programa y que habían reducido su nota, así como su comparación con la media de los alumnos. De esta manera lográbamos objetivar y cuantificar la evaluación de la calidad del software, sin que fuese necesario que interviniese el profesor calificador en el proceso de revisión.

En la convocatoria de septiembre (donde el sistema que desarrollar era el mismo, salvo en una pequeña variante), los pesos aprendidos en la convocatoria de junio, sirvieron de orientación al profesor que examinó a los alumnos, no produciéndose ninguna solicitud de revisión, aliviando parcialmente la carga de dedicación del profesor y aprovechando al máximo la experiencia de la convocatoria de junio.

5. DISCUSIÓN

Algunos de los resultados obtenidos en el apartado 3.4.1 nos resultaron contraintuitivos y merecen una discusión de carácter cualitativo.

- **Uso de los modos de direccionamiento:** la mayor relevancia del direccionamiento indexado está relacionada con su uso para implementar tablas o recorrer listas; para muchos alumnos resulta el modo de direccionamiento más complejo, ya que involucra simultáneamente un registro de direcciones y uno de datos, así como diversos tamaños (el del dato, el del registro de direcciones y el del registro de datos). Todo ello explica que en general sólo los mejores alumnos lo empleen con fluidez, mientras que el resto de los alumnos prefieran evitarlo. La mayor calidad asociada al modo indirecto indexado provoca que el resto de los modos indirectos se asocien a las prácticas de menor calidad. El número total de modos de direccionamiento empleados, valorado de manera también positiva, responde al mismo fenómeno.
- **Otros recursos de la CPU:** los mejores alumnos, los que realizan prácticas más complejas, parecen emplear una mayor variedad de instrucciones pero, sobre todo, emplean más registros: los alumnos con menos dominio de los recursos parecen emplear siempre los mismos recursos, aquellos que les dan más seguridad a la hora de programar.
- **Estructuras de datos:** el empleo de un mayor número de constantes está demostrado que conduce a programas de mayor calidad, entendida esta no sólo como un menor tiempo de desarrollo, sino también como una mayor facilidad para adaptar o reconfigurar el software y afrontar extensiones a la funcionalidad básica (algo muy apreciado en una asignatura de sistemas como la nuestra). Pero la característica más relevante es que el empleo de tablas y mensajes se traduce en programas de mayor calidad; las tablas están relacionadas con procedimientos más elegantes y compactos para abordar numerosos problemas, mientras que los mensajes se asocian con una mejor política de detección y señalización de situaciones de error (que conducen a un mejor sistema desarrollado).
- **Características estructurales:** como era de esperar, los programas con más subrutinas se destacan por su calidad, y la presencia de una subrutina muy larga destaca justo por lo contrario; sin embargo, esto no se traduce en que la longitud media de las subrutinas sea más corta en los mejores programas: esto se debe a que los alumnos con menos subrutinas suelen además hacerlas muy cortas, suelen realizar programas más cortos (porque no realizan tantas extensiones de funcionalidad) y suelen incluir bastante código directamente en el programa principal y en la rutina de atención a la interrupción (código que no computamos como código de subrutina). La relación existente entre el número de líneas del programa y su calidad tiene realmente que ver con que los programas con extensiones de funcionalidad son más largos y están más valorados (no con que las soluciones largas sean mejores que las más compactas).
- **Salto condicionales o incondicionales:** el manejo de los saltos también está estilísticamente marcado y los peores alumnos realizan programas menos estructurados y por lo tanto presentan saltos condicionales o incondicionales más cortos, a pesar de que usan más porcentaje de instrucciones de salto (esto está relacionado con programas más complejos que cuentan con un mayor número de condiciones).

- **Los comentarios:** los evaluadores no han puntuado mejor los programas con más comentarios, posiblemente porque los alumnos con mejor software se han concentrado más en ampliar la funcionalidad del programa que en comentar de una manera muy profusa.

6. CONCLUSIONES Y LÍNEAS FUTURAS

Hemos conseguido desarrollar un conjunto de herramientas automáticas que ayuden a profesores y alumnos a abordar con éxito una asignatura basada en un proyecto (dominantemente software). Las herramientas desarrolladas nos permiten recoger los programas que van desarrollando los alumnos, analizarlos y compararlos con los patrones de calidad adquiridos en el curso anterior: con esta información los alumnos pueden comprobar cómo es la calidad estimada de su programa al compararlo con la media de los de sus compañeros. Así en el presente curso hemos podido emplear las herramientas para que los alumnos pudiesen conocer si el estilo de programación que empleaban era adecuado, cuando aún estaban a tiempo de corregir sus defectos.

El análisis de los parámetros que contribuyen a dotar de calidad a un programa también nos ha permitido crear un evaluador automático que orienta al profesor en su tarea de evaluación de una manera objetiva y cuantitativamente precisa, permitiéndole que, al rebajar su carga de trabajo, se fije más en aspectos cualitativos más sutiles. Igualmente se consigue reducir las diferencias de criterio entre profesores, inevitables al afrontar una evaluación tan compleja; esto es especialmente útil cuando algunos profesores tiene menos experiencia o, por su carga docente, sólo examinan a unos pocos alumnos y conocen menos el contexto global a la hora de evaluarlos: el programa les ayuda a situar a sus alumnos dentro del contexto general del curso y dentro del contexto de la asignatura en cursos precedentes.

En el análisis de parámetros realizado hemos corroborado que las características más relevantes son: emplear bastantes subrutinas, que ninguna de ellas sea muy larga y que los saltos que se realicen sean más bien cortos, para así estructurar adecuadamente la solución; usar más estructuras de datos complejas, que en general dan lugar a soluciones software más elegantes; emplear más modos de direccionamiento y, en especial, el modo indirecto indexado, todo ello también relacionado con las estructuras de datos y los algoritmos elegantes; definir constantes no literales, que permiten reconfigurar el programa con facilidad y contribuyen a su mantenibilidad; emplear más registros diferentes, signo de dominio de la máquina y un mejor aprovechamiento de sus recursos.

De manera aparentemente paradójica hemos descubierto que: el resto de los modos de direccionamiento indirectos no se traducen en mayor calidad en términos relativos; emplear más saltos es un indicio de calidad, quizá relacionado con una estructura más rica de la solución; la cantidad de comentarios no ha sido muy valorada por los profesores, y quizá sea necesario darle más peso en la orientación que se da a los alumnos y en los criterios de evaluación de los profesores.

El análisis realizado nos ha llevado a reforzar las orientaciones que cada curso damos a nuestros alumnos en cuanto a aspectos de calidad de software, con lo que esperamos que la calidad de los programas entregados por los alumnos

mejore respecto a la convocatoria anterior, con lo que habremos conseguido una mejor formación con un esfuerzo similar.

Con las calificaciones que se asignarán en la convocatoria de Junio de este curso 2003-2004 podremos estudiar si el clasificador entrenado el curso pasado es aplicable a otro problema y resulta general, o si es necesario reentrenarlo para que resulte independiente del problema planteado.

Agradecimientos

A todos los miembros del Departamento de Ingeniería Electrónica de la Universidad Politécnica de Madrid que han colaborado en la docencia del Laboratorio de Sistemas Electrónicos Digitales, porque con sus comentarios y con su dedicación han participado indirectamente en la realización de este trabajo.

Referencias

- [1] Barrows, H. S. (1996). Problem-based learning in medicine and beyond: A brief overview. In L. Wilkerson & W. H. Gijsselaers (Eds.) Bringing problem-based learning to higher education: Theory and practice (pp. 3-12). San Francisco: Jossey-Bass.
- [2] Solomon, Gwen. 2003. Project-Based Learning: a Primer. Technology and Learning, January 2003. Vol. 23. No. 6.
- [3] Katz, L.G. and S.C. Chard. (1989). Engaging Children's Minds: the Project Approach. Norwood, NJ: Ablex.
- [4] Chard, Sylvia C. (1992). The Project Approach: A Practical Guide for Teachers. Edmonton, Alberta: University of Alberta Printing Services.
- [5] Albanese, M. A. & Mitchell, S. (1993). Problem-based learning: A review of literature on its outcomes and implementation issues. Academic Medicine, 68(1), 52-81.
- [6] Vernon, D. T. A. & Blake, R. L. (1993). Does problem-based learning work? A meta-analysis of evaluation research. Academic Medicine, 68(7), 550-563.
- [7] Alcober, J., Ruiz, S., Valero, M., "Evaluación de la implantación del aprendizaje basado en proyectos en la EPSC (2001-2003)" en Actas del XI Congreso Universitario de Innovación educativa, Vilanova i la Geltrú Julio de 2003.
- [8] J.M. Montero, J. Ferreiros, J. Macías Guarasa, R. de Córdoba y J.D. Romeral "Enseñanza en Laboratorios de Electrónica: Una filosofía basada en diseños no guiados del mundo real" en Actas del XI Congreso Universitario de Innovación educativa, Vilanova i la Geltrú Julio de 2003.
- [9] An undergraduate microcontroller systems laboratory. Hedley, M.; Education, IEEE Transactions on Education, Volume: 41, Issue: 4, Nov. 1998.
- [10] S.A. Ambrose and C.H. Amon "Systematic design of a first-year mechanical engineering course at Carnegie-Mellon University" J. Eng. Educ. Vol 86, pp. 173-182, April 1997.

- [11] J.G. Harris (Moderator), "Journal of engineering education round table: reflexions on the grinter report". J. Eng. Educ. Vol 83, no 1, pp. 69-94, Jan. 1994.
- [12] Ryser, G. R, Beeler, J. E., & McKenzie, C. M. (1995). Effects of a Computer-Supported Intentional Learning Environment (CSILE) on students' self-concept, self-regulatory behavior, and critical thinking ability. Journal of Educational Computing Research 13(4), 375-385.
- [13] Pellegrino, J.W., Hickey, D., Heath, A., Rewey, K., & Vye, N. J. (1992). Assessing the outcomes of an innovative instructional program: The 1990-1991 implementation of the "Adventures of Jasper Woodbury." Nashville, TN: Vanderbilt University, Learning Technology Center.
- [14] Means, B. & Olson, K. (1997). Technology and education reform (ORAD 96-1330). Washington, DC: U.S. Government Printing Office.
- [15] Coley, R. J., Cradler, J., & Engel, P. K. (1996). Computers and classrooms: The status of technology in U.S. schools (Policy information report). Princeton, NJ: Educational Testing Service.
- [16] <http://www.die.upm.es>
- [17] <http://www.etsit.upm.es>
- [18] <http://www.upm.es>
- [19] Jiménez-Leube, F.J., Almendra, A., González, C. y Sanz-Maudes, J. Networked implementation of an electrical measurement laboratory for first course engineering studies en IEEE Transactions on Education, Vol. 44(4), pp, 377-388, November. 2001.
- [20] Macías Guarasa, J., Montero, J.M., Ferreiros, J., Romeral, J.D. y Córdoba, R. Herramientas web de ayuda para la gestión automática de laboratorios masivos. en Actas del XI Congreso Universitario de Innovación educativa, Vilanova i la Geltrú Julio de 2003.
- [21] Barros, M., Verdejo, M., (2000). Analysing student interaction processes in order to improve collaboration. The DEGREE approach. Int. Journal of Artificial Intelligence in Education. (2000). 11, 221-241.
- [22] Martínez, A., Dimitriadis, Y., Rubia, B., Gómez, E., Garachón, I., Marcos, J.A. (2002). Studying social aspects of computer-supported collaboration with a mixed evaluation approach. Proc. of the Int. Conf. On CSCL, 2002, Boulder, Colorado, USA, 631-632.
- [23] Zumbach, J., Mühlenbrock, M., Jansen, M., Reimann, P., Hoppe, H.U., (2002). Multidimensional Tracking in Virtual Learning Teams. In G. Stahl (Ed.), Computer Support for Collaborative Learning: Foundations for a CSCL community. Hillsdale, NJ. Erlbaum, pp. 650-651.