Iniciativas para motivar a los alumnos de Programación

Maria Salamó, Joan Camps, Carles Vallespí, David Vernet, Xavier Llorà, Ester Bernadó, Josep Maria Garrell y Xavier González
Departamento de Informática
Enginyeria i Arquitectura la Salle
Universitat Ramon Llull
Passeig Bonanova, 8,
08022 Barcelona
e-mail: {mariasal, joanc, cvalles, dave, xevil, esterb, josepmg, xgonzalez}@salleurl.edu

Resumen

La asignatura de Programación constituye una base fundamental para las diversas carreras de ingeniería. Los posibles enfoques que se pueden utilizar, tanto de contenidos como de método docente, son muy diversos. En este artículo presentamos la solución adoptada en Enginyeria i Arquitectura la Salle (Universitat Ramon Llull) para ayudar a los alumnos en el estudio de esta asignatura.

El principal objetivo es dar una herramienta de ayuda para que el alumno se sienta motivado con la asignatura y amplíe sus horas de estudio, con la finalidad de afianzar los conocimientos.

1. Motivación

La asignatura es común a los diversos planes de estudios que se imparten: Ingeniería de Informática, Ingeniería de Telecomunicaciones, Ingeniería de Telemática, Ingeniería Electrónica, y Graduado en Multimedia. Por este motivo la asignatura es producto de un compromiso entre las diversas necesidades de cada carrera.

El hecho de que la asignatura sea común a diversas carreras, conlleva la necesidad de hacer coincidir objetivos dispares. Por este motivo el temario y el método docente son el producto de un compromiso, y se organizan para cubrir las diversas necesidades de los distintos planes de estudio. La solución adoptada se planteó en [1]. Actualmente, después de comprobar la progresión de los alumnos en nuestro plan de estudios combinado para diversas ingenierías, hemos buscado nuevos enfoques de la asignatura para motivar a los alumnos en su dedicación en horas de estudio.

El objetivo prioritario es motivar al alumno a conseguir el máximo rendimiento en la asignatura y a afianzar gradualmente los conocimientos sin depender de la proximidad del examen.

La asignatura, de nueve créditos repartidos entre teoría y prácticas tiene un carácter anual. La parte teórica se evalúa trimestralmente en tres exámenes repartidos a lo largo del curso académico. La parte práctica consta de 4 prácticas desarrolladas en C y C++. Actualmente, la asignatura consta de 680 alumnos matriculados.

El presente artículo se divide en los siguientes apartados. La sección 2 introduce la descripción de la asignatura donde se exponen los contenidos de la asignatura y la problemática que queremos solventar. A tal efecto, hemos adoptado 3 medidas complementarias. La primera de ellas, se introduce en la sección 3, donde presentamos la página WEB de la asignatura. La siguiente sección, la segunda medida, se centra en la evaluación continua. La última medida, en la sección 5, presenta el simulador de estructuras de datos SimEd. Finalmente, la sección 6 describe las conclusiones y líneas futuras.

2. Descripción de la asignatura

El objetivo global de la asignatura de Programación es introducir al alumno en el diseño y programación de algoritmos. En este proceso, se da énfasis tanto al planteamiento global del

problema, cómo a la fase de diseño y a la implementación.

Dada la gran diversidad de lenguajes de programación, metodología de diseño y técnicas de programación, se ha creído conveniente marcar unos objetivos de contenido. Dichos objetivos tienen que permitir al alumno tener una visión global de la programación, desligada de las herramientas específicas.

Además de los objetivos referentes al contenido de la asignatura, se plantean también una serie de objetivos didácticos, que permitan al alumno adquirir una metodología de trabajo. Este último punto, es uno de los más importantes dentro de nuestra experiencia docente. El alumno necesita adquirir una metodología de trabajo.

2.1. Contenido de la asignatura

En este subapartado se introducen todos aquellos conceptos de contenido que se consideran fundamentales dentro de la asignatura de programación [6] [5]. Los contenidos hacen referencia al paradigma de programación escogido, el lenguaje utilizado en clase para introducir los diferentes conceptos, la introducción a los tipos abstractos de datos y a la orientación a objetos.

Para la introducción de los conceptos básicos de programación, se utiliza pseudocódigo. De esta forma, se independiza el aprendizaje de los conceptos o esquemas generales de la programación, de los detalles de un lenguaje de programación [3]. De los diferentes paradigmas de lenguajes de programación (imperativos, funcionales, lógicos, etc.), la asignatura se centra en dos de ellos: el imperativo y el de orientación a objetos. De hecho, se enfatiza el modelo imperativo, ya que didácticamente se considera más sencillo como punto de partida para alumnos que se inician en la programación. El lenguaje imperativo utilizado es el lenguaje C [8].

Una vez se han explicado los conceptos básicos de la programación imperativa, y teniendo en cuenta que el alumno tendrá que enfrentarse a aplicaciones de una complejidad considerable, es necesario introducir el diseño modular como una base imprescindible para tratarla.

De forma complementaria al diseño modular, y para facilitar al alumno la posterior comprensión de la programación orientada a objetos, se introduce el concepto de Tipo Abstracto de Datos (TAD) [4]. El hecho de utilizar TADs facilita la posterior introducción de estructuras de datos como entidades encapsuladas y reutilizables. Asimismo, introducimos en el curso las estructuras de datos lineales básicas, como la pila, la cola y la lista. Es muy importante hacer comprender al alumno la estrecha relación que existe entre el diseño modular y los TADs.

La orientación a objetos [2] se ha convertido durante los últimos años en una herramienta muy útil para tratar la complejidad de las aplicaciones. Por este motivo se considera importante introducir este paradigma a los alumnos, de manera que conozcan conceptos tales como encapsulación, abstracción, herencia, polimorfismo, etc. El lenguaje de programación utilizado es el lenguaje C++ [9] [10].

2.2. Problemática a tratar

Nuestro mayor problema ha sido encontrar fuentes de motivación para nuestros alumnos. Se observa que la mayoría de los alumnos no tienen un hábito de trabajo constante. Además, debe añadirse el cambio que supone la nueva dinámica de las sesiones en la Universidad, así como la distribución de los exámenes a lo largo del curso.

Este problema lleva a nuestros alumnos a abandonar la asignatura prematuramente al considerarla demasiado difícil debido a su falta de hábito en asimilar conceptos abstractos y en su posterior uso en la resolución de problemas. Los alumnos buscan analogías directas entre problemas resueltos en clase respecto a un problema nuevo que se les plantea. Normalmente no tienden a aplicar los conceptos teóricos sino que aplican las mismas “recetas” en el mismo orden.

Dada esta problemática, nos planteamos dedicar mayor atención a la motivación del

alumno. Esto nos llevó a aplicar las siguientes iniciativas:

- Creación de una *página WEB* con diferentes contenidos didácticos. Entre ellos encontramos: ejercicios, temario resumido, problemas resueltos, etc.
- Introducción de una *evaluación continua* de los alumnos. Esto los acerca más a la filosofía de trabajo que tenían adquirida en niveles de estudio inferiores y también incentiva el trabajo continuo del alumno.
- Creación de un *simulador de estructuras lineales con soporte WEB* para poder practicar gráficamente con algunos de los contenidos explicados en el temario.

Con estas iniciativas nos proponemos:

- Conseguir que el alumno desarrolle la capacidad de trabajo continuado, aunque la fecha de los exámenes sea lejana.
- Potenciar al alumno en su trabajo, proporcionándole materiales didácticos que le motiven a intentar realizar todas las tareas relacionadas con la asignatura. Buscamos que tengan interés en la asignatura y que con la nueva tarea intenten comprobar si han adquirido los conocimientos necesarios para superarla.
- Ayudar al alumno a comprobar si los conocimientos se han afianzado suficientemente para resolver cualquier problema planteado.
- Mejorar la capacidad de respuesta del alumno ante un problema. Típicamente la capacidad de respuesta de un alumno ante un problema nuevo suele ser mayor que cuando ha adquirido los conocimientos necesarios. De esta manera, tendrán mejor preparación de cara al futuro examen.
- Conocer la evolución de los alumnos a lo largo del curso. La evaluación continua nos permite conocer si un alumno necesita ejercicios de refuerzo o si, por el contrario, su trabajo se desarrolla dentro de la normalidad establecida para el curso docente.

3. Página WEB

El desarrollo de la página WEB de la asignatura (*www.salleurl.edu/Eng/Assignatures/Programacio1*) consistió en buscar una área común para los alumnos y los profesores de la misma.

La página WEB nos ha permitido establecer un canal de comunicación entre profesores y alumnos. A los alumnos les permite acceder fácilmente a los diferentes contenidos de la asignatura, temarios, ejercicios, etc. y a los profesores nos permite tener un medio de comunicación directo sobre nuestros alumnos cada vez que queremos proporcionales un nuevo material.

Los contenidos de la página WEB se dividen en dos secciones:

- Parte *pública*: se presenta toda la información general de la asignatura
- Parte *privada* o interna (sólo para alumnos matriculados y profesores)

La parte pública define los siguientes contenidos:

- *Presentación*: muestra información general de la asignatura, a quien está dirigida, el enfoque general de la misma (lenguajes utilizados, metodología de trabajo, etc.), qué objetivos deben alcanzar los alumnos al finalizar el curso docente y la división de la asignatura en parte teórica y parte práctica.
- *Temario*: se presenta detallado cada tema que se imparte en la asignatura, definiendo además, cuáles serán los puntos estudiados en cada uno de ellos.
- *Bibliografía*: recopilación de referencias bibliográficas necesarias a lo largo del curso docente. La bibliografía se ha dividido en tres partes: la necesaria para las clases teóricas, para las prácticas de C y C++.
- *Profesores*: presentamos cada uno de los profesores de la asignatura, junto con sus datos de ubicación dentro de la universidad, así como el correo electrónico.
- *Horarios*: presentamos los horarios de clase de todos los profesores, horarios de laboratorio y los horarios de consultas de cada profesor.
- *Conexiones*: links a otras páginas de interés sobre temas relacionados con la asignatura. Entre los links encontramos páginas de otras universidades, páginas de bibliotecas próximas, etc.

La parte privada o interna es accesible sólo para alumnos y/o profesores de la asignatura. Dentro de la parte privada encontramos:

- *Alumnos*: permite a los alumnos controlar sus datos personales y *consultar las notas* que tienen hasta un determinado momento. Por otro lado, un alumno puede *acceder cada semana a un ejercicio* de evaluación continua, relacionado con el temario explicado la semana anterior. *Acceso a las sesiones de laboratorio*, entre los materiales se puede encontrar los diferentes ejercicios propuestos durante una sesión de laboratorio, los enunciados de las prácticas y los formatos para realizar las pruebas de algunas prácticas. Finalmente, también pueden consultar las noticias relacionadas con la asignatura

- *Intranet Profesores*: nos permite tener acceso a la información de nuestros alumnos. La información de los alumnos se puede visualizar en formato individual o global.

  El formato individual, muestra la ficha técnica del alumno junto con su fotografía. La ficha técnica es el conjunto de sus notas, tanto teóricas como prácticas y sus datos personales.

  El formato global, muestra tanto los datos para todo un grupo de teoría como para todos los alumnos matriculados en la asignatura. Se pueden extraer listados en diferente formato, mostrando o no las fotografías de los alumnos.

  Otra utilidad es el control estadístico que realizamos de la asignatura. Una vez coleccionados los datos de diferentes años, podemos ver el transcurso de los alumnos a lo largo de los mismos. También podemos extraer estadísticas de los alumnos en un curso docente a lo largo de las diversas convocatorias de exámenes.

4. Evaluación continua

La evaluación continua se planteó para incentivar a los alumnos en el trabajo regular de la asignatura, ya que en cursos anteriores éste había sido el mayor problema. Los alumnos no mantienen una dinámica de trabajo constante. Esto hace que al llegar las proximidades de los exámenes se encuentren abrumados al descubrir la gran cantidad de conceptos que deben asumir. Esto provoca el abandono de la asignatura justo antes de la realización del examen.

La evaluación continua quiere resolver este problema. Nuestro mayor objetivo es conseguir una dinámica de trabajo continuado para evitar el abandono justo antes de los exámenes o al final del curso.

La asignatura debe ser superada en su parte teórica y práctica por separado.

La evaluación continua esta formada por tareas de diversa índole:

- *Ejercicios*: se proponen ejercicios del mismo libro de problemas de la asignatura para la siguiente semana. Estos ejercicios permiten conseguir una parte de la nota de evaluación continua.
- *Problema de la semana*: en la página WEB se proponen ejercicios semanalmente. Estos ejercicios están relacionados con el temario impartido la semana anterior en las clases teóricas.
- *Controles*: durante todo el trimestre, se dedica una o dos horas para realizar alguna prueba, ya sea tipo test o un pequeño ejercicio.
- *Trabajo en clase o laboratorio:* se proponen ejercicios específicos sobre algún concepto que deben realizar en las clases teóricas o prácticas. En el caso de los ejercicios realizados en el laboratorio, estos están enfocados a practicar algún concepto desde el paradigma de programación con el que se está trabajando en ese momento.

La evaluación continua facilita al alumno marcar un ritmo constante de trabajo a lo largo del curso. Además les proporciona un mecanismo para conocer sus dificultades con cada uno de los temas explicados. Si un alumno descubre a tiempo que no ha asumido un concepto, puede preguntar dudas a los profesores, pedir más ejercicios de apoyo y/o buscar bibliografía sobre el tema para profundizar.

El beneficio para los profesores es doble. En primer lugar nos permite conocer cuál es la evolución del alumno en la asignatura. En segundo lugar, nos permite descubrir cuáles son los puntos que plantean más dificultades a los alumnos. Así podemos introducir nuevos ejercicios, replantear el porcentaje de horas lectivas en algún concepto y/o plantear algún ejercicio extra, vía WEB o laboratorio.

5. Simulador SIMED

Para reforzar la comprensión de las estructuras lineales de datos, que se consideran como temario imprescindible de la asignatura, se ha llevado a cabo el diseño y desarrollo de una aplicación destinada a este fin [7]. La figura 1 muestra la pantalla principal del simulador. La figura 2 muestra un ejemplo de simulación para una estructura de datos, en esta figura se pueden observar los diferentes botones de ejecución para realizar diversos movimientos sobre la estructura de datos.

Figura 1. Pantalla principal del simulador.

Esta aplicación, llamada SimEd (Simulador de Estructuras), permite hacer uso de las estructuras estudiadas, así como experimentar como van evolucionando éstas según se incorpora o se elimina información. Además, permite visualizar su funcionamiento externo (el concepto de la estructura prescindiendo de su implementación) así como el funcionamiento interno (según sea la implementación).

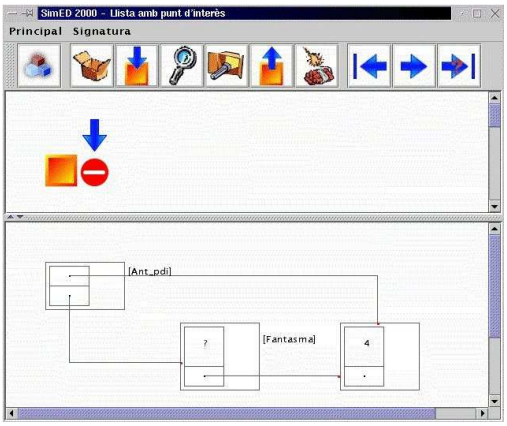

Figura 2. Ejemplo de estructura de datos en el simulador.

- Con la finalidad de favorecer su uso, SimEd es accesible desde Internet. Desde un principio, se ha concebido de forma integrada en la plataforma WEB de la asignatura, como una herramienta más que permite reforzar los conocimientos de los alumnos.
- Como consecuencia inmediata al criterio anterior, para permitir su ejecución en cualquier sistema, se ha desarrollado en lenguaje Java.
- Otro objetivo importante es la posibilidad de poder hacer un seguimiento del uso que hacen de ella los alumnos. A medida que trabajan con SimEd se va registrando su uso en una base de datos que, a posteriori, nos permitirá sacar alguna conclusión estadística sobre la utilidad de la herramienta según el grado de uso que se haga de ella.
- La primera versión de SimEd se ha desarrollado en un entorno genérico que permite añadir nuevas estructuras de forma realmente fácil. En este entorno aparece un listado de las estructuras disponibles para su uso. Una vez seleccionada una de ellas, permite trabajar con ésta desde el punto de vista de cualquier programador-usuario del TAD en cuestión, o sea, a partir de su interfaz (las operaciones que gestionan correctamente la estructura).
- Actualmente contiene las siguientes estructuras: la pila, la cola y algunas listas (la lista con pdi, la lista ordenada y la lista

ordenada bidireccional). Cada estructura de datos está representada de varias formas. Como línea futura queda por ampliar SimEd con otras estructuras de datos.

- El propósito inicial del diseño de la aplicación fue trabajar con las estructuras básicas impartidas en la asignatura de programación. No obstante, puede ser ampliada con otras estructuras mucho más complejas tales como multilistas, grafos o árboles. Esto ayudaría a la comprensión de las mismas a alumnos de asignaturas tales como estructuras de datos o programación II.

6. Conclusión

En este artículo hemos presentado cómo enfrentarnos desde diferentes puntos de vista al problema de la falta de motivación de los alumnos. La motivación es baja debido a la novedad de la asignatura, la dificultad de los alumnos en aprender conceptos abstractos y la falta de capacidad de trabajo continuo.

A tal efecto, hemos propuesto tres modalidades para evitar todos estos problemas. En primer lugar, planteamos un entorno vía WEB de comunicación entre alumnos y profesores. En segundo lugar, proponemos un plan de trabajo continuo que, a la vez, nos permite conocer la evolución de los alumnos. Y finalmente, empleamos una aplicación de entorno educativo para introducir a los alumnos el concepto de TAD, a la vez que mostramos el funcionamiento de las estructuras de datos lineales más básicas.

Los tres últimos años hemos empleado esta metodología de trabajo con los alumnos. Los resultados estadísticos obtenidos en los diferentes cursos, avalan que esta metodología ayuda a conseguir un número mayor de alumnos presentados al final de curso. También permite que los alumnos consigan una mejor asimilación de los diversos contenidos.

Nuestras líneas de trabajo futuras están dirigidas a: (1) ampliar el simulador SimEd con nuevas estructuras de datos e incluir este simulador en las clases prácticas, y (2) incorporar la evaluación continua desde la propia página WEB con la finalidad de disponer de más datos sobre el trabajo de nuestros alumnos.

Referencias

[1] Bernadó, Ester y Garrell, Josep Maria y Román, Manuel y Salamó, Maria y Camps, Joan y Abella, Jaume. "*Introducción a la programación en el ámbito de diversas ingenierías*", Jenui, 1998.

[2] Budd, Timothy. "*Introducción a la Programación Orientada a Objetos*". Addison-Wesley Iberoamericana, 1994.

[3] Camps, Joan y Salamó, Maria y Vallespí, Carles y Vernet, David. "*Exercicis programació Curs 2000*", Departament d'Informàtica, Enginyeria i Arquitectura La Salle, 2000.

[4] Camps, Joan y Román, Manuel y Abella, Jaume. "*Exercicis d'estructures de dades i TAD's*". Departament d'Informàtica, Enginyeria i Arquitectura la Salle, 1998.

[5] Castro, Jorge y Cucker, Felipe y Messeguer, Xavier y Rubio, Albert y Solano, Lluis y Valles, Borja. "*Curs de programació*", McGraw-Hill ,1992.

[6] Escudero Costa, Francesc y Garrell i Guiu, Josep Maria. "*Fonaments de programació*". Bruño/EUETT, 1993.

[7] González, Xavier. "*Entorn Educatiu de Teleensenyament de Programació I, Estructures de Dades Lineals*". Treball Final de Carrera, Departament d'Informàtica, Enginyeria i Arquitectura La Salle, Universitat Ramon Llull, 2000.

[8] Kernighan, Brian W. and Ritchie, Dennis M. "*The C programming language*", Prentice Hall, 1988.

[9] Perry, Jo Ellen and Levin, Harold, D. "*An Introduction to Object Oriented in Design in C++*". Addison-Wesley, 1996.

[10] Stroustrup, Bjarne. "*The C++ Programming Language*". Addison-Wesley, 1995.