Una apuesta por la motivación al alumnado en las asignaturas de programación: el sistema de evaluación continuada

Marisa Durán
Dpto. de Informática
Universidad de Extremadura
10071 Cáceres
e-mail: mlduran@unex.es

Andrés Caro
Dpto. de Informática
Universidad de Extremadura
10071 Cáceres
e-mail: andresc@unex.es

Pablo G. Rodríguez
Dpto. de Informática
Universidad de Extremadura
10071 Cáceres
e-mail: pablogr@unex.es

Resumen

En el presente trabajo se exponen las dificultades ante las que se encuentran la mayoría profesores que imparten asignaturas de programación en la Universidad española. La excesiva masificación en las aulas, la convivencia de diferentes planes de estudios, la limitación de aulas y de espacio en las mismas, etc... provocan una desmotivación en el alumnado que conduce al fracaso. En este artículo se propone como una posible solución la realización de un sistema de evaluación continuado que implique mayor motivación en el alumnado. Además, ante un foro de discusión como el presente, se comparten las experiencias que, en los últimos cuatro años, han ido adquiriendo los autores del trabajo.

1. Introducción

Las asignaturas programación suponen un gran número de créditos en las titulaciones de informática que se imparten en la actualidad en la Universidad española. Los créditos prácticos conllevan la resolución de un supuesto de una cierta envergadura. Uno de los objetivos de este tipo de asignaturas es conseguir que el alumno aprenda a afrontar problemas importantes y pueda llevar a cabo su resolución utilizando las estructuras de datos y las técnicas algorítmicas adecuadas. Sin embargo, la masificación en las aulas en este tipo de titulaciones es un problema muy común, que viene aumentado con el hecho de que, además, existe la posibilidad de que convivan diferentes planes de estudios simultáneamente. Además, la limitación de equipos, aulas, y espacios en las mismas, inevitablemente, lleva aparejada una cierta desmotivación del alumnado, lo que provoca un alto índice de fracaso en estas asignaturas.

En la sección 2 de este artículo se expone la problemática de las asignaturas en los últimos años en nuestra Universidad. En la sección 3 se comenta el material de apoyo elaborado por los profesores de la asignatura. Ya en la sección 4 se expone la propuesta de evaluación continua puesta en marcha hace tres años. En la sección 5 se discute acerca de los resultados obtenidos, exponiendo la opinión de los alumnos en la sección 6. Para terminar, en la sección 7 se exponen las principales conclusiones obtenidas.

2. Las asignaturas de programación en nuestra Universidad

En la Universidad de Extremadura, para las tres titulaciones de informática (I.I., I.T.I.S. e I.T.I.G.), se imparten dos asignaturas de programación troncales en primer curso de carrera: “Elementos de Programación” (anual, de 9 créditos) y “Laboratorio de Programación I” (2º cuatrimestre, 6 créditos). En segundo de carrera se imparte una asignatura troncal, “Estructuras de Datos y Algoritmos” (anual, 9 créditos) y otra obligatoria, “Laboratorio de Programación II” (2º cuatrimestre, 6 créditos).

Las dos asignaturas de *laboratorio* suponen la realización de prácticas de una cierta magnitud (según el curso) que complementen los conceptos teóricos expuestos en las correspondientes asignaturas anuales. El éxito en este tipo de asignatura influiría, irremediablemente, en las

asignaturas anuales. Sin embargo, son asignaturas que han venido sufriendo un alto índice de abandono en los últimos años, debido, en gran medida, a que el alumno pospone la realización de la práctica hasta el final del curso, encontrándose en ese momento con falta de tiempo y de fluidez. Por tanto, la desazón y la falta de motivación del alumnado están a la orden del día.

En un intento por promover la realización de los supuestos prácticos en los tiempos establecidos, hace tres cursos los autores de este trabajo decidieron ofrecer al alumno la posibilidad de someterse a una evaluación continuada en la asignatura de "Laboratorio de Programación II", asignatura en la que imparten docencia fundamentalmente y en la que se centrará el estudio de ahora en adelante. Para aprobar esta asignatura se exige que los alumnos desarrollen *individualmente* una práctica en C++, utilizando técnicas de P.O.O., con las especificaciones expuestas en un enunciado que se les entrega al principio del curso.

La evaluación de esta asignatura se basa en los siguientes criterios:

- Realización de una práctica desarrollada de forma individual, en lenguaje C++. La práctica debe entregarse debidamente resuelta y documentada.
- Superación, si se estima necesario, de un examen práctico que consistirá en realizar una modificación de la práctica.
- Entrega de una documentación que consistirá en: Manual del Usuario y Manual del Programador (que incluirá el código fuente), así como de discos con los ficheros fuente de la práctica y modificación.

Dicha materia consta de 6 créditos, 4.5 prácticos y 1.5 teóricos, y se imparte durante el segundo cuatrimestre del segundo curso de la carrera.

De lo expuesto anteriormente se concluye que el método de evaluar a los alumnos es el de que cada alumno, de forma individual, realice la práctica durante el cuatrimestre, y sea convocado a un examen en el que realicen, en un periodo de dos horas, una modificación a la práctica. Este tipo de exámenes suele consistir en añadir nuevos métodos en algunas de las clases, en añadir una nueva clase derivada a alguna de las jerarquías de herencia que conforman la práctica o en desarrollar algoritmos que solucionen los problemas propuestos.

El objetivo perseguido con este tipo de examen es la discriminación entre alumnos que han desarrollado la práctica por sí mismos con respecto a aquellos que acuden a otros medios para obtener la resolución final de la misma. El alumno es evaluado en función de la capacidad demostrada para modificar su ejercicio. Así, se premia a las prácticas bien estructuradas y organizadas en cuanto a modularización. El alumno entrega al profesor el código fuente desarrollado, el de su modificación y una documentación que incluye un manual del usuario y un manual del programador.

Durante los últimos años la práctica planteada a los alumnos consiste, a grandes rasgos, en utilizar estructuras de almacenamiento lineales, tales como colas y listas, sobre las que aplican polimorfimo y genericidad. Siempre han de llevar a cabo el diseño de una jerarquía de clases con métodos propios y métodos heredados, a fin de aplicar el polimorfismo, y entender el mecanismo de la herencia. Además trabajan con una estructura de almacenamiento jerárquico, que suele ser un árbol, manejan a su vez entrada y salida sobre ficheros de texto e implementan estructuras de tipo grafo.

Se ha de considerar el hecho de que desde el curso 1998/99 se introduce un nuevo plan de estudios, y pierden su derecho a docencia aquellos alumnos que no se acogen a este nuevo plan. A esta situación se ha de añadir que en los últimos años se venía arrastrando un alto nivel de fracaso, lo que originaba desmotivación previa en los alumnos y, por otro lado, una sensación generalizada de "cierto temor" y abandono de la asignatura desde los primeros días del curso.

3. Material de apoyo a la docencia

Los autores de este trabajo han confeccionado un guión del curso teniendo en cuenta las semanas lectivas de que consta el cuatrimestre, elaborando una colección de supuestos prácticos totalmente resueltos, que sirven, en algunos casos, como ejercicios que se desarrollan y se discuten en clase, y en otros casos como material complementario que se facilita a los alumnos, para motivar su autoformación.

Los ejercicios propuestos, aunque diferentes en su planteamiento, mantienen una relación directa con la resolución del supuesto práctico que deben realizar los alumnos, de modo que se proponen para que sean los propios estudiantes los que, durante las sesiones prácticas desarrolladas en las salas de ordenadores, van confeccionando su propia biblioteca de ejercicios propuestos y resueltos por ellos mismos. Estos ejercicios suponen ejemplos claros y evidentes de los conceptos fundamentales de la asignatura, tales como encapsulación, herencia, polimorfismo, sobrecarga, genericidad y uso de estructuras de datos.

La bibliografía básica utilizada para el desarrollo de esta asignatura se muestra al final del trabajo.

4. Evaluación continua

En el curso 2000-01 el número de alumnos ascendía aproximadamente a 550 (400 del plan nuevo y el resto del plan antiguo). Ante esta situación parecía difícil introducir innovaciones docentes en la asignatura. Sin embargo, tras analizar la situación de los años anteriores, los autores consideraron necesario establecer mecanismos que contribuyesen a incrementar el grado de motivación de los alumnos y su implicación en la asignatura durante el curso. La experiencia demostraba que el alumno se enfrentaba a la materia con total "desconfianza" sobre sus capacidades y, en muchas ocasiones, no resolvía la práctica por sí mismo, sino que buscaba ayuda en otros medios, tales como academias, alumnos de años anteriores u otras soluciones. A fin de evitar esta situación, se apostó por llevar a cabo un sistema de *evaluación continua*. Dicho sistema consistía en exigir a los alumnos que, a lo largo del curso, fuesen entregando módulos de la práctica. No era suficiente la entrega de dichos módulos, sino que, además, se iría haciendo un pequeño examen por cada una de las entregas. De esta forma, se fijaba una fecha para la conclusión de los módulos y además, el alumno recibía información acerca de los errores que iba cometiendo, permitiéndose así la posibilidad de subsanarlos, en lugar de arrastrarlos hasta el fin de la práctica como podía ocurrir con el método de evaluación anterior. La idea de que "a programar se aprende programando" no sólo se transmite al alumno, sino que, además, se fomenta y potencia mediante el trabajo diario que supone la realización por partes de la práctica.

El alumno es informado de las fechas en las que se van a llevar a cabo estos controles en el mismo momento en que se les entrega el enunciado de la práctica. Después de tres cursos realizando este tipo de evaluación, el esquema se fundamenta en cinco entregas: las tres primeras suponen la resolución de forma independiente de determinados aspectos de la práctica, tal y como se comenta a continuación, siendo la cuarta la que implica la integración de estas tres entregas en un único proyecto.

- 1ª Entrega: finales de marzo. Se exigen las estructuras de datos lineales y la implementación de algunas clases en herencia, así como el uso de polimorfismo.
- 2ª Entrega: finales de abril. Se exige el Árbol Binario Ordenado y especificación de las clases que se almacenan en el mismo.
- 3ª Entrega: mediados de mayo. Se pretende que se implemente la estructura de tipo grafo y las clases relacionadas con el mismo, así como el uso de programación genérica.
- 4ª Entrega, finales de mayo. Integración final de los módulos. Entrega de código fuente.
- 5ª Entrega, en junio. Documentación (manual del usuario y del programador).

Hay que tener en cuenta que el alumno no pierde nunca la posibilidad de ser evaluado del mismo modo en que se venía haciendo en anteriores cursos. No obstante, con esta propuesta, los alumnos que han ido superando un número mínimo de entregas, no tendrán la necesidad de presentarse al examen final que seguirá consistiendo en realizar una modificación de la práctica tal y como se ha expuesto en la sección 2.

Así mismo, la evaluación continua y los controles periódicos que ésta supone ha conllevado un espectacular aumento en las consultas realizadas por los alumnos en los horarios de tutorías. Si bien antes los alumnos apenas asistían a consultar sus dudas, y lo hacían a escasos días de la finalización del curso, en la actualidad los alumnos utilizan este recurso docente con bastante frecuencia, lo cual propicia una mayor interacción alumno-profesor, mejores relaciones entre ambos y más confianza mutua.

5. Discusión

Según lo expuesto, se ve bastante clara la necesidad de mejorar la *actitud participativa* de los alumnos en la asignatura. El cambio de planes de estudio, la nota mínima de acceso tan asequible para estas titulaciones y las condiciones socioeconómicas de la región, son factores que influyen en el alto número de matriculados con bajo interés en la carrera. La mejora expuesta en este trabajo hizo pensar a los autores del trabajo en la posibilidad de que los alumnos aún se desentendiesen más ante el esfuerzo que supone la evaluación continua. Sin embargo, con la perspectiva que suponen los tres cursos académicos que se lleva realizando este sistema de evaluación, los resultados no pueden haber sido mejores. La gran mayoría de alumnos matriculados se acoge al sistema de evaluación continua, intentando, al menos, seguir el ritmo de realización de la práctica. En la figura 1 se observa cómo el porcentaje de aprobados en los últimos años en las asignaturas de programación de segundo curso han evolucionado de una forma bastante similar. Desde el curso 1999/00 se ha incrementado el porcentaje de aprobados en las asignaturas, de forma constante. Sin duda, la evaluación continua y la posibilidad para los alumnos de aprobar la asignatura sin presentarse al examen final, hecho que, por experiencia, puede decirse que les motiva bastante, propicia un estudio y trabajo diario que implica que la asignatura sea superada casi sin esfuerzo. Y este estudio se ve reflejado en las dos asignaturas de programación del curso.

6. Opinión de los alumnos

Es muy destacable la buena acogida que el sistema de evaluación continua ha tenido en estos años por parte de los alumnos. Los autores han realizado encuestas anónimas a los alumnos al final de cada curso, siendo unánime la buena acogida que el sistema de evaluación continua ha tenido entre lo mismos.

A continuación se exponen algunas de las opiniones más repetidas expresadas por los alumnos:

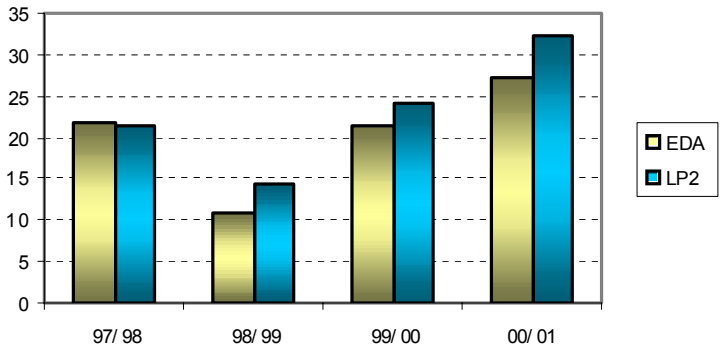

Figura 1. Porcentaje de aprobados en los últimos años en las asignaturas de programación de segundo curso

- El método de aprendizaje es muy bueno. Si te tomas en serio la evaluación continua, aprendes. Es duro, pero es necesario y eficiente.
- Lo mejor de la asignatura es la evaluación continua: el seguimiento a los alumnos es bueno.
- La "obligatoriedad" de ir entregando por partes la práctica para la evaluación continua hace que nos planteemos la asignatura día a día y no la dejemos para el final, que es lo que suele ocurrir casi siempre con las prácticas de este tipo.
- El método nos obliga a trabajar a diario y nos hacemos una idea de cómo se implementa un proyecto poco a poco, ya que de otra manera no sabemos ni como empezar.
- El método de evaluación es más exigente, pero creo que así se aprende más y se te evalúa de forma más justa.
- El método de evaluación es muy bueno. Se obliga al alumno y al profesor a un avance progresivo. El alumno aprende poco a poco y el profesor corrige poco a poco.
- La asignatura es más fácil de llevar, porque hacerlo todo al final y jugarse la asignatura en un examen es bastante más difícil.
- Con el trabajo diario he tenido mayor motivación y he prestado más atención a la asignatura.
- En mi caso, el tener una fecha tope para entregar algo es una buena manera de obligarme a trabajar.

7. Conclusión

La evaluación continuada de las prácticas de programación han resultado muy satisfactorias en la enseñanza de las asignaturas de programación, ya que han permitido la motivación de los alumnos, el estudio cotidiano de los conceptos teóricos y el desarrollo diario de ejercicios, programas y de la propia práctica de la asignatura. El estímulo que supone para el alumno la posibilidad de "aprobar sin presentarse al examen", así como el temor al examen final, propician la elevada motivación que los autores han observado en el alumnado. La elevada aceptación de este sistema por parte del alumnado, así como los buenos resultados obtenidos en los últimos años hacen que los autores del trabajo recomienden este sistema de evaluación para las asignaturas de programación.

Bibliografía

[1] Bell, D., *The essence of programming using C++* . Prentice-Hall, 1997.
[2] Charte, F. *Programación Orientada a Objetos con Borland C++*. Anaya, 1993.
[3] Deitel, H.M. y Deitel, P.J.: *Cómo programar en* C/C++. 2ª edición, Prentice-Hall, 1998.
[4] Mansfield, K.C. *An introduction to programming in C++*. Prentice-Hall International, 1997.
[5] Joyanes, L. *Programación en C++: algoritmos, estructuras de datos y objetos*. McGraw-Hill, 2000.
[6] Heileman, G. *Estructuras de Datos, Algoritmos y Programación Orientada a Objetos*. McGraw-Hill, 2000.
[7] Schildt, H. *C/C++: Manual de referencia*. McGraw-Hill, 1999
[8] Schildt, H. *C: Guía para usuarios expertos*. Osborne / Mc Graw-Hill, 1991.
[9] Stroupstrup, B. *El lenguaje de programación C++*. Addison-Wesley, 2002.