

4.8. SEL: UN SISTEMA DE APRENDIZAJE NO PRESENCIAL PARA LA PROGRAMACIÓN MEDIANTE EJERCICIOS EN INTERNET

M. A. Varó Giner; S. Meliá Beigbeder; S. Luján-Mora; F. B. Navarro Colorado; I. Garrigós Fernández; F. Llopis Pascual; J. Peral Cortés; A. Ferrández Rodríguez; M. A. Baeza Ripoll; E. Saquete Boro; J. Aragonés Ferrero

*Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante*

RESUMEN

Siguiendo un método continuista basado en la docencia tradicional, la enseñanza de los conceptos teóricos en las asignaturas de programación se fundamenta en la impartición de clases presenciales y teóricas. En estas clases, el alumno asiste la mayor parte del tiempo de forma pasiva y el tiempo es reducido para realizar tareas u ejercicios. Sin embargo, este método ha resultado ineficaz en muchas ocasiones, ya que la programación exige un mayor aprendizaje activo y constructor para poder ser entendido apropiadamente. Para solventar esta carencia, hemos diseñado una herramienta llamada SEL (Sistema de Enseñanza en Línea) que permite a los alumnos puedan realizar ejercicios de forma no presencial y complementar el aprendizaje recibido en las clases. SEL intenta cubrir todos los tipos de ejercicios de programación y ayuda al profesor a realizar un seguimiento del alumno que permita conocer cual es su evolución e incluso evaluarle. Actualmente, SEL está siendo aplicada con éxito en la asignatura Fundamentos de la Programación 2, siendo nuestra intención aplicarla en cualquiera de las asignaturas de programación.

1. INTRODUCCIÓN

El aprendizaje de la programación es un punto fundamental dentro de la formación de un alumno en cualquiera de las titulaciones de Informática. La programación es una materia que requiere una metodología de aprendizaje activa y constructorista por parte del alumno. Por ello, todos los esfuerzos desarrollados a la hora de mejorar tanto el seguimiento como la autoevaluación de las tareas de programación es de gran beneficio para la docencia dentro de estas titulaciones.

Tradicionalmente, las asignaturas de programación han presentado una estructura basada en dos partes principales, (1) una parte teórica donde se imparten los conocimientos de programación, y (2) una parte práctica donde los alumnos deben desarrollar dichos conocimientos y aplicarlos a diferentes casos. Sin embargo, en muchas ocasiones dichas partes no están lo suficientemente integradas, y no todos los conceptos explicados en la parte teórica van a poder ser tratados en la parte práctica y viceversa. Es por ello que normalmente se le propone al alumno la realización de una serie de ejercicios, que van de la mano de cada tema teórico, y que le permiten aplicar el conocimiento adquirido, ya sean en casa o en la propia clase, sobre aquello que el profesor explicó en clase teórica.

Sin embargo, hemos detectado a lo largo de estos años de enseñanza activa, que en la mayoría de los casos, ya sea por la falta de motivación o por una no adecuada planificación por parte del alumno, los ejercicios planteados en las diferentes clases no son realizados y por consiguiente, hay un seguimiento insuficiente de la asignatura por parte del alumno, lo que arrastrará los posteriores problemas de comprensión y posible fracaso a la hora de aplicar dichos conocimientos.

Para solucionar este problema, proponemos un método basado en la realización de los diferentes ejercicios de programación desde Internet. Consiste en una sencilla aplicación Web, que permite la realización de ciertos ejercicios, ya sean mediante tipo test, o mediante pequeños programas, que van a permitir al alumno poder realizar dichos ejercicios desde casa, o desde cualquier laboratorio con acceso a Internet. Este método va a potenciar principalmente la motivación por parte del alumno para realizarlos, y además le va a permitir al profesor un mejor seguimiento sobre su evolución, e incluso realizar cierto mecanismo de evaluación si fuera conveniente.

En nuestra red docente contamos con diferentes asignaturas, cada una de las cuales presenta una problemática diferente a la hora de poder implantar ejercicios adecuados en dicha aplicación Web. Inicialmente, nuestra investigación se ha centrado en aquellas asignaturas que resultaba más sencillas de abordar por el tipo de ejercicios, más concretamente en *Fundamentos de la Programación 2* (FP2), en el que hemos podido implantar con éxito los ejercicios planteados tradicionalmente a los alumnos.

La memoria está organizada de la siguiente manera: en la sección 2 presentaremos un estudio de la programación sobre la plataforma Web. A continuación, en la sección 3, se introducirá la aplicación *Servicio de Ejercicios en Línea* (SEL), la cual se ha aplicado a la asignatura de Fundamentos de la Programación 2.; presentaremos cada una de las vistas, como profesor y como alumno, y cuáles son los diferentes tipos de ejercicios sobre los que hemos desarrollado dicha aplicación. En la sección 4 se presentarán los trabajos relacionados en el campo de la enseñanza por Internet. Y por último, en la sección 5 presentaremos las conclusiones y el trabajo futuro.

2. ESTUDIOS DE LA PROGRAMACIÓN EN WEB

Antes de proceder a realizar el programa informático SEL, realizamos un estudio que pretendía tipificar cuáles son los posibles ejercicios de programación que nos permitiría implantar una plataforma tan limitada como la Web. Basándonos en trabajos previos, y en la propia experiencia con la Web, conseguimos tipificar los diferentes ejercicios que potencialmente podríamos implantar en la Web. Finalmente, concluimos indicando los que hemos considerado interesantes para implantar.

- a) **Ejercicio de Test.** Es el tipo de ejercicio más común y sencillo de realizar, de hecho ya ha sido implantado en muchos sistemas de aprendizaje por Internet como [1] [3]. Se plantea al usuario elegir entre una serie de posibles respuestas que se le presentan delante del código ya introducido. Entonces el usuario elige una de las posibles respuestas entre las 3, 4 o más planteadas. Esto lo hace mediante un mecanismo de introducción de datos HTML llamado radio button. Una vez introducido se le indica al usuario si ha acertado o ha fallado. Además se le puede indicar junto a la solución correcta una explicación adecuada. Veamos a continuación un ejemplo de pregunta planteada mediante test:

Indica qué valdrían las variables del programa a, b y c:

```
int a,b,c;  
a=10;  
b=1;  
c=1;  
while ( a = b ) {  
  b--;  
  c++;  
}
```

a) Se produciría un bucle infinito

b) a y b valen 0, c vale 2 -> "No sale del while hasta que a valga la condición 0. Para ello es necesario ejecutarse una sola iteración, cuando b valga 0."

c) a vale 10, b vale 1 y c vale 1

d) Ninguna de las anteriores

En el ejemplo podemos apreciar que se presenta una pregunta en la que el código ya está introducido, y donde tenemos que indicar cuáles son los valores de las variables. Podría aplicarse a preguntas sobre el valor de salida de una función, el mensaje mostrado por pantalla, etc. La ventaja principal de este tipo de preguntas es su sencillez a la hora de implementarse, y la desventaja es que cubre un reducido conjunto de preguntas.

- b) **Introducción de parte del código.** Este tipo de ejercicio propone al usuario una pregunta en la que se le pide que complete un programa introduciendo una parte de código necesario para que cumpla una condición, devuelva unos valores, etc. La inserción de dicho código se realiza mediante un campo de texto. La tarea por parte del programa consiste en comprobar si la cadena introducida es la correcta.

Indica qué condición tendría que introducirse para que al terminar de ejecutarse el código las variables tuvieran los valores: $a=b=0$ y $c=2$:

```
int a,b,c;
a=10;
b=1;
c=1;
while (  ) {
    b--;
    c++;
}
```

Con este tipo de pregunta disponemos de más posibilidades a la hora de proponer posibles preguntas de programación, ya que se le puede solicitar la completitud de código en cualquier parte del programa y situación. En el ejemplo se plantea rellenar la condición de control, otra alternativa sería una inicialización, o cualquier línea de instrucción.

La ventaja principal de este tipo de ejercicio es que aborda un tipo de pregunta que requiere un mayor esfuerzo al alumno, ya que es el alumno quien tiene que programar parte del programa. Por otro lado, la principal desventaja se presenta a la hora de implementar dicha propuesta, ya que puede existir más de una posible porción de código como respuesta a cada pregunta, y se han de validar todas.

- c) **Traza de programación.** Un posible ejercicio que surge en multitud de ocasiones dentro de la programación, es la realización de las trazas de ejecución. Consiste en la introducción de los valores que tendrían las variables solicitadas dentro de una posible ejecución. Para ello solamente es

necesario la incorporación de una serie de campos de entrada (ya sea mediante un campo de texto o de selección) en la parte inferior del código mostrado.

Indica qué valores tendrían cada una de las variables en la condición del bucle while:

```
int a,b,c;  
a=10;  
b=1;  
c=1;  
while (a=b){  
  b--;  
  c++;  
}
```

1ª Iteración: a= b= c=

2ª Iteración: a= b= c=

La ventaja principal es que deja al profesor la posibilidad de indicar el número de variables e iteraciones que fueran necesarias. Sin embargo, su principal desventaja es la especialización, ya que exige un esfuerzo de implementación grande, para un número reducido de posibles preguntas.

d) Ejercicio de programación de una función o parte de un programa.

Este tipo de pregunta consiste en la introducción del código de un programa, ya sea de una función o de parte de un programa. La forma de introducirlo es mediante un campo de entrada HTML que se denomina área de texto (textarea). La forma de plantear una pregunta de este tipo se realiza mediante la modularización del código solicitado, es decir, solicitamos al usuario que implemente un módulo que tiene que realizar una determinada tarea, recibiendo como entrada un conjunto de parámetros especificados, y devolviendo los datos que le indicamos. Dicho código sería enviado al servidor que, inicialmente, lo compilaría a continuación lo ejecutaría, mostrando el resultado por pantalla.

Como se puede apreciar en el ejemplo, el usuario sigue las indicaciones del enunciado e implementa la función solicitada. La ventaja principal de este tipo de ejercicio, es que permite poner una amplia variedad de ejercicios donde le podemos solicitar que implemente multitud de funciones que cubran muchos aspectos de programación. La desventaja es que hemos de mejorar el interfaz para que permita el coloreado de palabras clave, y el trabajo que exige al profesor que implementa la pregunta, ya que ha de realizar el validador de esta pregunta para comprobar si ha sido correctamente implementada.

Realizar una función que se llame "QuitarBlancos" que abra un fichero de texto, y proceda a eliminar sus espacios en blanco. Recibirá como entrada el nombre del fichero. Devolverá 0 si todo ha ido bien, y 1 si hubo un error al abrir el fichero.

```
int Quitar Blancos (string nomFich)
{
    const char BLANCO = ' ';
    fstream f1;
    char c;

    f1.open(nomFich.c_str(),ios::in);
    if (!f1) cout <<"Error de apertura del fichero\n";
    else {
        //Lee caracter a caracter y comprueba si
        f1.get(c);
        while (!f1.eof()) {
            if (c==BLANCO) cout<<c; //Si no es un l
            f1.get(c);
        }
    }
}
```

Ejecución correcta. Salida 0.

Una vez realizado el análisis sobre los diferentes tipos de problemas que podemos plantear en Web, consideramos que son 2 los tipos que hemos visto más interesantes a la hora de implantar en nuestro sistema. El tipo ejercicio de test, porque nos permite fácilmente introducir una gran cantidad de preguntas de programación relacionada con el conocimiento teórico del alumno, y el tipo ejercicio de programación que permite al estudiante realizar la tarea más constructora de programar un código solicitado. De esta manera creemos que cubrimos los aspectos teórico y práctico que consideramos imprescindibles para el aprendizaje de dichas asignaturas.

3. PROGRAMA DE SERVICIO DE EJERCICIOS EN LÍNEA (SEL)

El programa SEL (*Sistema de Ejercicios en Línea*) consiste en una aplicación Web que pone a disposición del alumno del conjunto de ejercicios propuestos a lo largo del curso, para que los realice desde cualquier ordenador que disponga de acceso a Internet, y en cualquier momento fuera del horario de clases. Este programa permite al profesor de la asignatura de programación, implantar aquellos ejercicios que no puedan realizarse en clase de forma presencial, de manera que solicite la realización a sus alumnos, y pueda controlar quiénes los han realizado y cuál es el éxito que han tenido con cada una de las preguntas solicitadas.

Además, el programa se puede considerar una herramienta de autoaprendizaje para la programación, ya que va informando al alumno de la solución correcta según va realizando los diferentes ejercicios, tanto si acierta como cuando falla, lo que le permite un aprendizaje de forma individual y no presencial.

Por otro lado, SEL permite la evaluación de los alumnos en cada tema de la asignatura. De manera que presenta 10 ejercicios de cada tema de forma aleatoria. El sistema mostrará si un alumno ha superado los ejercicios de un determinado tema, o por el contrario dicho tema todavía no lo tiene cubierto. Esto exige un aprendizaje progresivo al alumno, y evita la aparición de posibles lagunas en determinados temas que afectaría en temas posteriores.

Por último, desde el punto de vista del profesor, dicha aplicación da la posibilidad de que sea él quien introduzca los ejercicios, ya sean de tipo test o de programación. Y además, también es el profesor quien se encargaría de realizar el módulo de validación de la pregunta de programación. De esta manera distribuye perfectamente los roles de responsabilidad, sin necesidad de que hayan terceras personas encargadas de realizar ninguna tarea de implantación.

A continuación mostraremos una descripción detallada de SEL. Inicialmente veremos el acceso que se considera común para ambas vistas, para posteriormente ver las opciones que ofrece el programa en la vista del alumno y desde el punto de vista del profesor.

3.1 ACCESO A LA APLICACIÓN

La aplicación proporciona un acceso mediante una página común, en la que se le solicita al usuario la introducción de su identificación o login, y de su clave o password. Así que una vez introducida la identificación del usuario, según el rol que tenga asignado, podrá acceder a la aplicación en la vista de profesor, en la vista de alumno o en ambas.

En la figura 1 podemos apreciar la página principal de la aplicación que se solicita la información de validación de usuario. De esta manera protegemos el programa de posibles accesos no adecuados, por ejemplo, alumnos que accedan a la vista de profesor, o personas que no estén matriculadas accedan al sistema SEL.

3.2. VISTA DEL ALUMNO

Principalmente esta vista ofrece al alumno la posibilidad de realizar los ejercicios en los diferentes temas, y tener una referencia de cuál es su evolución en cada uno de ellos. SEL le muestra el porcentaje de aciertos y de errores que ha tenido en cada tema. En el caso de que dicho porcentaje de aciertos supere un umbral, que estará determinado por el profesor (p.e. un 80%), el alumno tendrá dicho tema superado. En el caso de que no haya superado dicho umbral, tendrá la oportunidad de poder realizar de nuevo los ejercicios de dicho tema. Eso sí,

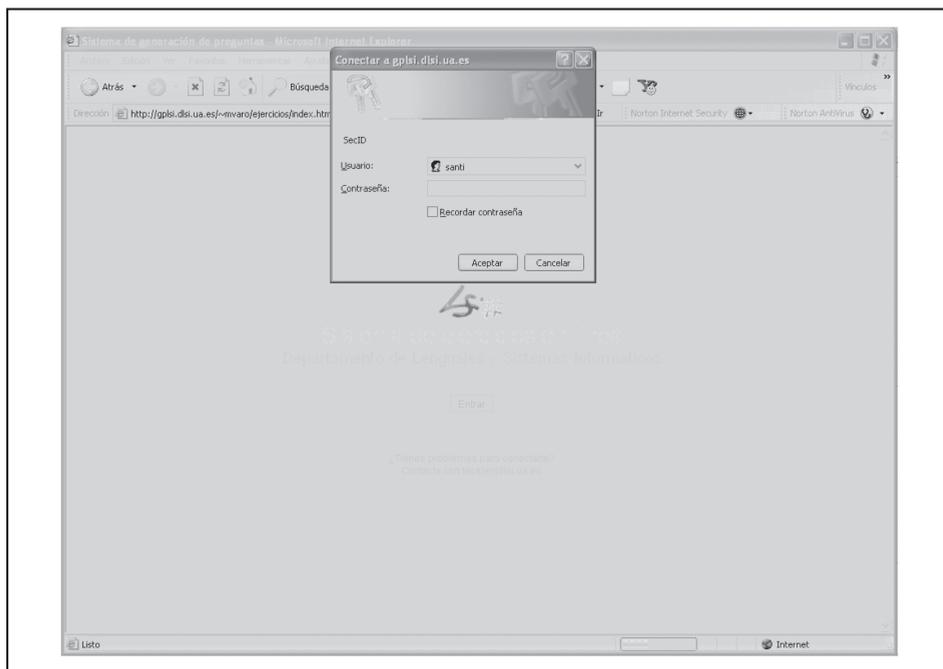


Figura 1. Pantalla de acceso a SEL.

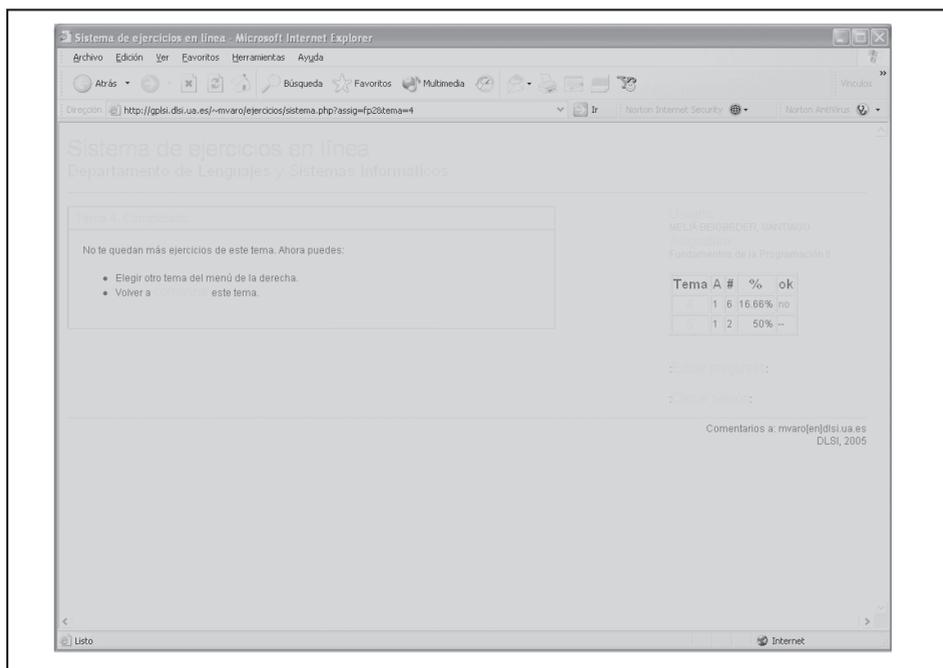


Figura 2. Pantalla de acceso en la vista de alumno.

como hemos mencionado previamente, por cada tema se muestran 10 ejercicios aleatoriamente de un mayor conjunto de ejercicios que haya introducido el profesor, de esta manera intentamos evitar que mediante la repetición lo puedan superar fácilmente.

Por otro lado, el alumno tiene la oportunidad de saltar de un tema a otro, ya que se guarda en todo momento la información sobre el ejercicio en el que se ha quedado el alumno. Así, se permite que en el caso de que dicho alumno se encuentre bloqueado en un determinado ejercicio, pueda continuar con un ejercicio de otro tema. De la misma manera, el estado se guarda después de terminada una sesión del alumno, para que continúe en el mismo sitio donde lo dejó.

Una vez el usuario ha sido identificado como alumno dentro del sistema, se nos muestra una pantalla en la que se nos indica la asignatura de programación que estamos cursando, y los temas en los que hemos progresado. Como se puede apreciar en la parte derecha de la figura 2, aparece el nombre y los apellidos del alumno y debajo la asignatura que está cursando, en este caso Fundamentos de la programación II. Por otro lado, se le muestra al alumno la evolución que ha tenido en los diferentes temas en una tabla donde se indica el número del tema, el número de aciertos (debajo de la letra A), y el número de fallos (indicado con la #), además se indica si el alumno ha superado o no un determinado tema (debajo de la palabra ok).

A continuación indicamos los dos tipos de ejercicios que podemos encontrar en la aplicación SEL.

3.2.1. Ejercicio de Test

Como ya hemos comentado en la sección 3, en el ejercicio de test se plantea al usuario elegir entre una serie de posibles respuestas que son presentadas delante de una cuestión, ya sea sobre una parte de código de un programa, o una cuestión referente al conocimiento teórico de la programación. En el ejemplo presentado de la figura 3, podemos apreciar que el usuario elige entre cuatro posibles respuestas. SEL admite establecer desde 2 hasta 6 posibles alternativas a una pregunta de tipo test. La selección se realiza mediante el mecanismo del radio button. Una vez elegido el usuario, pulsa la tecla responder y pasa a otra página que le indica si ha acertado o por el contrario ha fallado en la respuesta.

Tanto si la respuesta introducida ha sido correcta, como si ha sido incorrecta, al usuario se le muestra la solución y una explicación indicando cuál es el motivo. Esto se debe a que en muchas ocasiones se necesita de una explicación clara para que el alumno entienda perfectamente la respuesta.

En la figura 4 podemos apreciar un caso de respuesta incorrecta, y cómo al alumno se le explica cuál es la solución y por qué. Esto, como veremos posteriormente, queda en manos del profesor quien debe indicar si quiere introducir una explicación o no, junto con el ejercicio propuesto.

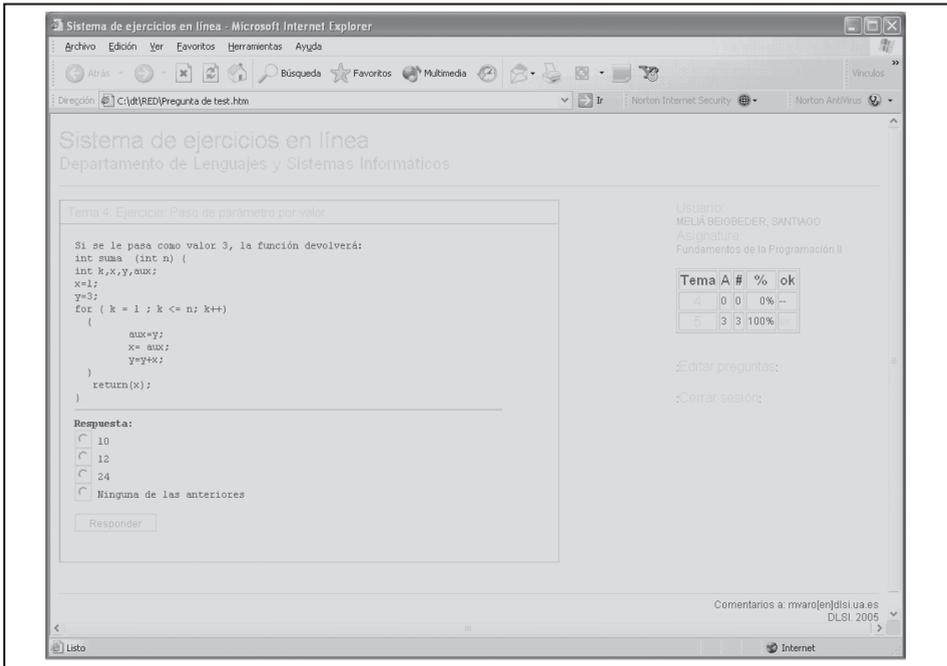


Figura 3. Ejemplo de pregunta de tipo test.

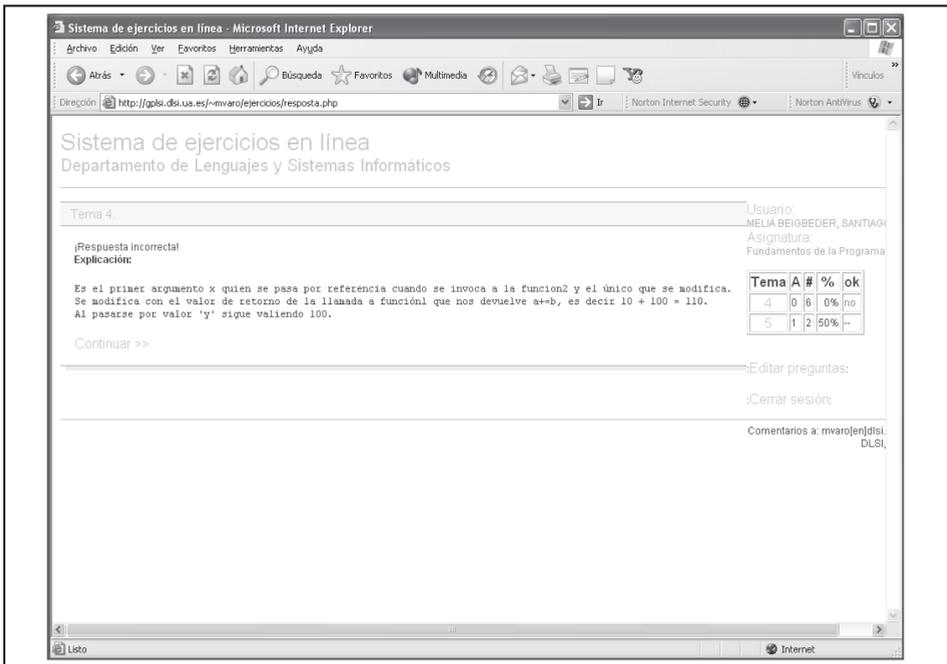


Figura 4. Ejemplo de respuesta de test incorrecta.

3.2.2. Ejercicio de programación

La forma de plantear este tipo de ejercicio al alumno es proponerle que realice un módulo o función de un programa. El enunciado del ejercicio indica cuál es el conjunto de tareas que debe realizar la función, cuáles son los datos que va a recibir como entrada y cuáles va a devolver como salida. Desde el punto de vista de la asignatura de fundamentos de la programación 2, este tipo de ejercicios aparece en multitud de ocasiones cuando nos referimos a temas como el manejo de vectores, tratamiento de registros y estructuras dinámicas. Es por ello que hemos podido introducir fácilmente dichos ejercicios dentro del programa SEL.

En el ejemplo que presentamos en la figura 5, mostramos la aplicación SEL cuando solicita un ejercicio de programación del tema 5 referente a los vectores. En este ejercicio se le solicita al alumno que realice una función llamada “ObtenerMayorMenor”, programada en el lenguaje C++, que realice la tarea de devolver el mayor y menor elemento del vector, así recibirá como entrada un vector de enteros con su longitud y devolverá como salida un número mayor y menor.

La tarea comienza para el alumno, definiendo dicha función, y teniendo en cuenta primordialmente los parámetros de la función, y el nombre que dicha función tiene. Esto es imprescindible ya que si no introduce el nombre correcta-

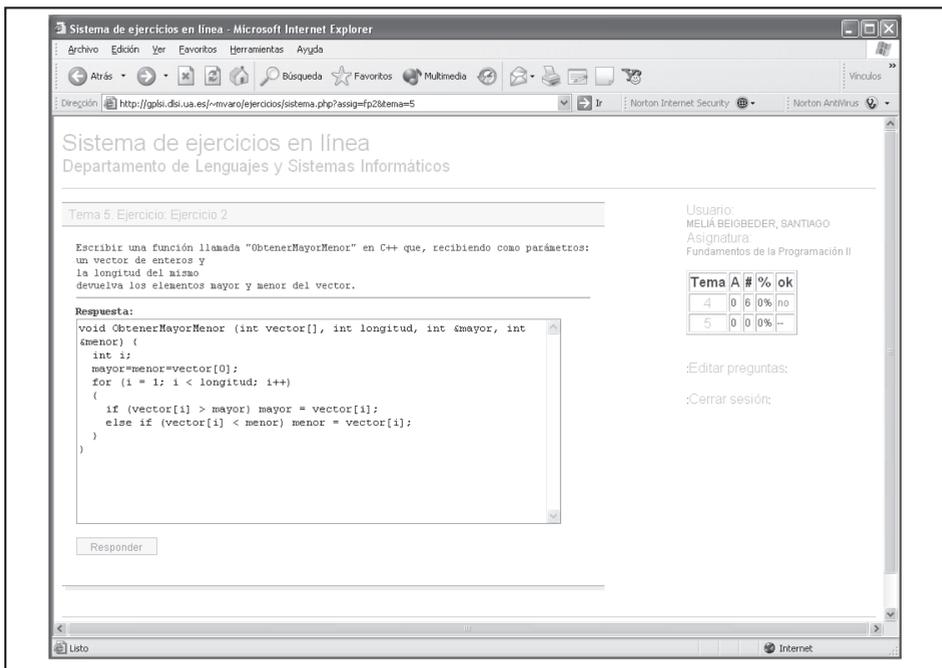


Figura 5. Ejercicio de programación.

mente, la aplicación SEL no la dará como válida. A partir de ahí, el programa intentará compilar la función y hará las comprobaciones necesarias para ver si la ha realizado convenientemente.

Una vez introducido el código de la función, damos al botón responder y nos aparecerá la salida que vemos en la Fig. 6. Se nos muestran dos salidas, (1) la salida del compilador, que nos indica los errores de compilación o nada si compila bien. Y (2) la salida del comprobador indicando si han ido correctamente todas las pruebas a las que ha sometido a la función. En el caso de que las pruebas hayan sido correctas, nos muestra el mensaje de resultado correcto.

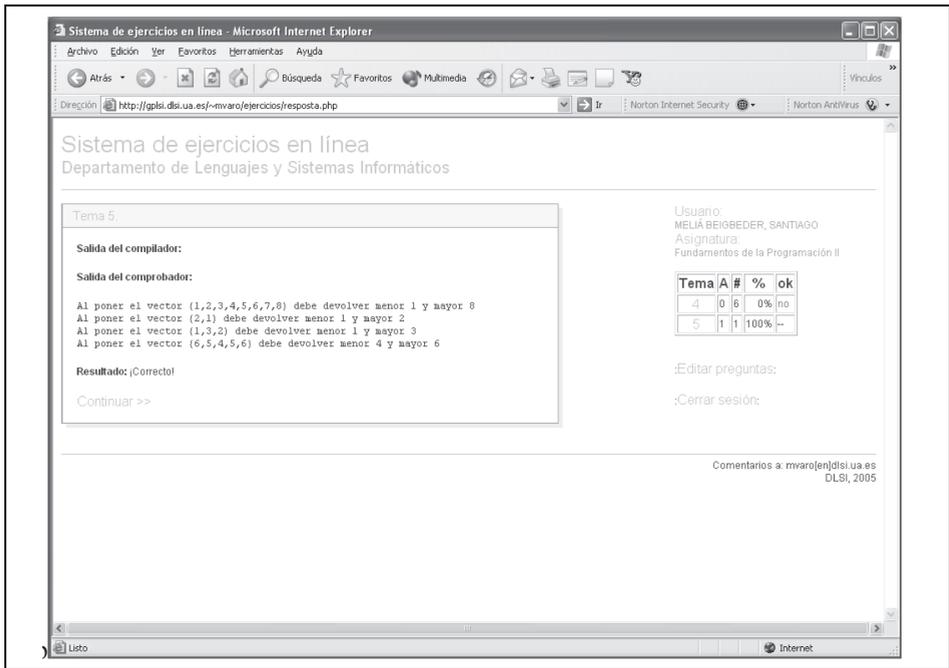


Figura 6. Resultado correcto en un ejercicio de programación.

En ocasiones, el alumno puede encontrarse con errores de compilación, que son los mismos que devolvería cualquier entorno de programación, o con errores de comprobación porque ha diseñado un programa que funciona correctamente pero no cumple con los requisitos que le pide el ejercicio. En caso de que haya un error aparecerá una etiqueta que indique “me rindo”, este mecanismo lo hemos propuesto para aquellas ocasiones en las que el alumno se vea incapaz de realizar el programa y así pueda pasar a la siguiente pregunta.

3.3. VISTA DEL PROFESOR

Esta vista proporciona al profesor el acceso al sistema para realizar básicamente dos tareas, (1) la introducción de los ejercicios de los diferentes temas, indicando la información sobre la asignatura, el tema y la respuesta válida del ejercicio. (2) Permite el control sobre las respuestas realizadas por los alumnos, de esta manera puede averiguar quiénes son los alumnos que realizan los ejercicios, qué días los realizan y cómo.

Por un lado, para poder realizar la introducción de los diferentes ejercicios, el profesor va a disponer de dos tipos diferentes de páginas de introducción de ejercicios, cada una de ellas acondicionadas para cada tipo de ejercicio, de test y de programación.

Por otro lado, va a poder tener acceso a las diferentes respuestas de forma individualizada para cada uno de los alumnos.

3.3.1. Administración del ejercicio de test

El sistema le va a facilitar al profesor un formulario particular para los ejercicios de tipo test, que le va a permitir introducir y modificar cada uno de los ejercicios de una forma muy sencilla.

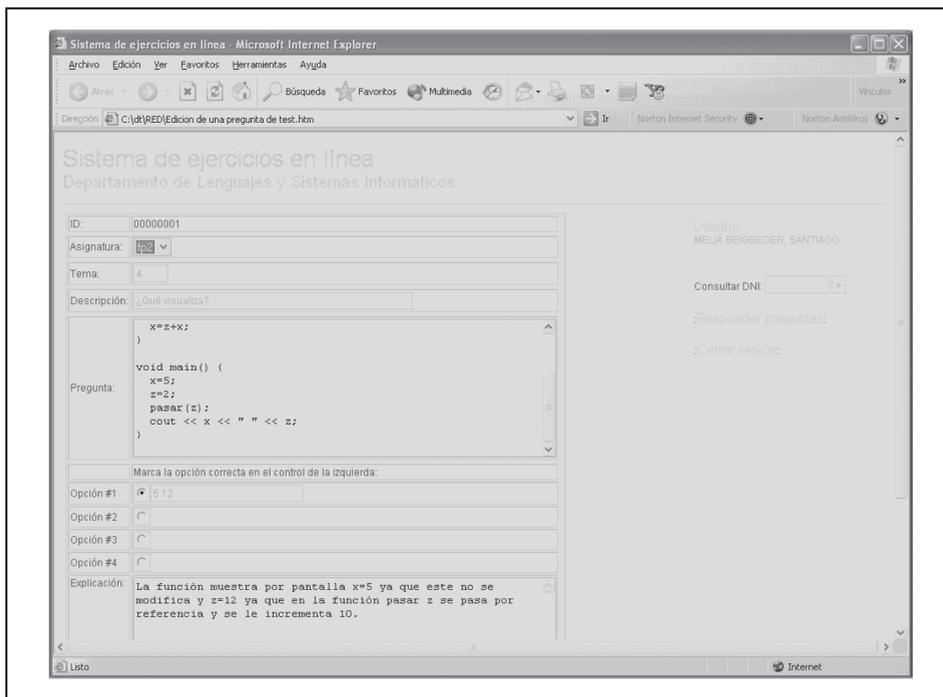


Figura 7. Formulario de administración del ejercicio de test.

Los datos se van a introducir en el formulario para el tipo test (ver figura 7) son los siguientes:

- ID: número identificador del ejercicio dentro de un determinado tema y asignatura.
- Asignatura: indicamos la asignatura de programación a la que pertenece el ejercicio.
- Descripción: le damos un título a la pregunta. Este título nos va a servir posteriormente para distinguir dicha pregunta del resto, en el formulario general de preguntas.
- Pregunta: en dicho campo vamos a tener el cuerpo de la pregunta.
- Opción: aquí disponemos de la posibilidad de activar el número de opciones que vamos a mostrar, y además introducir el texto que aparece junto a cada opción.
- Explicación: es la aclaración del motivo sobre la solución indicada. Aparecerá tanto si acierta como si falla el alumno.

3.3.2. Administración del ejercicio de programación

Hemos definido dentro de SEL otro formulario para que el profesor administre los ejercicios de programación. Este tipo de ejercicio exige al profesor un

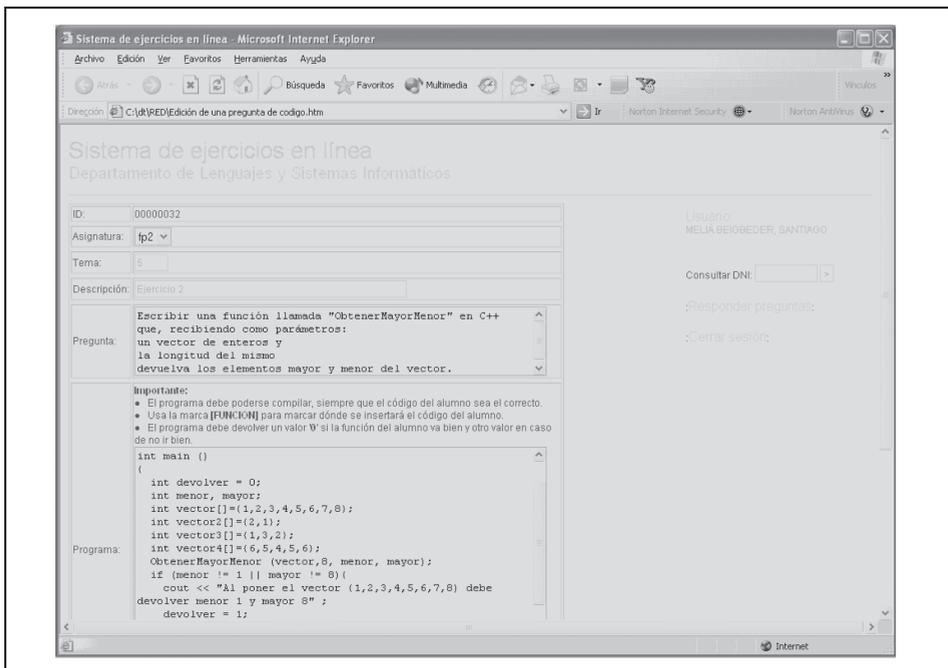


Figura 8. Formulario de administración del ejercicio de programación.

mayor trabajo, ya que implica la introducción de la pregunta y del módulo de validación que compruebe que el programa introducido por el alumno compila y funciona correctamente.

Los datos que se le van a solicitar en el formulario del ejercicio de programación (ver figura 8) son los siguientes:

- ID: número identificador del ejercicio dentro de un determinado tema y asignatura.
- Asignatura: indicamos la asignatura de programación a la que pertenece el ejercicio.
- Descripción: le damos un título a la pregunta. Además, este título nos va a servir posteriormente para distinguir dicha pregunta del resto, en el formulario general de preguntas.
- Pregunta: en dicho campo vamos a introducir el cuerpo de la pregunta.
- Programa: en este campo introducimos el programa validador que invocará al módulo o función que realice el alumno. Para ello seguimos un convenio en el cual la función que introduzca el alumno la ubicaremos donde el profesor coloque la etiqueta **[funcion]**. De esta manera, el programa junto con la función del alumno se compilarán conjuntamente, detectando los fallos si los habido en la definición de los parámetros o el valor de res-

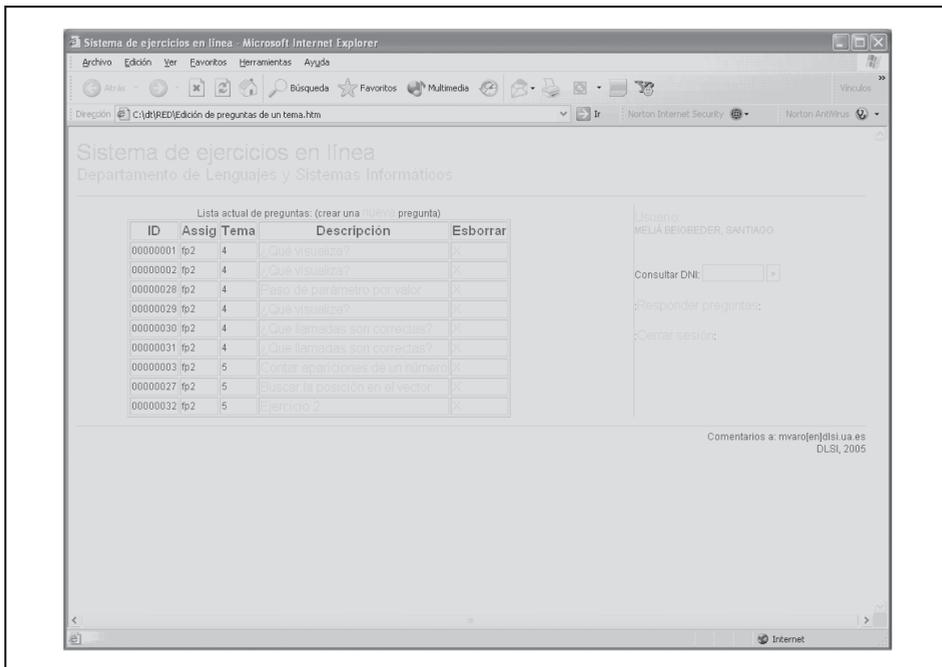


Figura 9. Formulario de acceso a la lista de ejercicios.

puesta de la función introducida. Por otro lado, el programa validador realizará las llamadas necesarias a la función para comprobar que este funciona correctamente. Junto a cada función se introducirá un mensaje de salida, para que muestre al alumno indicándole si dicha prueba funcionó o por el contrario falló al realizarse.

Por último, en la figura 9, podemos ver un formulario general que muestra al profesor la lista actual de ejercicios introducidos. De esta manera este formulario nos permitirá introducir una pregunta nueva cuando le damos a la palabra “nueva” que aparece en la parte superior de la lista de ejercicios. Por otro lado, podemos también editar una misma pregunta ya introducida previamente, o podemos borrarla mediante el botón que aparece debajo de la cabecera “Esborrar”.

3.3.3. Control de accesos a los alumnos

Como parte de una aplicación dedicada a la docencia, resultaba imprescindible que el profesor dispusiera de la posibilidad de acceder a los progresos realizados por los diferentes alumnos. Es por ello que facilitamos al profesor de una consulta sobre cada uno de los accesos que había realizado el alumno para responder a los ejercicios planteados. El mecanismo es el siguiente: el profesor procede a seleccionar un alumno por su DNI, y a continuación le aparece la lista de

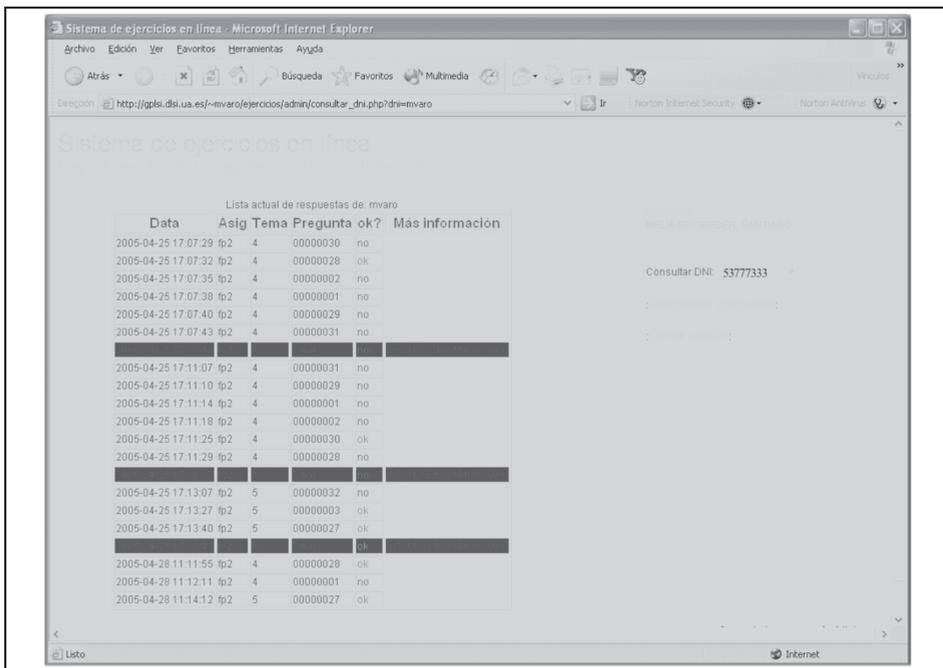


Figura 10. Lista de respuestas de un alumno.

respuestas que éste ha tenido hasta la fecha. En la figura 10, podemos ver cómo hemos introducido un DNI en el campo de entrada Consultar DNI; una vez lanzada la consulta nos aparece en la parte izquierda de la pantalla una tabla en la que se muestran las diferentes respuestas del alumno con el login “mvaro”.

Se puede apreciar en la tabla que aparecen diferentes campos, vamos a explicar detalladamente cada uno de ellos:

- **Date:** indica la fecha y la hora en la que el usuario respondió a al ejercicio.
- **Asig:** indica la asignatura a la que pertenece el ejercicio.
- **Tema:** tema de la asignatura a la que pertenece el ejercicio.
- **Pregunta:** muestra el identificador numérico que representa al ejercicio.
- **Ok?:** muestra si ha realizado correctamente el ejercicio mostrando en el campo el texto ok en verde, o si ha respondido incorrectamente al ejercicio mostrando un no en rojo.
- **Información:** este campo es utilizado cuando un alumno ha finalizado con un determinado tema. En el caso de que sea así, muestra la cantidad de aciertos y fracasos, junto al porcentaje y además si ha superado el mínimo porcentaje necesario para superar el tema. En la figura 10, podemos ver en el primer final la cantidad de acierto respecto al número de respuestas, que es (1/6), es decir, un acierto de 6 ejercicios. Por lo tanto, el porcentaje de acierto es del 16,66%. Como el mínimo exigido es un 60%, dicho alumno no ha superado el tema.

4. TRABAJOS RELACIONADOS

En los últimos años el uso de Internet como herramienta docente dentro del ámbito universitario ha aumentado de forma muy notable. Existen muchas aplicaciones que ayudan tanto al alumno como al profesorado, ya sea en cualquier tipo de gestión, acceso al material docente o simplemente como herramienta de comunicación entre alumno y profesor. Un ejemplo muy cercano de este tipo de aplicación es el Campus Virtual de la Universidad. Más concretamente, si nos centramos en los trabajos relacionados con el aprendizaje a través de Internet, nos encontramos con muchos trabajos como los de Gayo et al. (2001), Barchino et al. (2001), Luján-Mora y Llopis (2002) y Más y Acosta (2001).

De los anteriores trabajos nos interesan principalmente tres por su semejanza con nuestra propuesta. Por un lado, en Barchino et al. (2001) se presenta EDVI que es un sistema de apoyo a la enseñanza presencial basado en Internet. El sistema permite publicar materiales docentes, y además, los alumnos pueden verificar los conocimientos adquiridos en cada tema, mediante la realización de exámenes de tipo test generados por el sistema, a diferencia de nuestro trabajo donde las preguntas pueden realizarse no solamente de tipo test, sino que permiten la realización de ejercicios más complejos.

Por otro lado, en Más y Acosta (2001) se presenta otro sistema basado en la Web, que también posee ciertas similitudes con nuestra propuesta. En este sistema el alumno puede acceder a través de Internet a la lista de problemas generados de forma aleatoria a partir de una base de datos. Sin embargo, esta propuesta carece de interactividad e inmediatez de nuestro sistema (no dispone de un módulo para preguntas-respuestas con corrección automática).

Por último, mencionar el sistema AWAM (Lujan-Mora y Llopis, 2002), que se diseñó hace tres años en nuestra Red con un objetivo muy similar a nuestro sistema SEL. AWAM intentaba cubrir los ejercicios realizados dentro de las asignaturas de programación, utilizando ejercicios de tipo test. Sin embargo, en base a la experiencia adquirida, dicho sistema ha resultado insuficiente para la docencia de la programación, ya que existe una gran cantidad de ejercicios que no pueden ser cubiertos utilizando únicamente el mecanismo del test.

5. CONCLUSIONES Y TRABAJOS FUTUROS

En el presente trabajo hemos presentado un programa llamado SEL, cuyo principal objetivo es mejorar la docencia en la parte teórica en las asignaturas de programación. Para ello, presenta un mecanismo para la publicación de los ejercicios teóricos en Internet, para que el alumno pueda acceder desde cualquier lugar y cualquier momento a dichos ejercicios. Además, proporciona un mecanismo de control sobre los ejercicios que ayuda al profesor a conocer la evolución de los diferentes alumnos. En contraposición con otras propuestas presentadas previamente, que se limitaban únicamente a la publicación de diferentes ejercicios de tipo test, SEL presenta la posibilidad de proponer ejercicios de programación, que además son introducidos y validados por el propio profesor.

El siguiente paso consiste en la adaptación del presente sistema al resto de asignaturas que forman parte de nuestra red. Ahora mismo el programa está desarrollada para su correcto funcionamiento en la asignatura de primer ciclo, Fundamentos de la programación 2. Sin embargo, se presentan retos importantes ya que existen otras asignaturas donde la programación implica además la utilización de librerías externas, estructuras más complejas, etc. lo que supone unas mayores exigencias técnicas a la hora de desarrollar adaptar el programa SEL.

Por otro lado, en el curso 2005-2006, se ha implantado el programa dentro de la asignatura de Fundamentos de la programación 2. Esto va a ayudar a evaluar la satisfacción tanto del alumnado como del propio profesorado en su uso. Además, podremos obtener las primeras opiniones sobre la aplicación, si realmente es una herramienta de autoaprendizaje conveniente para los alumnos, y si por otro lado, es suficiente el mecanismo ofrecido para el correcto control del profesor sobre la evolución de ellos, esto nos permitirá optimizar la herramienta introduciendo mejoras basadas en la experiencia.

6. REFERENCIAS

- BARCHINO, R.; GUTIÉRREZ, J. M; GARCÍA, E.; HILERA, J. R. (2001) *EDVI: Un sistema de apoyo a la enseñanza presencial basado en Internet*. Actas de JENUI (pp. 451-453).
- GAYO, D.; LÓPEZ, B.; LABRA, J. E.(2001) *Desarrollo del portal web de la E.U. de Ingeniería Técnica en Informática de Oviedo*. Actas del JENUI. (pp. 39-44).
- LUJÁN-MORA, S.; LLOPIS, F.(2002) *Resolución de ejercicios de programación en la Web*. VII Jornadas de enseñanza universitaria de la Informática. (pp. 29-36).
- MÁS, R.; LACOSTA, I. (2001) *Aplicaciones de Internet a la Enseñanza: Un Sistema de Autoevaluación*. Actas de JENUI. (pp. 500-503).