easyTest: una herramienta para la generación y corrección automáticas de exámenes tipo test.

Macario Polo Usaola y Francisco Ruiz González
Grupo Alarcos - Departamento de Informática
Universidad de Castilla-La Mancha
Ronda de Calatrava, 5
13071-Ciudad Real
e-mail: mpolo@inf-cr.uclm.es

1. Introducción

easyTest es una herramienta desarrollada en Java que genera de forma automática tantos exámenes tipo tests como el usuario (habitualmente un profesor) elija.

Todos los exámenes generados constan de las mismas preguntas, pero colocadas en distinto orden en cada examen. Así mismo, las respuestas que se ofrecen con cada pregunta están también colocadas de distinto modo en cada examen. Estas características hacen que se dificulten la copia de exámenes y los "soplos" de respuestas ente los alumnos.

Mientras easyTest va generando cada examen, determina la secuencia correcta de respuestas de las preguntas contenidas en ese examen. Esta secuencia de respuestas correctas está constituida por una serie de números, que corresponden al orden que la opción correcta ocupa en las respuestas ofrecidas.

Cada examen generado por easyTest recibe un número único. A la hora de corregir, el profesor sólo debe buscar en la lista de exámenes el número del examen correspondiente y teclear la secuencia introducida por el alumno. Del mismo modo, puede asociar de forma muy cómoda el examen a un alumno, que puede estar previamente almacenado en la base de datos. Acto seguido, easyTest calcula la nota según unos parámetros que el profesor puede establecer y la almacena en la base de datos.

easyTest no es realmente una herramienta con fines de docencia exclusiva en informática. Sin embargo, es en este foro donde hemos querido darlo a conocer para ponerlo a disposición de todos los compañeros que quieran hacer uso del programa.

En la página web de easyTest (http://www.inf-cr.uclm.es/www/mpolo/easytest.html) puede encontrarse el manual del usuario del programa, por lo que en este artículo describiremos muy brevemente cómo funciona y explicaremos algunos detalles de su diseño e implementación que pueden resultar útiles para la enseñanza de ciertas asignaturas.

2. Breve descripción del funcionamiento de easyTest

easyTest genera *n* exámenes tipo test a partir de una "plantilla de examen" construida por el usuario. La plantilla consta de tantas preguntas como se desee, y cada pregunta con cierto número de respuestas. A medida que el usuario construye la plantilla de examen, va indicando cuál de las respuestas asociadas a una pregunta es la correcta.

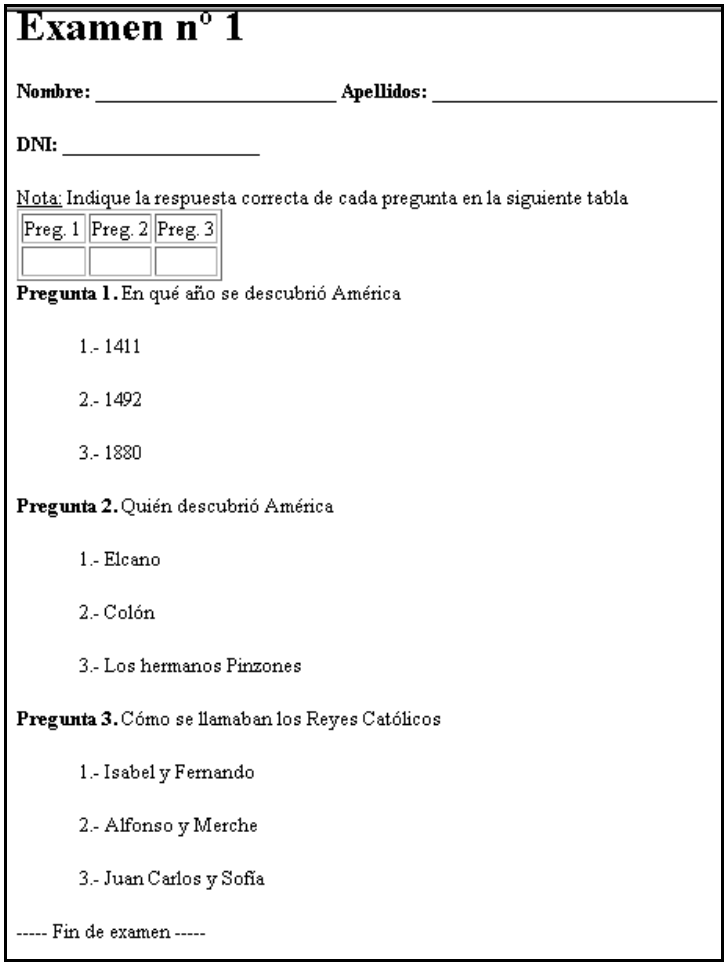
Una vez que la plantilla de examen ha sido construida, el usuario puede generar tantos exámenes como desee. Las preguntas de cada examen son colocadas al azar, lo que también ocurre con las repuestas de cada pregunta. Este proceso garantiza (casi con total seguridad) que los alumnos no cometan fraude al contestar los exámenes.

Todos los exámenes son generados juntos en un fichero HTML (Figura 1). Dentro de cada plantilla, a cada examen se le asigna un número único, que luego se utilizará para asociarlo a un alumno y poder calificarle. Cada examen comienza con una tabla en la que el alumno debe anotar las respuestas que considera correctas, y termina con el texto "----- Fin de examen -----", lo que facilita el tratamiento de este fichero con un procesador de textos (por

ejemplo: para que cada examen empiece en una hoja nueva, podemos decir al procesador de textos que sustituya la marca de fin de examen por un salto de página).

La corrección de los exámenes es muy sencilla, ya que basta que el profesor teclee la secuencia de respuestas introducida por cada alumno. easyTest conoce la secuencia correcta de respuestas asociada a cada examen; de este modo, easyTest calcula la nota del examen calculando el número de aciertos, errores y preguntas en blanco, y utilizando las puntuaciones que el profesor hay asignado a estos tres posibles tipos de respuestas.

El programa almacena toda la información en una base de datos hecha con Microsoft Access. En ella se almacenan todos los datos de plantillas, exámenes, alumnos, preguntas, respuestas y notas.



Examen nº 1

Nombre: _____ **Apellidos:** _____

DNI: _____

Nota: Indique la respuesta correcta de cada pregunta en la siguiente tabla

Preg. 1	Preg. 2	Preg. 3

Pregunta 1. En qué año se descubrió América

1.- 1411

2.- 1492

3.- 1880

Pregunta 2. Quién descubrió América

1.- Elcano

2.- Colón

3.- Los hermanos Pinzones

Pregunta 3. Cómo se llamaban los Reyes Católicos

1.- Isabel y Fernando

2.- Alfonso y Merche

3.- Juan Carlos y Sofía

----- Fin de examen -----

Figura *¡Error!Argumento de modificador desconocido.*. ***Aspecto del fichero de exámenes.***

3. Descripción del diseño y la implementación de easyTest

Además de las capacidades de generación y corrección de exámenes, consideramos que easyTest posee varias características que lo convierten en una interesante herramienta docente:

- Posee arquitectura multicapa
- Utiliza el patrón RCRUD para proporcionar persistencia a las clases de la capa de dominio (*Alumno*, *Examen*, *Plantilla*, etc.)

- El algoritmo de desordenación de las preguntas y respuestas es muy eficiente

3.1. Arquitectura multicapa

easyTest es una aplicación orientada a objetos estructurada en las capas de presentación, dominio y almacenamiento.

En las capa de presentación se encuentran situadas todas las pantallas de la aplicación. Existe un gran número de pantallas que muestran una ficha de alguna de las clases de la capa de dominio con persistencia que hemos mencionado. Todas estas "pantallas tipo ficha" contienen una referencia a un objeto de la capa de dominio, al cual delegan las operaciones persistentes (como insertarse, modificarse o borrarse) que desencadena el usuario. La Figura 2 muestra una pantalla de este tipo, concretamente la que se utiliza para manipular fichas de alumnos.

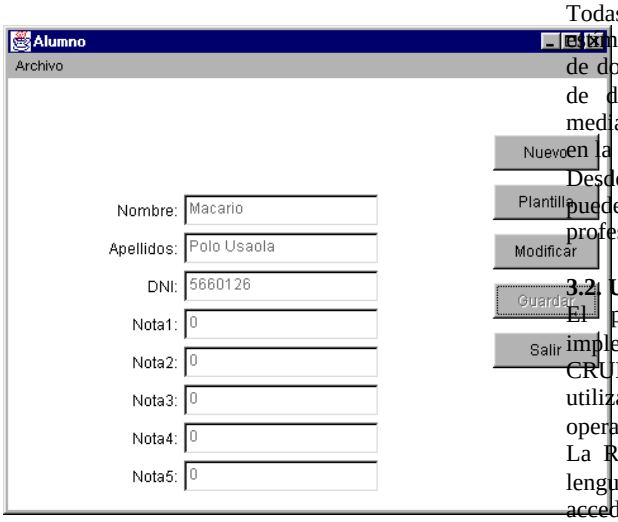


Figura ¡Error!Argumento de modificador desconocido.**. Ventana para crear un nuevo alumno**

Los botones que aparecen en la ventana de la Figura 2 se encuentran en todas las pantallas tipo ficha y tienen el siguiente comportamiento:

- Mediante el botón "Nuevo", se habilitan y vacían las cajas de texto mostradas en la pantalla para que el usuario pueda escribir información. Cuando, más adelante, se pulse el botón "Guardar", se producirá la inserción de este alumno en la base de datos.
- Mediante el botón "Plantilla", se habilitan las cajas de texto pero no se vacían. Cuando se pulse "Guardar", se insertará un nuevo alumno. Por tanto, esta opción utiliza el registro actual como plantilla para uno nuevo.
- Mediante el botón "Modificar", se habilitan las cajas de texto para que el usuario pueda modificar su contenido; cuando, posteriormente, se pulse el botón "Guardar", ejecuta una operación de actualización sobre la base de datos.
- El menú "Archivo" mostrado en la Figura 1 dispone de la opción "Eliminar", que borra el registro que se está mostrando de la base de datos.

Todas estas operaciones producen algún tipo de estímulo en el correspondiente objeto de la capa de dominio. El acceso de los objetos de la capa de dominio a la base de datos se realiza mediante un agente de base de datos (contenido en la clase "Inicio").

Desde luego, la observación de esta estructura puede resultar muy interesante para los profesores que enseñen Ingeniería del Software.

3.2. Uso del patrón RCRUD

El patrón RCRUD (Polo et al., 2001a) implementa mediante Reflexión las operaciones CRUD (Create, Read, Update y Delete), que se utiliza para proporcionar a una clase las cuatro operaciones básicas de persistencia.

La Reflexión es una característica de algunos lenguajes, como Java, que permite a un objeto acceder a su estructura interna en tiempo de ejecución. Así, por ejemplo, un objeto de Java puede conocer cómo se llaman sus atributos, de qué tipo son, cómo se llaman sus métodos, qué tipo devuelven, cuántos parámetros tiene cada uno y de qué tipo, etc. Hemos aprovechado la Reflexión y los patrones de diseño que hacen corresponder una clase de dominio con una tabla para generar, en tiempo de ejecución, el código de todos los métodos persistentes de un objeto. Puede encontrarse más información acerca del patrón RCRUD en http://zeus.inf-cr.uclm.es/www/mpolo/asig/is2/rcrud.pdf.

La Reflexión no se encuentra en la actualidad muy explotada, aunque está demostrando ser potentísima. Además de la definición de RCRUD, nosotros hemos iniciado algunos trabajos en los que utilizamos Reflexión para automatizar algunas tareas del ciclo de vida software (como Polo et al., 2001b).

3.3. Algoritmo eficiente de desordenación

A partir de las preguntas de una plantilla, y de las respuestas asociadas a cada una, easyTest utiliza dos procesos de desordenación para generar cada examen: en primer lugar, desordena las preguntas de la plantilla para que aparezcan en distinto orden en cada examen; en segundo, desordena las respuestas de cada pregunta para dificultar aún más el fraude.

En las primeras versiones de easyTest, realizábamos varios accesos a disco por cada examen que había que generar, con lo que el proceso empleaba mucho tiempo cuando queríamos generar un número de exámenes elevado. En la versión actual se accede una única vez a la base de datos, de la que se lee toda la información y se lleva a memoria, en la que son realizadas todas las desordenaciones en un tiempo muy pequeño. El algoritmo, sin embargo, puede ser estudiado para mejorar su coste.

Por otro lado, es de destacar que hemos intentado seguir el patrón de diseño "Alta cohesión", con lo que la plantilla es responsable de generar los exámenes, el examen de desordenar las preguntas, la pregunta de desordenar las respuestas, etc.

4. Conclusiones

En este artículo hemos presentado algunas características de easyTest, un programa que genera exámenes tipo tests de forma automática y que permite corregirlos de forma muy cómoda.

Puesto que el manual del usuario puede encontrarse en la web de estas Jornadas y en las de la propia herramienta, hemos preferido dedicar estas páginas a explicar algunas características del programa que pueden resultar útiles para utilizar de ejemplo en la docencia, pensamos que especialmente de Ingeniería del Software: en efecto, easyTest posee arquitectura multicapa, accede a la base de datos mediante un agente, utiliza el patrón RCRUD, etc.

Para terminar, nos gustaría indicar que la primera experiencia "industrial" de utilización de easyTest ha sido en la evaluación de los alumnos del curso de mantenimiento del software, cuya experiencia relatamos en las actas de estas mismas Jornadas.

5. Referencias

Polo, M., Piattini, M. y Ruiz, F. (2001a). *Reflective CRUD (RCRUD: Reflective Create, Read, Update & Delete).* En proceso de "shepperding" en EuroPlop 2001.

Polo, M., Peña, M., Piattini, M. y Ruiz, F. (2001b). *Automatic Testing of Java Programs Using Reflection.* Proc. of the 2nd Workshop on Automated Program Analysis, Testing and Verification. 23rd International Conference on Software Engineering. Toronto, Canadá.