Herramienta Autor para la Gestión de Tests Informatizados dentro del Sistema AHA!

Cristóbal Romero, Sebastián Ventura, César Hervás, Isaac Ríder, Santiago Martín-Palomo
Departamento de Informática y Análisis Numérico
Universidad de Córdoba. (España)
e-mail: {cromero, sventura, chervaz, i82rijii, i02pagas}@uco.es

Resumen: En este artículo presentamos Test Editor, una herramienta autor para la construcción de test informatizados, tanto clásicos como adaptativos, a través del Web. Esta herramienta facilita el desarrollo y mantenimiento de diferentes tipos de test de tipo multi-opción o multi-respuesta, con el objetivo de poder utilizarlos dentro de sistemas educativos basados en web. Test editor es una herramienta modular que permite configurar varios parámetros sobre las preguntas o ítems y los tests. También proporciona información estadística sobre la utilización de los tests, que puede ser utilizada para el mantenimiento de los tests. Esta herramienta se ha integrado dentro del sistema AHA!, pero se puede utilizar también dentro de otros sistemas basados en web. Para probar el funcionamiento de la herramienta se ha utilizado en la creación de un test adaptativo para evaluar a dos grupos de alumnos de un mismo curso de extensión universitaria sobre programación en el lenguaje Java.
Palabras clave: Tests Informatizados Adaptativos, Sistemas para Educación basados en Web.

Abstract: In this paper we describe Test Editor, an authoring tool for building adaptive and classic web-based tests. This tool facilitates the development and maintenance of different types of multiple-choice tests for use in web-based education systems. Test Editor is a modular tool, which lets you configure several parameters about questions or items and tests. It also provides statistical information about tests usage that can be used in tests maintenance. We have integrated the Test Editor with the AHA! system, but it can be used in other web-based systems as well. In order to test the performance of the tool, we have used it to create an adaptive test to evaluate two groups of students of the same university extension course about Java language.
Key words: Computer Adaptive Tests, Web-based Education Systems.

1. Introducción

La educación a distancia ha sido uno de los sectores más beneficiados con la expansión que ha sufrido Internet. El número cada vez mayor de posibilidades de conexión a la Red ha propiciado que la Web sea un medio ideal para poner en contacto a autores y alumnos, superando los obstáculos de tiempo y espacio [Horton 00]. Esto ha producido un aumento considerable del desarrollo de aplicaciones y sistemas especializados en este tipo de enseñanza. Pero la gran mayoría de estas aplicaciones educativas son simplemente una red estática de páginas web. Una propuesta de solución a este problema consiste en un nuevo modelo de sistema llamado Sistema Hipermedia Adaptativo (SHA). Estos sistemas [Brusilovsky 03] son un intento de aumentar la eficacia educativa de los Sistemas Tutores Inteligentes (STI) y los Sistemas Hipermedia (SH), pues permite un término medio entre la enseñanza fuertemente guiada de los primeros y la libre búsqueda en los segundos. Los primeros convierten a la tarea educacional en excesivamente restringida a las órdenes del tutorial y los segundos dejan al usuario demasiada libertad, delegando la tarea de enseñar en el propio usuario del sistema. Es decir, tendremos la acción de un tutor enviando al alumno material a ser estudiado, de forma pedagógica, conforme al dominio y al modelo del estudiante, y habrá también la posibilidad del alumno de recorrer

el material según su voluntad. En la actualidad existen multitud de sistemas hipermedia adaptativos, de entre los que podemos destacar el sistema AHA! (Adaptive Hypermedia Authoring) que permite convertir en adaptativos toda clase de aplicaciones basadas en la Web, a través de un simple pero poderoso motor de adaptación [De Bra et al. 03]. El sistema AHA! es actualmente una de las plataformas adaptativas más utilizadas, al ser de código abierto y disponer de un entorno sencillo, que permite desarrollar aplicaciones que por si solas se adaptan a cada usuario. La arquitectura global del sistema AHA! está inspirada en el modelo de referencia AHAM, que es una extensión del modelo Dexter [Halasz et al. 94] construido para capturar las estructuras y funcionalidades de los SHA existentes y futuros.

Por otro lado, los tests son las herramientas de evaluación más ampliamente utilizadas y desarrolladas en educación basada en web [Brusilovsky 99]. Un test informatizado convencional consiste en una secuencia de preguntas, donde cada pregunta puede ser evaluada como correcta, incorrecta o incompleta. Existen diferentes tipos de tests computerizados, dependiendo del tipo de las preguntas (si/no, multiple-opción/simple-respuesta, rellena, relaciona, etc.) y dependiendo del tipo del algoritmo de control (adaptativo y clásico o convencional). El sistema AHA! ya dispone de una herramienta básica para realizar la evaluación del alumno mediante tests clásicos computerizados de tipo multi-respuesta [Brusilovsky 99]. Este tipo de tests es el traspaso directo de los tests en papel a tests informatizados, por lo que están formados por un número de preguntas y respuestas preestablecidas, que se presentan de igual forma a todos los alumnos. La puntuación es igual a suma de los aciertos y se asume que los errores son aleatorios. Los tests clásicos presentan serios inconvenientes, algunos de estos son: la presentación de las mismas preguntas a todos los usuarios del sistema, número muy elevado de preguntas en cada test, escasa o nula adaptación al usuario según su nivel de conocimiento.

Existe otro tipo de tests denominados Tests Adaptativos Informatizados (TAI) [Wainier 90] que presentan la ventaja de adaptar las preguntas o items mostradas al usuario dependiendo del nivel de conocimiento que posea, en un momento determinado, sobre el tema o concepto que se está evaluando. Utilizan un método adaptativo para seleccionar la siguiente pregunta dependiendo de las respuestas anteriores. Los tests adaptativos intentan imitar lo que haría un examinador humano con experiencia, de forma que si se propone una pregunta que resulta ser demasiado fácil, la siguiente debería ser más difícil y viceversa. Con el uso de test adaptativos se consigue evitar la frustración de los usuarios principiantes (al no ser capaces de responder correctamente preguntas difíciles) y el aburrimiento de los avanzados (al resultarles demasiado sencillas las preguntas fáciles), se reduce significativamente la longitud del test (número de ítems utilizados), además de dar una estimación del nivel de conocimiento con una precisión de al menos tan buena como la de los tests clásicos.

Para poder aprovechar todas estas ventajas se ha desarrollado la aplicación Test Editor [Romero et al. 03], como una mejora dentro del sistema AHA! [De Bra et al. 03] que permite la gestión de baterías de tests tanto clásicas como adaptativas. Además, Test Editor dispone de un motor de evaluación, que permite ejecutar los tests a los usuarios de cursos del sistema, y genera de forma automática información histórica sobre las ejecuciones de las baterías de tests con el propósito de poder mostrar información estadística y de poder calibrar las preguntas o items, recalculando los parámetros adaptativos de las preguntas que formaron parte de las baterías de tests, a partir de las respuestas.

El artículo se ha organizado de la siguiente forma: Primero se hace una introducción a los tests adaptativos informatizados, después se describe la arquitectura utilizada en la herramienta Test Editor, entonces se muestra la funcionalidad tanto desde el punto de vista de autor como de los usuarios, a continuación se presentan las pruebas de evaluación que se han realizado y los resultados de dicha evaluación, y por último se describen las conclusiones y futuras mejoras.

2. Test Adaptativos Informatizados

Un Test Adaptativo Informatizado (TAI) [Wainier 90] es una prueba también denominada batería de test, construida para fines de evaluación psicológica o

educativa, formada por un conjunto de ítems o preguntas que se presentan al alumno y responden mediante un ordenador, siendo su característica fundamental la adaptación progresiva al nivel de competencia que va manifestando la persona.

Los elementos básicos de un Test Adaptativo Informatizado son [Olea et al. 03]:

- Un banco de ítems o preguntas suficientemente grande, con propiedades psicométricas conocidas, es decir, con parámetros estimados desde un modelo de la Teoría de la Respuesta al Ítem (TRI) determinado [Van der Linde et al. 97]: discrimación (parámetro a, que sirve para discriminar los niveles de conocimiento superiores e inferiores a la dificultad del item), dificultad (parámetro b, que indica la facilidad o dificultad de acertar el item), adivinanza o pseudoazar (parámetro c, que representa la probabilidad de acertar el item personas con un nivel de conocimiento extremadamente bajo), olvido, etc.
- Un procedimiento que establezca la manera de seleccionar progresivamente los siguientes ítems a utilizar. Consiste en seleccionar el ítem adecuado en cada momento, es decir, aquel que maximiza la función de información para la estimación actual del nivel de conocimiento del alumno. Esta función de información utiliza una estimación del actual nivel de conocimiento del alumno y la probabilidad de responder correctamente al ítem, que depende de los parámetros del item (a, b, c, etc.).
- Un método estadístico de estimación del nivel de conocimiento de los alumnos. La ecuación que se emplea para obtener el nivel de conocimiento adquirido en cada paso de la realización de un test adaptativo, por parte de un alumno o usuario depende de las respuestas correctas e incorrectas, de la suma de las funciones de información de todos los items que ya han sido evaluados y la estimación anterior del conocimiento.
- Un procedimiento de parada, que indica cuando se debe de detener el test. Algunos ejemplos típicos de criterios de parada son: cuando se alcanza un nivel de precisión prefijado, o lo que es lo mismo, cuando el error estándar asociado con la estimación del conocimiento alcanza un valor mínimo (normalmente un valor menor a 0.33), cuando se alcancen un número máximo de preguntas o cuando se sobrepase un tiempo máximo. Estos criterios se pueden combinar, de forma que el test finalice cuando alguno de los criterios se cumpla.

Todos estos elementos básicos se combinarán en un algoritmo [Rudner 98], que se seguirá para el proceso de evaluación (Ver Figura 2) y es distinto al algoritmo clásico (Ver Figura 1). Un TAI es un algoritmo iterativo que comienza con una estimación inicial del nivel de conocimiento del alumno y que tiene los siguientes pasos: (1) Todas las preguntas que no se han administrado todavía son examinadas para determinar cual será la mejor para ser propuesta a continuación, según el nivel de conocimiento estimado del alumno; (2) la pregunta es planteada y el alumno responde; (3) de acuerdo con la respuesta del alumno, se realiza una nueva estimación de su nivel de conocimiento. Los pasos del 1 al 3 se repiten hasta que se cumpla alguno de los criterios de terminación definidos.

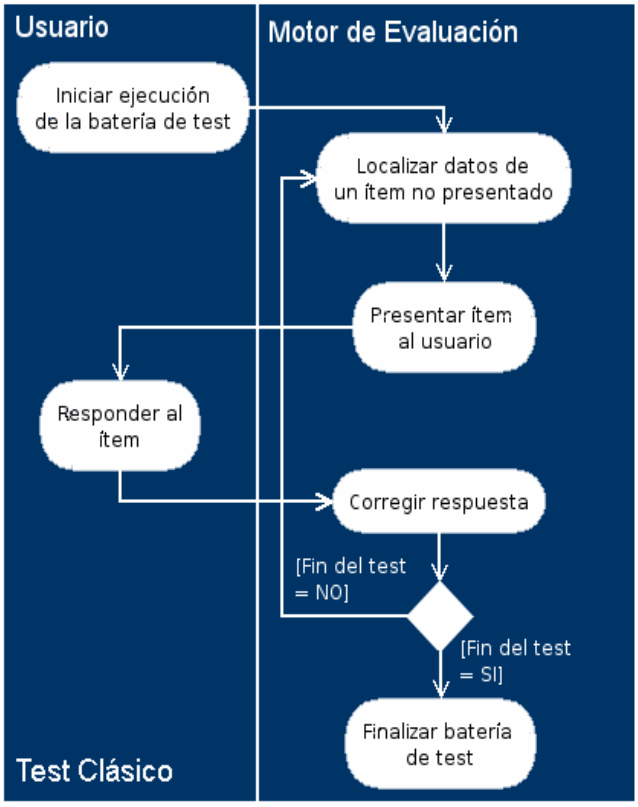


Figura 1: Diagrama de flujo de un test clásico.

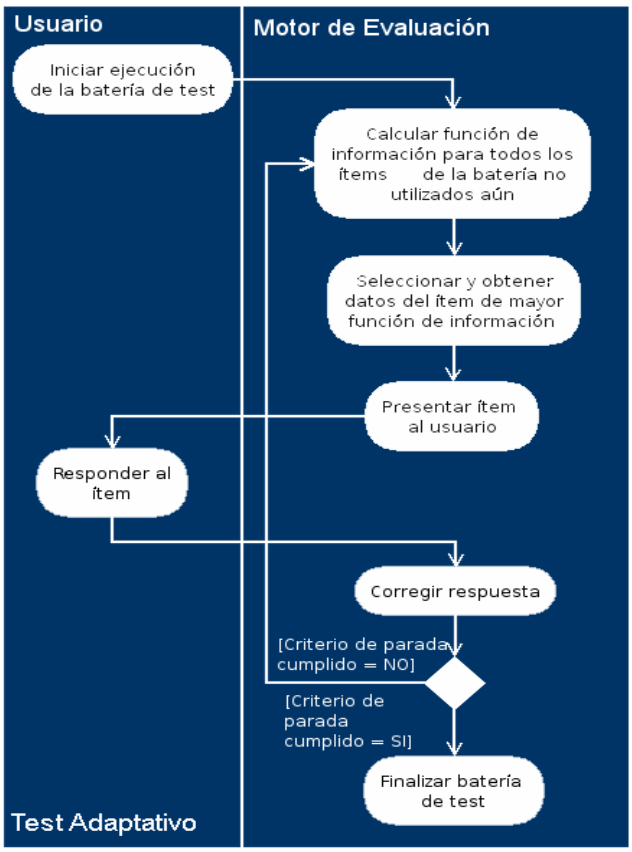


Figura 2: Diagrama de flujo de un test adaptativo.

3. Arquitectura de Test Editor

La aplicación Test Editor [De Bra et al. 03] tiene una arquitectura tipo Cliente-Servidor (ver Figura 2), pero a diferencia de otros sistemas o aplicaciones de este tipo, el procesamiento de la información que requiera de operaciones complejas o prolongadas se realiza en el lado del cliente, reservando al servidor para operaciones de localización de datos, actualización y modificación de los mismos. Se ha utilizado esta filosofía para liberar de procesamiento extra al equipo servidor y mantenerlo, de esta forma, disponible la mayor parte del tiempo, con lo que se consigue que el número de usuarios que pueden ejecutar la aplicación al mismo tiempo sea más elevado, y se reduce considerablemente el tiempo de espera de las distintas ejecuciones de la aplicación por parte de dichos usuarios. Con respecto a las tecnologías utilizadas en el desarrollo, en la parte cliente se ha utilizando Applets Java, en la parte servidor se ha utilizado Servlets Java, y la información de ítems, test e información de ejecuciones de los usuarios se ha utilizado ficheros XML. Los módulos que forman la arquitectura de la aplicación Test Editor (ver Figura 3) son los siguientes:

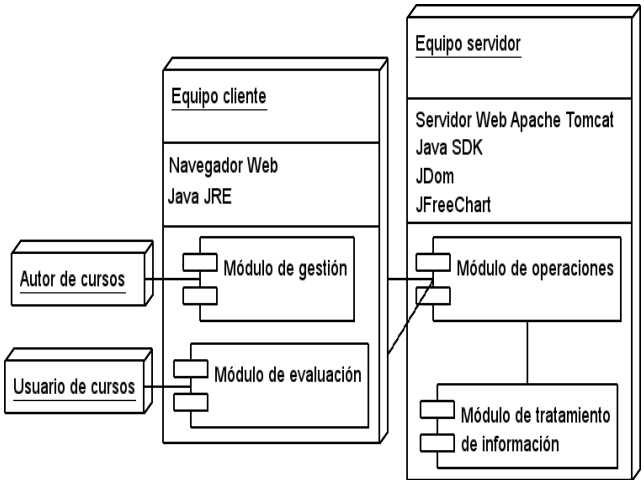


Figura 3: Arquitectura de la aplicación Test Editor.

- **Módulo de gestión**. Permite al autor de tests gestionar toda la información mantenida por la aplicación Test Editor y es además responsable de la ejecución de gran parte de las operaciones que se podrán realizar con dicha información.
- **Módulo de evaluación**. Permite a los usuarios de los test ser evaluados sobre uno o varios conceptos de un curso determinado. Además es responsable de la ejecución de gran parte de las operaciones que se necesitarán para realizar dicha evaluación.
- **Módulo de operaciones**. Es el responsable de recibir las solicitudes de localización, actualización y modificación de la información provenientes de los dos módulos anteriores. También será el encargado de devolver a estos módulos los resultados de la realización de dichas solicitudes.
- **Módulo de tratamiento de información**. Es el responsable de realizar las operaciones físicas sobre la información, es decir, será el encargado del tratamiento de los ficheros que contendrán la información que mantiene la aplicación Test Editor. Recibirá las solicitudes de realización de operaciones sobre dicha información del módulo de operaciones, y devolverá al mismo los resultados de éstas.

- **Módulo de utilidades**. Este módulo es el encargado de realizar ciertas operaciones matemáticas y estadísticas complejas que son necesarias durante la ejecución de un test adaptativo y durante el proceso de calibración de los parámetros adaptativos de los ítems que forman parte de la misma. Es un módulo auxiliar que puede ser utilizado por los cuatro módulos anteriores, por lo que puede ejecutarse tanto en la parte del cliente como en la del servidor.

4. Utilización de Test Editor

Para poder utilizar la herramienta Test Editor, existen distintos requisitos dependiendo del equipo (cliente o servidor) y del tipo de usuario cliente (autor o usuario). En un equipo cliente para un usuario de test, sólo necesita un simple navegador web. En un equipo cliente para un usuario autor de test, necesita que el navegador web tenga instalado un pluging Java con Java RunTime (JRE). Y en el equipo servidor se necesita tener instalado el servidor Web Apache Tomcat, el JRE y el sistema AHA! [De Bra et al. 03]. El proceso general de utilización de la herramienta Test Editor (o ciclo de vida de un test) tanto por parte del autor como por parte de los usuarios, se muestra en la Figura 4.

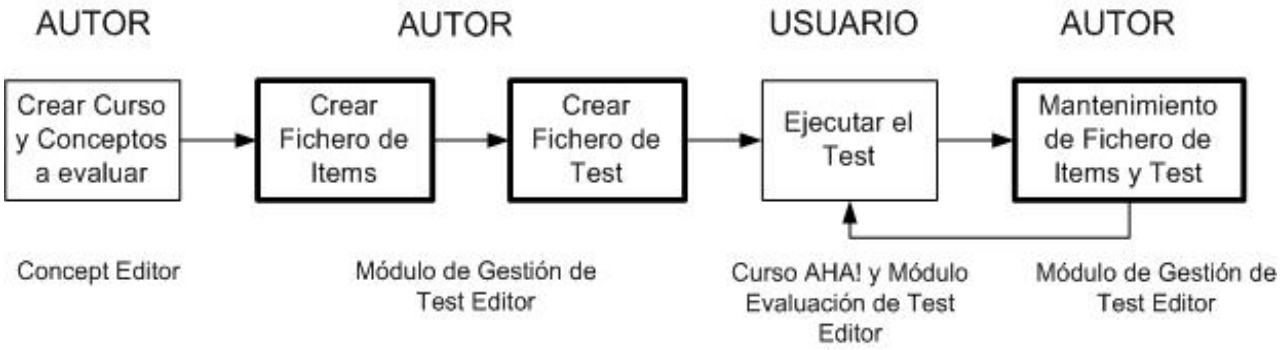


Figura 4: Proceso general de ejecución de Test Editor.

Este proceso, comienza con la construcción del curso y los conceptos que posteriormente van a ser evaluados por el test. Para ello el autor del curso utiliza la herramienta Concept Editor que proporciona AHA! [De Bra et al. 03] para seleccionar el conjunto de páginas HTML/XHTML que representan los conceptos e indicar las relaciones de dependencia entre dichos conceptos. Indicar que se distinguen entre dos tipos de conceptos: conceptos básicos (referidos a un único concepto) y conceptos abstractos (referidos a varios conceptos que forman un tema o un capítulo del curso). Entonces se puede comenzar la creación de los ficheros de ítems que evalúen los conceptos anteriores, utilizando para ello la herramienta Test Editor a la que se accede desde la propia página Web para autores de cursos AHA!. El interfaz de la herramienta consiste en una ventana externa al navegador compuesta por una pantalla principal y un conjunto de pantallas internas a la principal. Por ejemplo, para crear un fichero de ítems el autor debe ir añadiendo las preguntas o items especificando sus parámetros (Ver Figura 5). Cada item es una pregunta formada por una serie de parámetros obligatorios como el enunciado, un número de respuestas mayor de dos, indicación de cuál es la respuesta correcta, y parámetros opcionales como una imagen, una explicación de la respuesta, y los parámetros TRI (dificultad y discriminación, con unos valores por defecto de 0.0 y 1.0 respectivamente). Cada fichero de ítems contiene preguntas sobre un concepto del curso, tanto conceptos básicos como abstractos. Desde el interfaz el autor también podrá añadir nuevos ítems, borrar ítems y examinar ítems previamente introducidos.

A continuación el autor debe crear el fichero de tests a partir de los ficheros de ítems y configurar el test. Para ello primero debe de decidir entre crear un test clásico o un test adaptativo, y si quiere evaluar un concepto básico (en este caso el test sería una actividad que evalúa solo un concepto) o un concepto abstracto (en este caso el test sería un examen de un capítulo o un curso que evalúa un conjunto de conceptos). Después debe seleccionar los ítems concretos que se van a utilizar en el test de entre todos los ítems contenidos en los distintos ficheros de ítems que se refirieren al concepto seleccionado. Para facilitar esta tarea, se dispone además de una forma manual de ir seleccionando item a item los deseados, de una forma automática que aleatoriamente selecciona los items, y de una forma aleatoria con restricciones impuestas por el usuario que deben cumplir los ítems. Entonces se muestran al usuario los ítems finalmente seleccionados para formar parte del test (ver Figura 6).

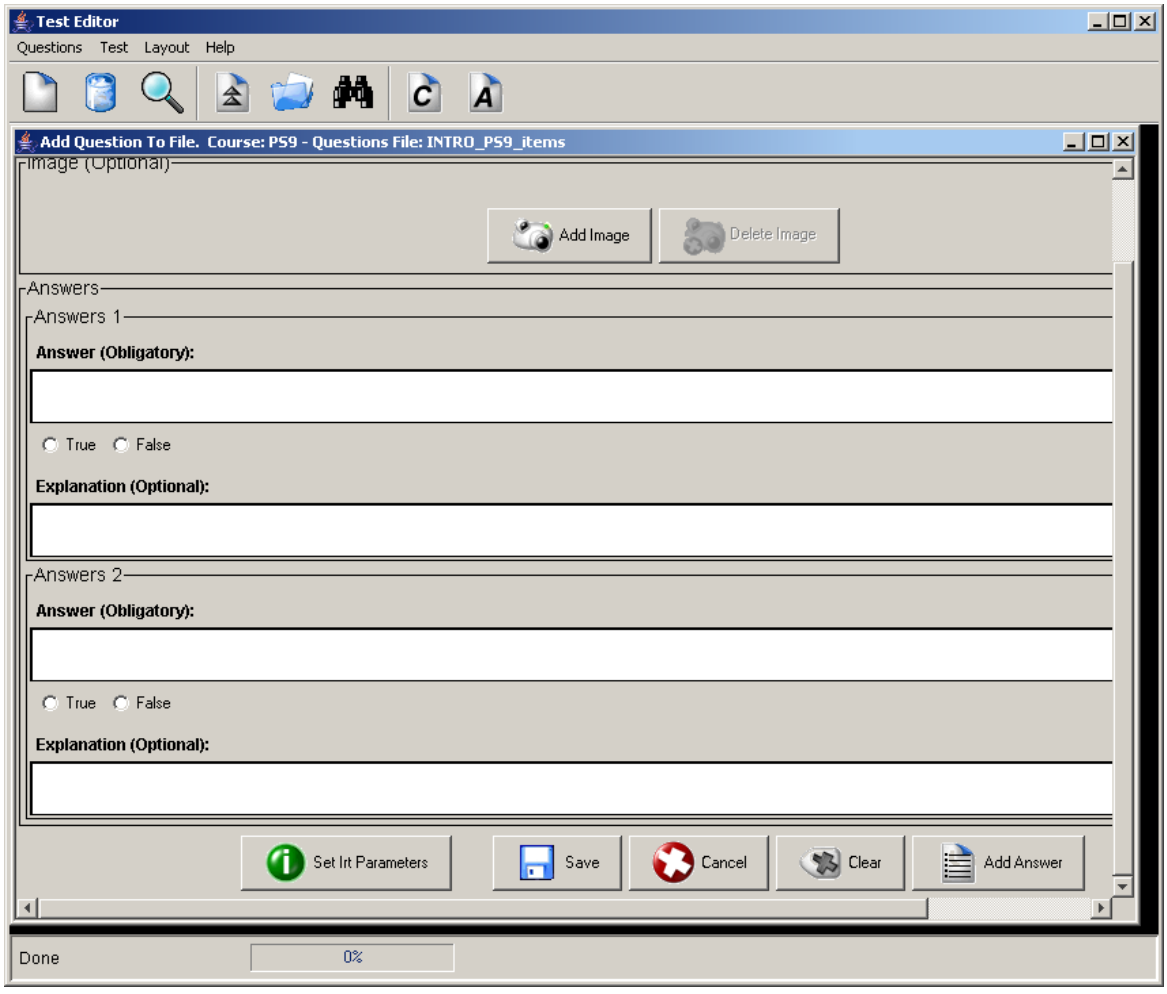


Figura 5: Pantalla para introducir las preguntas o ítems.

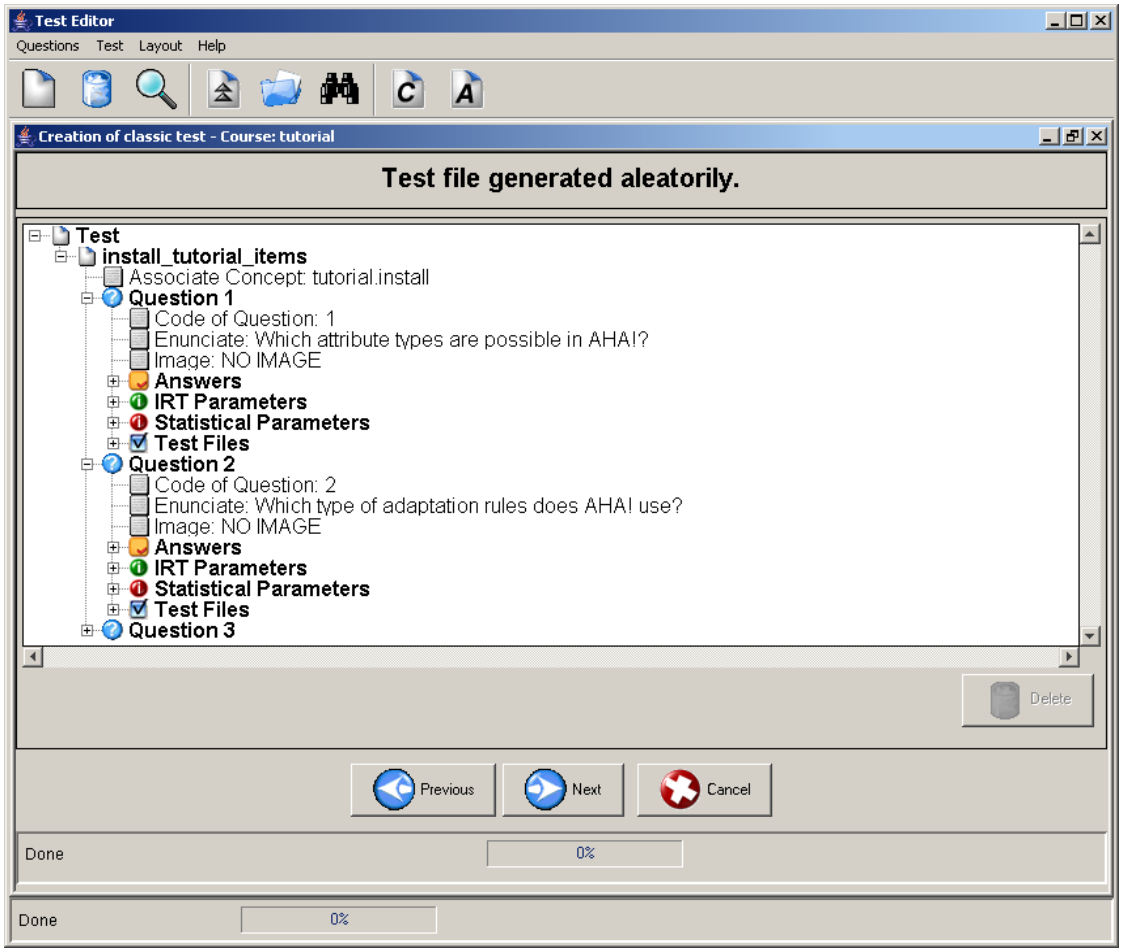


Figura 6: Pantalla de presentación de ítems seleccionados.

Para terminar de crear el test, el autor debe establecer los parámetros del test sobre la presentación, evaluación y ejecución del test. Los parámetros de presentación (ver Figura 7) determinan como se muestran las preguntas al usuario, de manera que se puede especificar si el orden de las preguntas y las respuestas es aleatorio o no, si se muestra o no la respuesta correcta después de contestar, si se muestra o no la explicación de la respuesta, si se va mostrando o no la puntuación y/o nivel de habilidad después de cada pregunta, si se impone o no un tiempo máximo para responder cada pregunta, incluso configurar la apariencia de la página web que va a contener el test, etc. Los parámetros de evaluación del test determinan como se corrigen las preguntas con respecto a la puntación final en el test, de forma que se puede indicar si se penalizan o no las respuestas incorrectas, si se penalizan o no las respuestas sin contestar, que porcentaje de conocimiento representa la nota obtenida en el test sobre el conocimiento del concepto evaluado, etc. Los parámetros de ejecución del test determinan el funcionamiento y parada del test. En el caso de un test clásico sólo se especifica el criterio de parada pudiendo ser un número máximo de preguntas o un tiempo máximo. Y en el caso de un test adaptativo se puede especificar el modelo de TRI utilizado de 1 parámetro, 2 parámetros o 3 parámetros y el criterio de parada, que puede ser un número máximo de preguntas, un tiempo máximo o un valor obtenido del error estándar sobre la habilidad de usuario menor a un valor indicado.

Entonces el test es generado y es publicado dentro del servidor web de AHA! y desde ese instante se encuentra disponible para ser utilizado dentro de un curso de AHA! simplemente haciendo un enlace a la página web generada para el test.

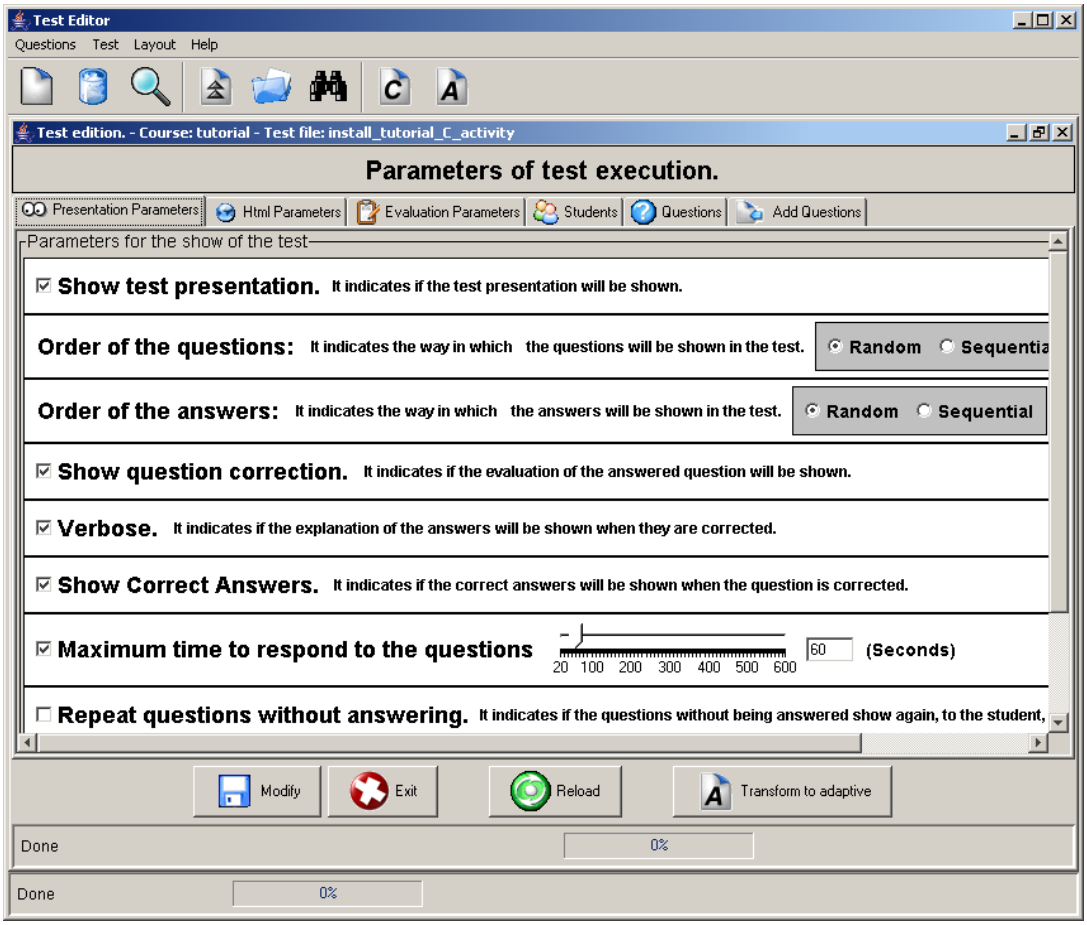


Figura 7: Pantalla de especificación de parámetros de presentación de tests.

Esta página contiene el Applet que es el motor de evaluación de Test Editor. Los usuarios del curso ejecutan el test y son evaluados de forma transparente mediante dicho motor de evaluación que va almacenando toda la información de la ejecución del test. Al comenzar un test lo primero que se muestra al usuario es un pantalla de presentación o descripción del test que muestra información sobre el test, como los nombres de los conceptos que evalúa el test, el número total de ítems que lo forman, el tiempo máximo para responder a cada ítem, si hay penalizaciones por respuestas incorrectas o ítems no contestados, el porcentaje de conocimiento que representa el test, etc. Entonces cuando el usuario desee empezar se van presentando los ítems del test para que seleccione la respuesta correcta (ver Figura 8). La información que se va mostrando es el enunciado del ítem, su imagen asociada si tiene, las posibles respuestas del mismo y un cronometro que indica el tiempo de respuesta si está establecido. Tras cada contestación y si el autor lo indicó se muestra la información referente a la corrección del ítem (si se ha contestado correcta o incorrectamente, la explicación de las respuestas, la respuesta correcta, etc.). De igual forma al finalizar el test y si el autor lo indicó se muestra información sobre la puntuación final obtenida, el número de respuestas correctas e incorrectas y de ítems no contestados, etc.

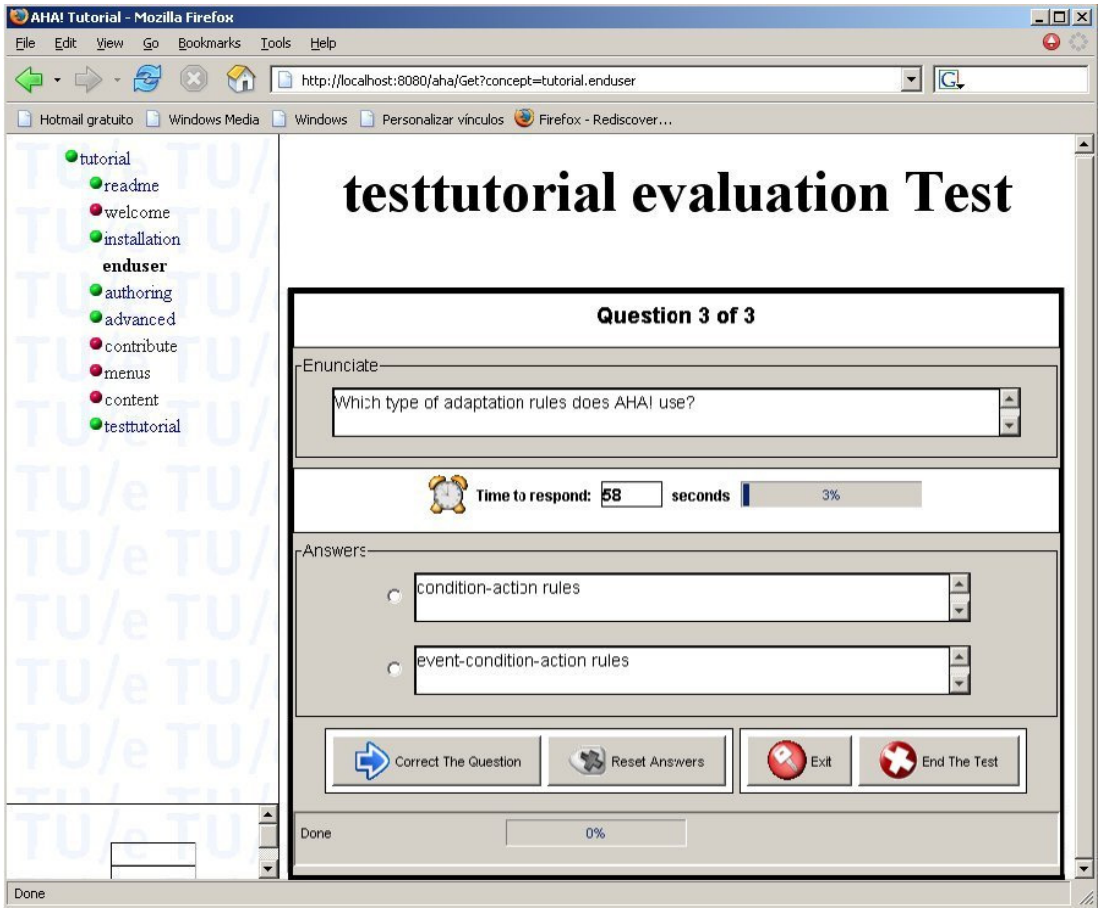
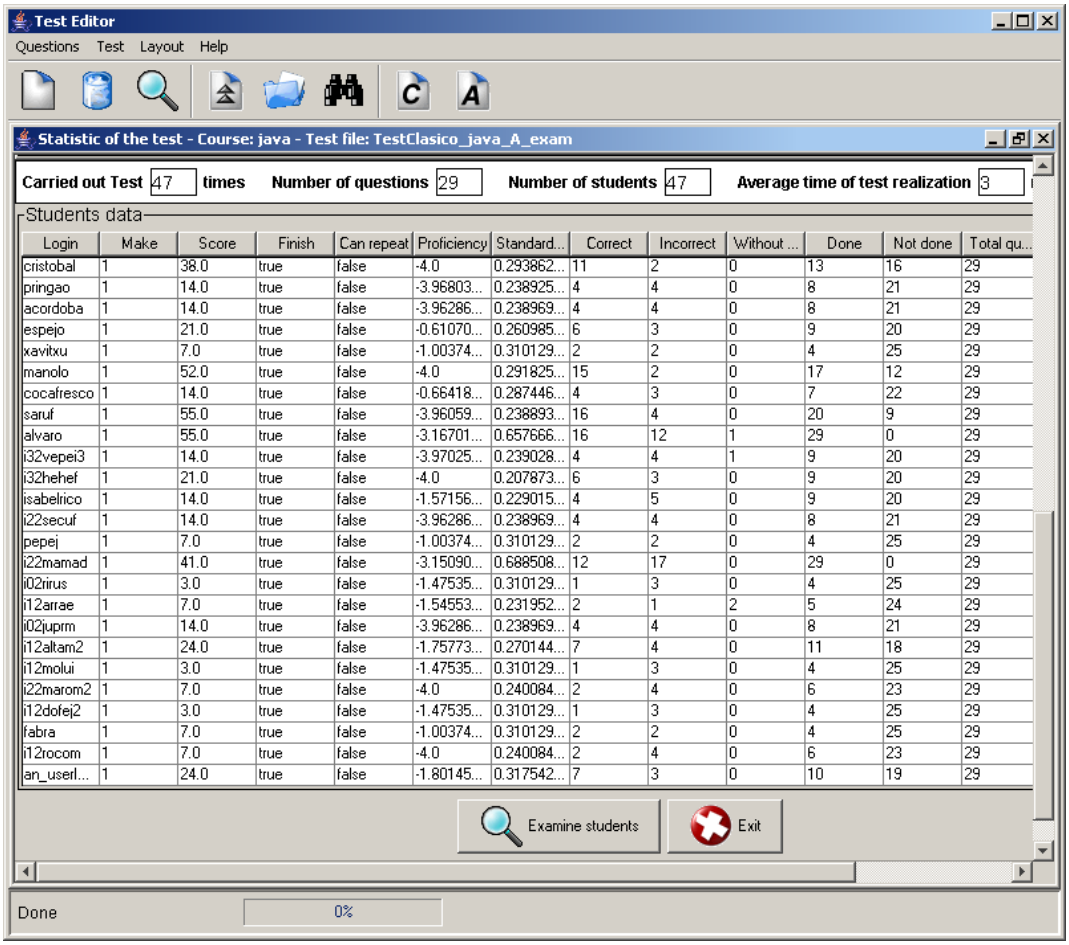


Figura 8: Pantalla de presentación de un test dentro del propio curso tutorial de AHA!.

Posteriormente, el autor del test puede realizar tareas de mantenimiento de los tests utilizando la información estadística de las ejecuciones de los tests. Esta información estadística muestra las puntuaciones obtenidas por los usuarios de cursos que han realizado el test y sobre los niveles de habilidad estimados para dichos usuarios, e información numérica general, el número de ejecuciones totales del test, el tiempo medio que necesitan los usuarios para su realización completa, el número de ítems que la componen, e información sobre las ejecuciones de cada alumno en particular (ver Figura 9). A partir de esta información los autores pueden editar y modificar la información sobre la configuración de ejecución de un test determinado. También pueden añadir nuevos ítems al test, eliminar algunos de los ítems que ya forman parte del test y realizar la conversión de un test de tipo clásico en adaptativo y viceversa. La calibración consiste en recalcular el valor de los parámetros TRI según las respuestas que hayan dado los usuarios de cursos a los ítems, cuando éstos les fueron mostrados mediante un test. Desde la pantalla de calibración se debe seleccionar los bancos de ítems que queremos calibrar e introducir el número de iteraciones que el algoritmo llevará a cabo para calibrar cada ítem, con un mínimo de 1 y un máximo de 99. El procedimiento de calibración que hemos utilizado se denomina estimación de máxima versosimilitud condicionada o 'maximum likelihood estimation' [Backer 92]. Y finalmente se considera que un item está definitivamente calibrado, y por tanto, puede tomar parte en evaluaciones fiables, si manifiesta buenos indicadores (un valor del estadístico chi-cuadrado pequeño y un valor del parámetro de discriminación alto).



Login	Make	Score	Finish	Can repeat	Proficiency	Standard...	Correct	Incorrect	Without ...	Done	Not done	Total qu...
cristobal	1	38.0	true	false	-4.0	0.293862...	11	2	0	13	16	29
pringao	1	14.0	true	false	-3.96803...	0.238925...	4	4	0	8	21	29
acordoba	1	14.0	true	false	-3.96286...	0.238969...	4	4	0	8	21	29
espejo	1	21.0	true	false	-0.61070...	0.260985...	6	3	0	9	20	29
xavitxu	1	7.0	true	false	-1.00374...	0.310129...	2	2	0	4	25	29
manolo	1	52.0	true	false	-4.0	0.291825...	15	2	0	17	12	29
cocafresco	1	14.0	true	false	-0.66418...	0.287446...	4	3	0	7	22	29
saruf	1	55.0	true	false	-3.96059...	0.238893...	16	4	0	20	9	29
alvaro	1	55.0	true	false	-3.16701...	0.657666...	16	12	1	29	0	29
i32vepei3	1	14.0	true	false	-3.97025...	0.239028...	4	4	1	9	20	29
i32hehef	1	21.0	true	false	-4.0	0.207873...	6	3	0	9	20	29
isabelrico	1	14.0	true	false	-1.57156...	0.229015...	4	5	0	9	20	29
i22secuf	1	14.0	true	false	-3.96286...	0.238969...	4	4	0	8	21	29
pepej	1	7.0	true	false	-1.00374...	0.310129...	2	2	0	4	25	29
i22mamad	1	41.0	true	false	-3.15090...	0.688508...	12	17	0	29	0	29
i02rirus	1	3.0	true	false	-1.47535...	0.310129...	1	3	0	4	25	29
i12arrae	1	7.0	true	false	-1.54553...	0.231952...	2	1	2	5	24	29
i02juprm	1	14.0	true	false	-3.96286...	0.238969...	4	4	0	8	21	29
i12altam2	1	24.0	true	false	-1.75773...	0.270144...	7	4	0	11	18	29
i12molui	1	3.0	true	false	-1.47535...	0.310129...	1	3	0	4	25	29
i22marom2	1	7.0	true	false	-4.0	0.240084...	2	4	0	6	23	29
i12dofej2	1	3.0	true	false	-1.47535...	0.310129...	1	3	0	4	25	29
fabra	1	7.0	true	false	-1.00374...	0.310129...	2	2	0	4	25	29
i12rocom	1	7.0	true	false	-4.0	0.240084...	2	4	0	6	23	29
an_userl...	1	24.0	true	false	-1.80145...	0.317542...	7	3	0	10	19	29

Figura 9: Pantallas de información estadística de un test.

5. Evaluación y Resultados

Para probar la herramienta Test Editor se ha utilizado en un curso AHA! como complemento de dos cursos de extensión universitaria sobre programación básica en el lenguaje Java [Romero et al. 02]. Además, se ha realizado una prueba sobre el efecto que tiene en los test adaptativos la especificación de los parámetros TRI por parte de un experto en la materia, en lugar del modo tradicional de tener que esperar a disponer de datos de usuarios suficientes para poder calibrar a partir de datos de utilización de un test clásico informatizado o de tipo papel y lápiz. Esta idea parte de la problemática de que para poder hacer la calibración se necesita un número muy alto de usuarios que hayan realizado el test. Nuestra propuesta va a consistir en utilizar desde el principio un test adaptativo aunque inicialmente los parámetros TRI son los puestos manualmente por un experto [Swaminathan et al. 03], y posteriormente pueda irse calibrando incrementalmente conforme más usuarios lo vayan ejecutando, de forma que los parámetros TRI se van ajustando mejor paulatinamente mientras más usuarios lo van ejecutando.

Para probar esta idea, se ha construido un test adaptativo formado por 27 items sobre programación básica en Java [Romero et al. 02]. Todas las preguntas tienen cuatro respuestas, con sólo una única respuesta correcta. Se ha utilizado un modelo de respuesta al ítem de 2 parámetros (dificultad y discriminación), y los valores de cada item han sido puestos por consenso entre los dos autores del curso y expertos en Java. Los criterios de parada han sido un número máximo de 27 preguntas y/o un error estándar menor de 0.33. El test se ha utilizado por dos grupos de 60 alumnos de Ingeniería Técnica en Informática de la Universidad de Córdoba. Los dos cursos son exactamente el mismo curso de tipo presencial y con una duración de 30 horas, pero

distanciados varios meses en el tiempo, a los que vamos a denominar Curso A, que es el primero en el tiempo y que contiene el Test A con los parámetros TRI puestos directamente por los expertos, y el Curso B, que es el segundo en el tiempo y que contiene el Test B con los parámetros TRI calibrados a partir de los datos de utilización del Test A. En la Tabla 1 se muestra una comparativa de los resultados obtenidos por los alumnos al ejecutar el Test A y el Test B con respecto a los valores medios y desviación típica del tiempo total de realización del test en segundos, el número total de ítems utilizados por el test, el nivel de habilidad final estimado para el alumno y el error estándar en el nivel de habilidad estimado.

	Tiempo realización	**Número de Ítems**	**Nivel Habilidad**	**Error Estándar**
Test A	434.6±88.8	26.9±1.6	-1.3±0.3	0.6±0.1
Test B	182.4±81.2	11.5±2.6	-2.2±0.3	0.4±0.1

Tabla 1: Comparación de resultados de los alumnos en las ejecuciones de los dos test.

En la tabla 1 se puede apreciar que el tiempo de realización del Test B en media es bastante menor que el Test A, y que el número de ítems utilizados en el Test B es también bastante menor que en el Test A. Estos valores nos indican que el funcionamiento del test adaptativo mejora, indicando que requiere menor número de preguntas y por tanto un menor tiempo de realización, que son las características principales de un test adaptativo. Por otro lado, se puede apreciar que el nivel de habilidad de los alumnos ha descendido en el Test B y también el error estándar ha disminuido en el Test B. Esto indica que, la precisión obtenida en el Test B ha sido mayor, y por lo tanto el nivel de los alumnos obtenido es más exacto en el Test B que en el Test A. De forma general se puede afirmar que los resultados obtenidos con el Test B o test calibrado son mejores que los obtenidos con el Test A como podría ser de esperar. Pero la gran ventaja de nuestra propuesta, es que se ha podido utilizar un Test Adaptativo sin tener que calibrarlo inicialmente y tras un número de ejecuciones no muy elevada ya se ha podido calibrar y se han obtenido mejoras notables en su rendimiento.

Con respecto a la evaluación de los alumnos, los resultados obtenidos no se han utilizado posteriormente en la nota del curso, y sólo han servido para dar de manera global una idea sobre el posible nivel de conocimiento de los alumnos en este tipo de curso. Para poder utilizar con fiabilidad los datos proporcionados por un test adaptativo se necesitarían una calibración con un mayor número de usuarios de forma que los parámetros TRI estén mejor ajustados y el error estándar sea menor. Actualmente queremos volver a ejecutar el test con más grupos de usuarios de forma que pueda volver a calibrar los parámetros adaptativos de los ítems de forma iterativa y comprobar como se van obteniendo unos mejores resultados conforme aumentan el número de usuarios.

6. Conclusiones

En este artículo se ha presentado una herramienta autor denominada Test Editor para el sistema AHA! que cubre las necesidades de gestión, tanto de tests clásicos como adaptativos, con una especial atención a estos últimos, para los que se han diseñado operaciones de mantenimiento específicas. La aplicación permite gestionar todas las fases del ciclo de vida de un test, desde la creación hasta el mantenimiento, actualización y borrado. Además se ha logrado gestionar de forma independiente la información relativa a los bancos de ítems y a las baterías de test, por lo tanto reutilizar sin provocar redundancia de información. Además dispone de un módulo de evaluación que permite evaluar a los usuarios de los cursos incluidos en el sistema AHA!, mediante la realización de las baterías de test clásicas y adaptativas previamente creadas con la herramienta Test Editor. El interfaz de la aplicación es intuitivo y fácil de manejar, además de ser capaz de dar servicio a múltiples usuarios al mismo tiempo, sin que se produzca una perdida considerable de rendimiento. Se han realizado pruebas para comprobar el correcto funcionamiento de la herramienta, en concreto se ha utilizado para la construcción de un test adaptativo sobre el lenguaje de programación Java que posteriormente han utilizado dos grupos de 60 personas. Además se ha probado la posibilidad de establecer los parámetros TRI de forma manual por expertos en la materia, posibilitando la utilización de un test adaptativo desde el primer momento, sin la

necesidad de tener que partir de un test clásico y tener que esperar a que sea ejecutado por un gran número de usuarios. Nuestra propuesta consiste en ir realizando posteriormente una calibración iterativa, de forma que el test se vuelve a calibrar cada vez que un número determinado de usuarios lo ejecuten. Aunque los resultados obtenidos son prometedores, se necesitaría probar con un mayor número de usuarios, para demostrar la mejora incremental, además de comparar los resultados con los obtenidos por un test clásico.

Cómo futuras mejoras se van a realizar más ejecuciones con más grupos de alumnos, para comprobar que la calibración incremental permite ir mejorando los resultados obtenidos, además de utilizar un mayor número de items, para poder realizar evaluaciones más fiables. También se va a ampliar el rango de posibles topologías [Brusilovsky 99] que puede tener las preguntas o ítems, de forma que se puedan crear preguntas de tipo relaciona, de tipo puzzle, de tipo rellanar huecos, etc. También se desea ampliar el número de parámetros posibles a utilizar como modelo de TRI contemplando por ejemplo un 3º parámetro denominado adivinanza que representa la posibilidad de acertar por azar, o un 4º parámetro denominado descuido [Millán et al. 02] que representa la posibilidad de que un alumno que conoce el concepto evaluado pueda tener un fallo y responder de forma errónea debido a un descuido por un mal día o cualquier otra causa.

Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2005-08386-C05-02 y los fondos FEDER.

Referencias bibliográficas

[Backer 92] F. Backer. "Item Response Theory, Parameter Estimation Techniques". Marcel Dekker. 1992.

[Brusilovsky 99] P. Brusilovsky, "Web-based Testing for Distance Education" Proceeding of World Conference of WWW and Internet. pp. 149-154. Honolulu. 1999.

[Brusilovsky 03] P. Brusilovsky, "Adaptive Educational Hypermedia" Proceeding of Tenth International PEG Conference. pp. 8-12. 2003.

[De Bra et al. 03] P. De Bra., A. Aerts, B. Berden, B. de Lange, B. Rousseau, T. Santic, D. Smits, N. Stash "!AHA The Adaptive Hypermedia Architecture" Proceedings of the ACM Hypertext Conference. pp.81-84. Notttingham. 2003.

[Millán et al. 02] E. Millán, J.L. Pérez, J.L. "A Bayesian Diagnostic Algorithm for Student Modeling and its Evaluation". User Modeling and User-Adapted Interaction. Vol. 12. No. 1-2. pp 281-330. 2002.

[Halasz et al. 94] F. Halasz, M. Schwartz, "The Dexter Hypertext Reference Model" Communications of the ACM, Vol. 37. No. 2. pp. 30-39. 1994.

[Horton 00] W. Horton, "Designing Web-Based Training" John Wiley&Sons. 2000.

[Olea et al. 03] J. Olea, V. Ponsoda, "Test adaptativos informatizados" Madrid. UNED. 2003.

[Rudner 98] L. Rudner, "An on-line, interactive, computer adaptive testing mini-tutorial" http:// EdRes.org/scripts/cat/catdemo. 1998.

[Romero et al. 02] C. Romero, E. López, C. de Castro "Curso básico de Java 2" Publicaciones de la Universidad de Córdoba. 2002.

[Romero et al. 03] C. Romero, S. Ventura, S. Palomo, P. de Bra., "An authoring tool for web-based adaptive and classic tests". Conference on E-Learning in Corporate, Healthcare & Higher Education. Washington. pp. 174-177. 2003.

[Swaminathan et al. 03] H. Swaminathan, R.K. Hambleton, S.G. Sireci, D. Xing, S.M. Rizavi "Small sample estimation in dichotomous item response models: effect of priors based on judgmental information on the accuracy of item parameter estimates" Applied Psychological Measurement. Vol. 27. No. 1. pp. 27-31. 2003.

[Van der Linde et al. 97] W. Van der Linde, R. Hambleton "Handbook of Modern Item Response Theory" Springer Verlag. 1997.

[Wainier 90] H. Wanier "Computerized Adaptive Testing: a Primer" Lawrence Erlbaum Associates. 1990.