EOL: Generación de exámenes online autocorregidos

Dolores Nogueras, Antonio Gómez

Uno de los momentos que nos causan más pereza a los profesores y maestros suele ser el momento de corregir los exámenes que hemos pasado a nuestros alumnos. Sería muy interesante disponer de alguna herramienta web que, a base de formularios, recogiera las respuestas de los niños, las corrigiera de modo automático y notificara los resultados al profesor.

es@lpmagazine.org

La mayoría de los profesores que estén leyendo este artículo se sentirán familiarizados con esta situación, sobre todo en las fechas en que se están escribiendo estas líneas: a final de curso, todo son prisas por ajustar notas, realizar pruebas de recuperación, calibrar los resultados de las pruebas escritas, teóricas y prácticas, considerar el grado en que estas estadísticas reflejan la realidad de la evolución del alumno con respecto a los objetivos educativos fijados... No sólo ahora, sino en cualquier momento del año, es muy común sentir un poco de agobio ante una pila de exámenes que pueden superar decenas, incluso centenas, de páginas escritas, y que obligatoriamente deben ser leídas, sopesadas y evaluadas por el docente. Sume el lector a estos elementos la ilegible caligrafía de muchos niños, la, digámoslo amablemente, "libre utilización de la gramática y la sintaxis" en algunas expresiones que denotan más esfuerzo memorístico que auténtica comprensión de los chicos de los conceptos por los que se les preguntan, o el desorden en la contestación de las preguntas que suele ser común en los alumnos mayores.

Los autores de este artículo, durante los últimos dos años, hemos estado estudiando con interés las posibilidades que ofrece la web 2.0, que permite la interactividad del usuario con el sitio web que visita, a la hora de diseñar una aplicación que construya formularios a gusto del usuario. Si podemos crear un formulario con un número de preguntas *npreguntas*, que ofrezca un número de opciones por pregunta *nopciones*, al tiempo que podemos especificar qué opción u opciones son correctas en cada caso, y dicho formulario registrara las respuestas de los visitantes, adelantaríamos mucho a la hora de automatizar tan tediosa tarea como la que está centrando el tema de nuestro artículo. Por ello, después de una pequeña consideración teórica acerca de la naturaleza y práctica de la evaluación en educación, analizaremos el potencial de los lenguajes XHTML, PHP y SQL a la hora de diseñar un formulario que permita crear nuevos formularios y registre toda la información en tablas de una base de datos en el servidor web que aloje nuestra aplicación. Actualmente, los resultados de nuestro trabajo están disponibles para el público bajo licencia GNU/UNIX, con el nombre de EOL (Exámenes On Line; nos gusta ser originales...).

¿Por qué hacemos exámenes?

Podríamos contestar, simplemente: "Porque sí. Desde que el mundo es mundo se ha comprobado si una persona tiene suficientes conocimientos en torno a cualquier área mediante una prueba o examen". Pero en un artículo sobre Linux y educación, parecería inapropiado no proporcionar, siquiera un poco a vuelapluma, una justificación pedagógica.

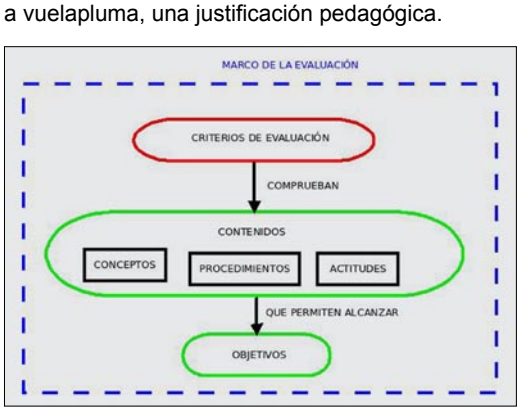

Figura 1. La evaluación nos permite calibrar el alcance de una o varias competencias en base al dominio de un contenido relacionado

Así pues, introduzcamos al lector en algunos conceptos. Para empezar, en la mayoría de los sistemas educativos, se formulan una serie de *OBJETIVOS* (en España se habla los últimos años de *COMPETENCIAS*) que se estima el alumno debe alcanzar. Los *OBJETIVOS GENERALES* (autonomía personal, afectiva, social, conocimiento de la propia sociedad, etc, etc...) se concretan en una serie de *OBJETIVOS ESPECÍFICOS*, que se formulan para cada materia, área o asignatura que el niño o discente debe superar a lo largo de su paso por la escuela o instituto.

Para alcanzar estos *OBJETIVOS*, en la programación de cada materia se establecen una serie de *CONTENIDOS* que el profesor va desarrollando y presentando durante la actividad educativa. Estos *CONTENIDOS* pueden ser *CONCEPTUALES* (teoría), *PROCEDIMENTALES* (práctica) y *ACTITUDINALES*. Si bien esta consideración ha evolucionado entre los teóricos de la pedagogía en los últimos años, y esta disertación puede resultar inadecuada en algunos círculos, a los efectos de introducir al profano en la necesidad de la evaluación, resulta de lo más útil, dado que EOL se orientaría sobre todo a la evaluación de contenidos de tipo conceptual.

En efecto, la evaluación consistiría en calibrar el grado en que el alumno ha alcanzado los *OBJETIVOS* propuestos a través del trabajo de los distintos *CONTENIDOS*. Para ello, se establecen una serie de *CRITERIOS DE EVALUACIÓN* que establecerían una serie de capacidades o procesos que el alumno ha demostrado durante la prueba, y que se corresponderían con los *OBJETIVOS/COMPETENCIAS* que se indican como referencia (véase Figura 1).

Todo lo expuesto supone una concreción algo burda, y en algunos aspectos, diríase que incluso forzada. Sin embargo, desde una perspectiva generalista, que abarca la gran mayoría de sistemas educativos modernos, nos resulta francamente útil.

¿Qué necesitamos para empezar a trabajar?

Como siempre que se habla de lenguajes PHP y SQL, está claro que necesitamos un servidor web. Lo mejor sería contar con un servidor propio, dedicado, equipado con APACHE. Naturalmente, es igualmente válido cualquier espacio web gratuito de carácter publicitario de los que pueblan la red. Los requisitos que hay que cumplir para correr una aplicación de este tipo se cumplen de manera casi universal: tener instalado el lenguaje PHP (mejor 5.0 que 4.0), una base de datos basada en lenguaje SQL, preferentemente acompañada de una interfaz que permita realizar consultas de manera directa por parte del administrador, como PHPMYADMIN.

Naturalmente, y tratándose de nuestra publicación, animamos al lector a que monte su propio servidor dedicado, puesto que cualquier sistema operativo basado en Linux dispone de sobradas herramientas para ello. De hecho, publicamos un artículo al efecto en los números de enero y febrero de este mismo año de Linux+.

En este segundo caso, no habría más que abrir la consola, identificarse como root, y proceder a la instalación mediante la herramienta *aptitude*:

```
# aptitude install mysql-server php5 libapache2-
mod-php5 php5-gd php5-dom php5-pgsql php5-mysql
phpmyadmin
```

En un servidor web propio dedicado, por norma general, cada sitio web (como es el caso de las aplicaciones) se instalará en el directorio */var/www/.*

Dependiendo del grado de conocimientos técnicos del usuario, se puede optar por construir su propia aplicación a partir de nuestra propuesta, o utilizar (modificándola o no) nuestra propuesta, EOL, que está disponible para su libre distribución y modificación en *http://www.informesevaluacion.com*, en la sección Descargas, o para testear una beta, en la sección SIMULADOR ONLINE.

Terminemos este apartado recordando que las aplicaciones PHP+SQL precisan que se haya definido una base de datos en el servidor, así como un usuario (con su correspondiente contraseña de acceso) con privilegios de lectura y escritura en dicha base de datos. Para más información, remitimos de nuevo al respetado lector a los números de enero y febrero de esta misma publicación.

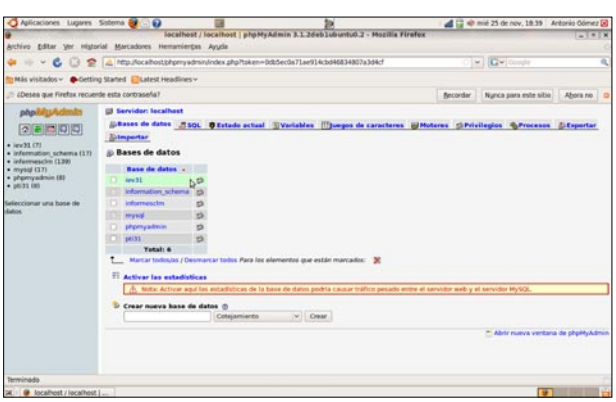

Figura 2. Un interfaz web como PHPMYADMIN, sin ser imprescindible, facilita mucho las tareas de administración y mantenimiento de las bases de datos que utilizan nuestras aplicaciones web

¿Cómo funciona una aplicación XHTML+PHP+SQL?

Desde la perspectiva del objetivo que deseamos conseguir (un sitio web que interactúe con el usuario, intercambiando información que irá registrando en una base de datos), el

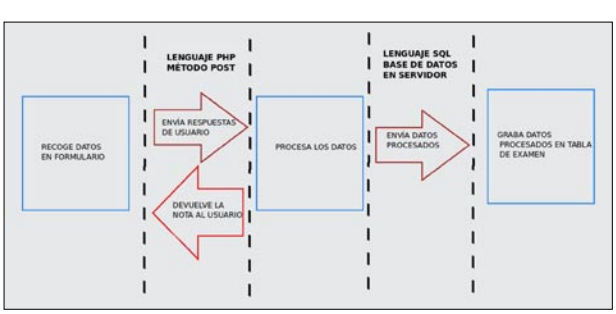

Figura 3. Este diagrama recoge de manera escueta el proceso de trabajo de la aplicación, en su parte XHTML, PHP y SQL

Listado 1. El archivo grabaconfig.php, en el directorio de instalación, utiliza los comandos fopen, fputs y fclose para crear un archivo en la raíz denominado config.php, que contendrá los parámetros de acceso a la base de datos. A continuación, pasa a crear las tablas correspondientes

```
$abro_fichero = fopen('../config.php','w');
$servidor=$_POST['servidor'];
$base=$_POST['base'];
$usuario=$_POST['usuario'];
$contra=$_POST['contra'];
$web=$_POST['web'];
         $salto = "\n";
         $linea_1 = '<?php';
         fputs($abro_fichero,$linea_1);
         fputs($abro_fichero,$salto);
         $linea_2 = '$servidor = \''.$servidor.'\';';
         fputs($abro_fichero,$linea_2);
         fputs($abro_fichero,$salto);
$linea_3 = '$base = \''.$base.'\';';
         fputs($abro_fichero,$linea_3);
         fputs($abro_fichero,$salto);
$linea_4 = '$usuario = \''.$usuario.'\';';
         fputs($abro_fichero,$linea_4);
         fputs($abro_fichero,$salto);
         $linea_5 = '$contra = \''.$contra.'\';';
         fputs($abro_fichero,$linea_5);
         fputs($abro_fichero,$salto);
         $linea_6 = '$web = \''.$web.'\';';
         fputs($abro_fichero,$linea_6);
         fputs($abro_fichero,$salto);
         $linea_8 = '?>';
         fputs($abro_fichero,$linea_8);
                  fputs($abro_fichero,$salto);
         fclose($abro_fichero);
         echo "<meta http-equiv=\"refresh\" content=\"0;URL=creatablas.php\">";
         ?>
```

Listado 2. Función en PHP de conexión a la base de datos

```
<?php
function Conectarse()
{require('config.php');
//el archivo config.php contiene los valores de las variables $servidor, $usuario y $contra
//y se graba durante la instalación de la aplicación
   if (!($link=mysql_connect($servidor,$usuario,$contra)))
   {echo "Error conectando a la base de datos.";
      exit(); }
   if (!mysql_select_db($base,$link))
   { echo "Error seleccionando la base de datos.";
      exit(); }
   return $link;
}
$link=Conectarse();
mysql_close($link); //cierra la conexion
?>
```

programador debería dividir su trabajo enfocándolo de tres maneras:

- La aplicación web deberá presentar un formulario al usuario que le pedirá una serie de datos (nombres, contraseñas, contenidos de preguntas, contenidos de opciones, etc.) en base al lenguaje XHTML de generación de hipertextos.
- Dichos datos serán pasados, por el método POST, a una aplicación en PHP que recogerá dichos datos y los procesará. En el caso de un examen online, irá sumando todas las respuestas correctas y ofrecerá el resultado al alumno una vez haya terminado el examen.
- Acto seguido, contrastará dicha información con el contenido de la base de datos, realizando las pertinentes operaciones de grabación, borrado y/o modificación en

Listado 3. Este código forma parte de la página en XHTML que realiza la consulta a la tabla listaexamenes de la base de datos para permitir al profesor seleccionar el examen cuyas calificaciones desea obtener

```
<form action="<?php echo $direccionweb.$nombrexamen" method="post" name="claves">
<?php
require("conectarse.php");
   $link=Conectarse();
   $result = mysql_query("SELECT * FROM listaexamenes order by titulo",$link);
while($row = mysql_fetch_array($result)) {
$direccion=$row['nombrexamen'];
?>
  Examen:
  <label for="select"></label>
//La función enviar() está en lenguaje javascript y ha sido previamente definida en la cabecera
//de la página en la que está alojado el código.
  <select name="nombrexamen" id="examen" onChange="enviar()">
  <option value=""> </option>
        <option value="<?php echo $row['nombrexamen'];?>"><?php echo $row['titulo']; ?></option>
  <?php
  }
  mysql_free_result();
?>
  </select>
</form>
```

Listado 4. Ejemplo de formulario en examen estándar (muestra.php)

```
//este formulario se alojaría en un examen denominado muestra.php, y enviaría
//los datos al archivo notamuestra.php
<form id="examen" name="examen" method="post" action="notamuestra.php">
  <p>Nombre: <input name="nombre" type="text" id="nombre" /> </p>
<p>Pregunta n°1: Señala la opción que NO define un tipo de malware</p>
//la respuesta correcta tiene un valor igual a 10/npreguntas
<select name="p1">
<option value="5">Jogging</option>
<option value="0">Hijacking</option>
<option value="0">Stealer</option></select></p>
<p>Pregunta n°2: ¿En qué consiste el PHISING?</p>
<select name="p2">
<option value="0">Es un tipo de muestreo de datos en una base de tipo Access</option>
<option value="5">Es una ciberestafa basada en el concepto de INGENIERÍA SOCIAL</option>
<option value="0">Consiste en programar mediante XHTML+AJAX</option></select></p>
    <input type="submit" name="Submit" value="Enviar">
</form>
```

Listado 5. El formulario muestra.php envía los datos a notamuestra.php, que suma los resultados, los graba en la tabla muestra y notifica la calificación al usuario

```
<?php
   include("conectarse.php");
  $link=Conectarse();
   $nombre=$_POST['nombre'];$nota=0;
$p1=$_POST['p1'];
$nota=$nota+$p1;
$p2=$_POST['p2'];
$nota=$nota+$p2;
mysql_query("insert into muestra(nombre,p1,p2,nota) values('$nombre','$p1','$p2','$nota')",$link);echo "Tu
nota en este examen ha sido:".$nota;
if ($nota<5){echo "¡Prueba de nuevo!";}
if ($nota>=5){echo "¡Felicidades";}?>
```

Listado 6. Código orientado a contrastar el nombre y contraseña proporcionados con los contenidos en la tabla claves

```
 <?php
   include("conectarse.php");
   $link=Conectarse();
   $nombre=md5($_POST['nombre']);
   $contra=md5($_POST['contra']);
        $result = mysql_query("SELECT id FROM claves WHERE nombre='$nombre' AND contra='$contra'");
$row = mysql_fetch_array($result);
if ((!isset($row[0]))) {
echo "El Usuario con Nombre <B>".$nombre."</B> no está registrado en nuestra base de datos o no ha
introducido adecuadamente su clave."; mysql_close();
?php>
```

Listado 7. Este código forma parte del archivo creaformulario.php, y se encarga de registrar las variables enviadas por el profesor en la base de datos SQL

```
<?php
$tablaexamen="CREATE TABLE $nombrexamen(
`id` int( 2 ) NOT NULL auto_increment,`nombre` TEXT NOT NULL,
PRIMARY KEY ( `id` )
) ";
mysql_query($tablaexamen,$link)or die ("no funciono...");
$p=1;
while($p<=$npreguntas){
$pregunta="p".$p;
$anyade="ALTER TABLE $nombrexamen ADD COLUMN $pregunta INT( 2 ) NOT NULL" ;
mysql_query($anyade,$link) or die ("no se anyadir");
$p++;}
$anyade2="ALTER TABLE $nombrexamen ADD COLUMN nota INT( 2 ) NOT NULL" ;
$anyadexamen="ALTER TABLE listaexamenes ADD COLUMN $nombrexamen TEXT NOT NULL" ;
mysql_query($anyade2,$link) or die (mysql_error());
  mysql_query("insert into listaexamenes (nombrexamen,titulo,numeropreguntas) values ('$nombrexamen','$titulo
','$npreguntas')",$link) or die (mysql_error());
?>
```

dichas tablas. En nuestro caso, consignará el nombre del alumno, la relación de respuestas realizadas, y la nota obtenida, en una tabla con el nombre del examen que el profesor haya consignado.

Instalación de EOL

Cuando se instala EOL en un servidor web, la aplicación busca automáticamente si existe el archivo *config.php*, que contendrá los datos de conexión a la base SQL. En caso de que no exista dicho archivo, el programa asume que es la primera vez que se conecta, y procede a remitir al navegador a la subaplicación de instalación, en la que tendremos que consignar los datos de conexión (dirección web del servidor SQL, nombre de usuario SQL con privilegios, contraseña SQL, y nombre y password que se requerirán al profesor cuando desee crear nuevos exámenes).

Al enviar estos datos por el método post, la aplicación comprobará si realmente puede conectarse a dicha base con los datos que el administrador web le ha facilitado (véase *Listado 1)*; de ser así, procederá a crear el archivo *config.php*, que contendrá las variables *$servidor*, *$usuario* y *$contra* que precisa la función en PHP *conectarse()* (se explica más adelante), incluida en todos los archivos que precisen en algún momento acceder a la base.

Características de la base de datos

En el directorio de instalación, de la subaplicación *grabaconfig.php* se pasa a la aplicación *creatablas.php*, que realiza una serie de consultas SQL orientadas a la creación de las siguientes tablas:

- La tabla *claves*, con los campos *id* (con autoincremento, de cara a futuros indexados), *nombre*, y *contra*. Estos dos últimos campos se introducirán previamente encriptados por el algoritmo md5.
- La tabla *listaexamenes*, con los campos *id* (siempre con autoincremento, por la misma razón), *nombrexamen* (referido al nombre del formulario que irá seguido de la extensión ".php", *titulo* (nombre del examen a presentar en pantalla) y *numeropreguntas*. Mediante consulta a esta tabla, EOL puede presentar al usuario (profesor o alumno) la relación de exámenes disponibles.

Cada vez que un profesor, debidamente identificado, quiera generar un examen nuevo, se creará una tabla con el nombre de la variable anteriormente referida, *nombrexamen*. Dicha tabla contará con los campos *id* que permita el autoindexado, un campo de texto *nombre* donde se consignará el nombre de cada alumno que pase el examen, un campo por cada pregunta (*p1*,*p2*, *p3* y así sucesivamente) con un valor 1 o 0 según el alumno haya acertado o no la pregunta, y el campo *nota* que, lógicamente, recogerá la calificación del alumnado (el valor a sumar por cada campo *px* acertado se hallará dividiendo el máximo, 10, por la variable *npreguntas*).

Como se verá en posteriores apartados de este mismo artículo, por cada examen generado, además de la correspondiente tabla en la base de datos, se generará el correspondiente formulario web, con el nombre de la variable *nombrexamen* seguido de la extensión .php, y el archivo "nota"+*nombrexamen*+".php", que recogerá los datos enviados por el formulario y los grabará en dicha tabla.

Organizando el trabajo. División en subtareas de programación

Ya hemos resaltado en otros artículos que no somos programadores expertos, a pesar del interés que tenemos con todo lo relacionado con Linux, GNU y el software libre en general. De hecho, somos conscientes de la actitud escandalizada con la que compañeros con auténticos conocimientos de programación que lean estas líneas analizarán unos diagramas de flujo bastante alejados de la ortodoxia, o recorrerán unos listados de programación con múltiples fallos y reiteraciones innecesarias. Discúlpesenos y permítanos el amable lector oponer a nuestras fallas la voluntad e interés con que afrontamos cada uno de estos nuevos retos. Siempre estamos dispuestos a seguir aprendiendo y a admitir y aprovechar cualquier tipo de crítica constructiva que se nos quiera presentar.

De todos modos, una cosa sí sabemos con seguridad. A la hora de realizar una tarea relativa (o absolutamente) compleja, es imprescindible un mínimo de planificación, que permita modularizar el trabajo y simplificarlo en subtareas algo más simples. En la Figura 5 recogemos lo que creemos que debería ser el funcionamiento general de nuestra aplicación de generación y presentación de exámenes online.

De momento, está claro que necesitamos que la aplicación distinga si el usuario es alumno o profesor. En este segundo caso, entendemos que debería probar su identidad mediante un *password* de usuario, puesto que va a acceder a información sensible relativa a los logros de su alumnado.

En el caso de la atención al alumno, a su vez, necesitamos distinguir dos líneas de programación: la que derive en

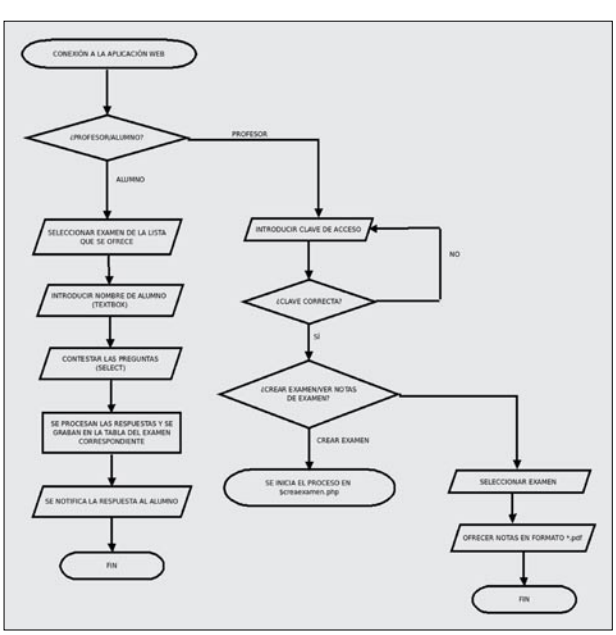

Figura 4. Este diagrama refleja el funcionamiento de cualquier examen que generemos: se creará una tabla que recoja las respuestas del alumnado, un formulario en PHP que presente las preguntas y recoja los datos y otra aplicación que procese dichos datos y los envía a la base SQL

el formulario *per se* que sería el examen a pasar como tal, y la que procese la información que introduzca el alumno, le devuelva la nota y la lleve a la correspondiente tabla de la base de datos.

Volviendo al trabajo con el profesor, también hay que distinguir dos grandes líneas de programación: la que le permita crear nuevos exámenes, especificando número de preguntas, número de opciones por pregunta, señalar la respuesta correcta en cada caso, etc..., y la orientada a permitir al docente la consulta de notas de alumnos que ya han pasado exámenes anteriormente generados.

Conexión a la base de datos

El primer escollo a saltar es asegurar que nuestra aplicación tiene acceso a la base de datos SQL en todo momento. Para ello, crearemos un archivo en PHP de nombre *conectarse.php*, en el que definiremos una función *conectarse()*, como la del *Listado 2*. Así, en cada parte de la aplicación que requiera la interacción con dicha base de datos, nos limitaremos a hacer una llamada a dicha función mediante el comando *include ("conectarse.php");*

Selección de un examen en concreto de entre todos los disponibles

Cuando el profesor desee acceder a la lista de calificaciones en un examen en particular, o el alumno se disponga a pasar su prueba y tenga que indicar cuál es la prueba que le corresponde, un nuevo formulario realizará una consulta a la tabla *listaexamenes* para ofrecer la relación de exámenes que ya se pueden consultar (véase el Listado 3).

Realización de un examen. Rellenado del formulario y autocorrección por parte de la aplicación

Como se ve en la Figura 3, el proceso de trabajo de cualquiera de los test generados por el profesorado sigue una línea de flujo muy definida: un formulario en XHTML recoge la opción seleccionada para cada pregunta, que ofrece el conjunto de respuestas en un control de tipo *select*. (Véase el Listado 4).

Dicho formulario envía dichas respuestas a otra página que las procesa, devuelve la calificación al usuario (Listado 5) y graba en la correspondiente tabla la información correspondiente.

Solicitud de identificación mediante contraseña

Durante la instalación de la aplicación, una de las tablas que se genera es *claves*, con los campos *id* (con autoincremento) que permita indexar a los usuarios, *nombre* y *contra,* ambos procesados previamente con la función de encriptación md5.

En el momento en que un profesor necesita identificarse para acceder a la aplicación de creación de nuevos exámenes o a los resultados de pruebas anteriores, se le presentará un formulario con dos *textbox*, de nombre *nombre* y *contra*, precisamente, que lo envía por el método POST a otro archivo en PHP, denominado *procesaclaves.php*. El *Listado 6* recogería la forma en que EOL recoge y compara dichos datos, previamente encriptados, con los que tiene en la tabla *claves*, otorgando o denegando el acceso de acuerdo a la veracidad de esos datos.

Creación de un examen (I). Modificación de la base SQL

Una vez el profesor se ha identificado correctamente, se le deben dar dos opciones: acceder a las calificaciones de exámenes ya generados, o crear una nueva prueba (siempre tipo test, si queremos que sea un examen autocorregido).

Si la elección es crear una nueva prueba, el primer paso será recoger los datos en un formulario. Dichos datos serán las variables ya mencionadas *$nombrexamen, $titulo, $npreguntas* y *$nopciones.*

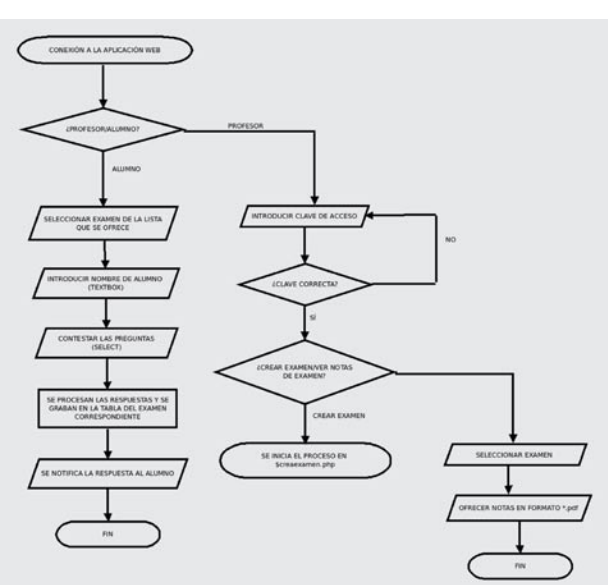

Figura 5. Un diagrama de flujo nos permite abordar una ingente tarea de programación de un modo más simplificado, al poder dividir el trabajo en varias subtareas

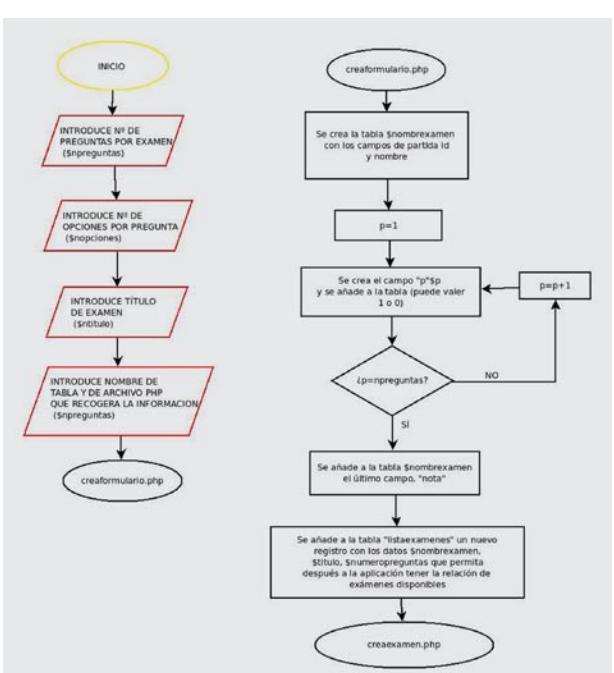

Figura 6. Un diagrama de flujo nos permite abordar una ingente tarea de programación de un modo más simplificado, al poder dividir el trabajo en varias subtareas

Listado 8. El formulario que se presenta al profesor debe mostrar tantas preguntas y número de opciones por pregunta como se haya especificado anteriormente

```
<?php $r=1;
while ($r<=$npreguntas){
$selecciona="r".$r;
echo "Pregunta nº: ".$r;
echo '<input name="p'.$r.'" type="text" size="100%" /><br>';
$o=1;
echo"<blockquote>";
while($o<=$nopciones){
$opcion="o".$o;
echo "<p>Opcion nº:".$o.'<input name="o'.$r.$o.'" type="text" size="60%" /><input name="v'.$r.$o.'"
type="checkbox" value="'.$opcion.'" /><br>';
$o++;
}
echo"</blockquote>";
$r++;
}
?>
```

Listado 9a. Éste es quizá el listado más importante de todos, pues crea dos archivos conforme a las preguntas y respuestas que el profesor desea

```
<?php
/*Todos los listados que precisan interacción con la base de datos SQL deberán hacer un include a los archivos
conectarse.php y config.php*/
   include("conectarse.php");
   include ("config.php");
   $link=Conectarse();
//Recogemos las variables del formulario anterior
$titulo=$_POST['titulo'];
$nombrexamen=$_POST['nombrexamen'];
$npreguntas=$_POST['npreguntas'];
$nopciones=$_POST['nopciones'];
$valorpregunta=10/$npreguntas;
//Creamos un archivo vacío para alojar nuestro formulario de examen
$archivo = fopen($nombrexamen.".php", "w");
$contenidopaginainicio='<html>
<head>
<title>Documento sin t&iacute;tulo</title>
<meta http-equiv='.'"Content-Type"'.'"content="'.'text/html; charset=iso-8859-1"'.'>
</head>
<body>
<form id="examen" name="examen" method="post" action="nota'.$nombrexamen.'.php">
  <p>Nombre:
    <input name="nombre" type="text" id="nombre" />
</p>
';
//inicializamos las variables $pregunta y $cuerpoexamen
$pregunta=1;
$cuerpoexamen="";
//Inicializamos un bucle para ir enriqueciendo el formulario con los contenidos de las //preguntas y opciones
enviados por el anterior formulario
while($pregunta<=$npreguntas){
$opcion=1;
$cuerpopciones="";
$texto=$_POST['p'.$pregunta];
$cuerpoexamen=$cuerpoexamen.'Pregunta nº'.$pregunta.': '.$texto.'</p>
<select name="p'.$pregunta.'">';
```

Listado 9b. Éste es quizá el listado más importante de todos, pues crea dos archivos conforme a las preguntas y respuestas que el profesor desea

```
while ($opcion<=$nopciones){
$textopcion=$_POST['o'.$pregunta.$opcion];
$valoropcion=$_POST['v'.$pregunta.$opcion];
if (isset ($valoropcion)){
$cuerpopciones=$cuerpopciones.'<option value="'.$valorpregunta.'">'.$textopcion.'</option>';
}
else {
$cuerpopciones=$cuerpopciones.'<option value="0">'.$textopcion.'</option>';
}
$opcion++;
}
$cuerpoexamen=$cuerpoexamen.$cuerpopciones.'</select></p>';
$pregunta++;
}
//Cuando terminamos con el formulario en sí, cerramos el formulario y la página conforme al  //estándar XHTML
$contenidopaginafinal='<p>
    <input type="submit" name="Submit" value="Enviar">
</p></form></body>
</html>';
// Se procede a continuación a insertar el texto XHTML generado en el archivo $archivo
fwrite($archivo, $contenidopaginainicio.$cuerpoexamen.$contenidopaginafinal);
fclose($archivo);
//De manera parecida actuamos para crear el archivo de procesado de datos
$archivocontador = fopen("nota".$nombrexamen.".php", "w");
$contenidocontadorinicio='<?php
   include("conectarse.php");
  $link=Conectarse();
   $nombre=$_POST['."'nombre'".'];';
   $insertandopreguntasentabla='$nota=0;';
   $i=1;
//Iniciamos un bucle para crear una variable de tipo $_POST por cada respuesta en el //formulario
   while($i<=$npreguntas){
   $insertandopreguntasentabla=$insertandopreguntasentabla.'$p'.$i.'=$_POST['."'p".$i."'".'];$nota=$nota+$p'.$i
.';';
   $i++;
   };
$insertandopreguntasentabla=$insertandopreguntasentabla.'mysql_query("insert into '.$nombrexamen.'(nombre,';
$ii=1;
while($ii<=$npreguntas){
$insertandopreguntasentabla=$insertandopreguntasentabla."p".$ii.",";
$ii++;
};
$insertandopreguntasentabla=$insertandopreguntasentabla."nota) values(."'".'$nombre'."'".',';
$iii=1;
while($iii<=$npreguntas){
$insertandopreguntasentabla=$insertandopreguntasentabla."'".'$p'.$iii."',";
$iii++;};
$insertandopreguntasentabla=$insertandopreguntasentabla."'".'$nota'."'".")".'"'.',$link);';
$insertandopreguntasentabla=$insertandopreguntasentabla.'echo "Tu nota en este examen ha sido:".$nota;
if ($nota<5){echo "¡Mala suerte!";}
if ($nota>=5){echo "¡Felicidades";}';
   $contenidocontadorfinal='?>';
   $contadortotal=$contenidocontadorinicio.$insertandopreguntasentabla.$contenidocontadorfinal;
fwrite($archivocontador,$contadortotal);
fclose($archivocontador);
echo "la dirección que debes poner en tu explorador para iniciar el examen será: ".$web."/
".$nombrexamen.".php";
 ?>
```

La siguiente fase consistirá en recoger dichos datos, integrarlos como registros en la tabla *listaexamenes*, y crear a continuación la tabla de nombre el contenido de la variable *nombrexamen* (por ejemplo, "formulacion"), de acuerdo a lo estipulado en anteriores apartados, según se expresa en el Listado 7.

Por último, con esos datos, se autoescribirá un formulario en el que el profesor deberá escribir el contenido de las preguntas que quiere realizar a los niños, con las correspondientes opciones que el niño deberá elegir. Junto a cada opción, un control de tipo *checkbox* indicará si dicha opción se considerará correcta (y, por lo tanto, puntuará en la calificación total) o no.

Creación de un examen (II). Generación de los formularios y archivos de procesado de datos

En una segunda fase dentro de la subtarea de generación de exámenes, necesitaremos crear dos archivos: el formulario que, como tal, realizará las preguntas y recogerá las respuestas (nombrado con el contenido de la variable *nombrexamen*), devolviendo la calificación al alumno, y el archivo en PHP (con el mismo nombre, precedido de la palabra "nota"), que procesará dichas respuestas y las registrará en la tabla con el mismo contenido de *nombrexamen*. En el *Listado 8* se recoge la parte de código que crearía un texto en XHTML, utilizando bucles, correspondiente a la generación del formulario correspondiente al examen (con *npreguntas* cajas de texto, denominadas "*p1*", "*p2*", y así sucesivamente, y *nopciones* por pregunta, denominadas "*o1", "o2*", etc.). Dicho texto se ha guardado en una variable de nombre muy simple, *$r*.

En el Listado 9 se recogen todos los datos de este formulario, y se crean dos archivos en PHP que corresponderían al formulario del examen y la aplicación de recolección, procesado y registro de los datos en la correspondiente tabla.

Consulta de resultados. Uso de la librería FPDF

Seleccionado el examen correspondiente, como puede analizarse en el Listado 10, la aplicación realizará la consulta a la tabla con el nombre especificado, y recogerá los nombres de examinandos y notas obtenidas. Para mejorar la apariencia del documento generado (una impresión de pantalla siempre resulta algo burda), hemos utilizado la librería FPDF, de libre uso, que genera un PDF sobre lenguaje PHP de acuerdo a los parámetros que dicha consulta nos facilite, conforme a una estructura de clase estándar en este tipo de lenguaje.

Nosotros también queremos tener nuestras FAQ

Emulando a auténticos programadores, y dado que la aplicación que aquí mostramos es medianamente operativa, no queremos dejar de instaurar nuestra propia sección de *FAQ, Frecuently Asked Questions* (vamos, las preguntas comunes de toda la vida). Para ello, mostramos nuestro trabajo a algunos colaboradores habituales de esta publicación, para que nos dieran su opinión. Hubo respuestas para todos los gustos, desde señalar posibles fallos en los listados, a plantear cuestiones más filosóficas, pero siempre desde la perspectiva del pensamiento constructivo. Todas estas aportaciones nos sirven para adelantarnos a preguntas que pueden surgir en la mente de nuestro apreciado lector. Sigan leyendo...

¿No resulta muy fácil copiar con esta aplicación?

Indudablemente. Para empezar, basta con analizar el código fuente de la página del examen desde el explorador, para localizar las opciones de cada control tipo *select* que no tiene un *value* igual a cero. Es cierto, aunque no es probable que un alumno de Educación Secundaria tenga formación y picardía para llevar a cabo esta trampa.

Más factible como técnica "chuletera" puede ser mantener, en una ventana minimizada, un navegador que permita al niño, expresándonos en su propia jerga, *rezarle a San Google o Santa Wikipedia Mártir.*

Pero todavía más fácil (y esta situación la conoce cualquier profesor que haya dado clase en un aula de informática) será echar una miradita al monitor del compañero. La distribución de este tipo de aulas no suele incluir obstáculos visuales que aíslen a unos alumnos de otros en cada puesto. Aún más, al menos en la mayoría de los centros públicos, es difícil lograr organizar al grupo en puestos individuales. Lo más normal es que un ordenador sea compartido en clase por dos e incluso por tres personas.

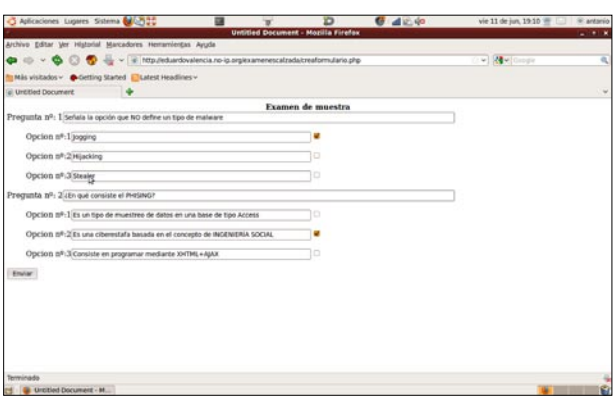

Figura 7. Un formulario autogenerado pedirá al profesor los contenidos de las distintas preguntas y opciones a elegir. El control checkbox indicará la respuesta que se considerará correcta

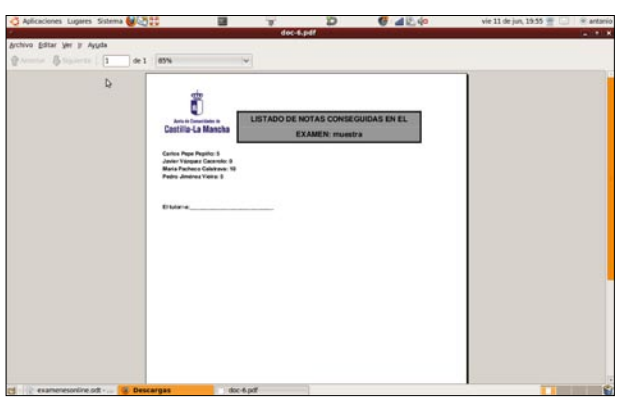

Figura 8. El uso de la librería FPDF permite una presentación algo más agradable y menos sobria que una simple impresión de pantalla

Así que nos remitimos, de nuevo, a dos puntos que hemos establecido al principio del artículo:

- No hay que olvidar que estamos hablando de una herramienta de *apoyo*. En ningún momento hemos pretendido sustituir al conjunto de pruebas de evaluación que tradicionalmente se llevan a cabo en los centros.
- El profesor organiza, coordina y controla cualquier tipo de prueba de evaluación que se desarrolle con el alumnado. No podemos pretender que una máquina desarrolle nuestro cometido.

¿No existen alternativas más recomendables a la forma de encriptado md5?

Muy posiblemente. No estamos protegidos contra ataques como las inyecciones SQL, por ejemplo. El algoritmo md5 sólo proporciona una seguridad relativa para separar contraseñas de profesores de alumnos que casualmente, en un aula de informática, puedan tener acceso al puesto del profesor. Cualquier cracker medianamente formado forzará en pocos minutos nuestro sistema. Un sistema (permítasenos la ironía), que de por sí no es nada seguro cuando depende de un profesorado que muchas veces no toma ninguna medida de

Listado 10. El uso de la clase PDF, de libre uso en la comunidad, se basa en la creación de celdas, al tratar el documento pdf a generar como una tabla, en base a una serie de parámetros de muy sencilla utilización

```
<?php require ('fpdf.php');
$nombrexamen=$_POST['nombrexamen'];
function Conectarse()
{require ('config.php');
   if (!($link=mysql_connect($servidor,$usuario,$contra)))
   {
   echo "Error conectando a la base de datos.";
      exit();
   }
   if (!mysql_select_db($base,$link))
   {
      echo "Error seleccionando la base de datos.";
      exit();
   }
   return $link;
}
$pdf=new PDF;
$link=Conectarse();
Require ("config.php");
$result=mysql_query("select * from $nombrexamen order by nombre",$link);
$encabezamiento='LISTADO DE NOTAS CONSEGUIDAS EN EL EXAMEN: '.$nombrexamen.$numeropreguntas;
$pdf->Addpage();
$pdf->SetFont('Arial','B',14);
$pdf->Image('logoclm.jpg',10,10);
$pdf->Ln(15);
$pdf->SetLineWidth(1);
$pdf->Cell(50,10,'');
$pdf->SetFillColor(150,150,150);
$pdf->Multicell(120,10,$encabezamiento,1,'C',1,1);
$pdf->Ln(5);
while ($row=mysql_fetch_array($result)){
$nombre=$row['nombre'];
$nota=$row['nota'];
$pdf->SetFont('Arial','B',10);
$pdf->Cell(180,5,$nombre.": ".$nota);
$pdf->Ln(5);
}
$pdf->Ln(15);
$pdf->Cell(60,5,'El tutor/-a:_____');
$pdf->Output();
?>
```

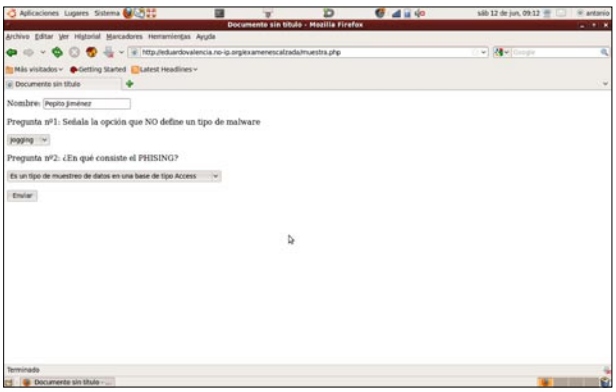

Figura 9. Una vez creado el examen, es muy fácil para los niños conectarse a nuestra web, seleccionar el examen que corresponda, poner su nombre al principio y empezar a contestar

seguridad con el password que se le ha propuesto, que lo deja apuntado en post-its, esquinas de hojas de cuadernos, etc., al alcance de los niños.

Queremos recordar al respetado lector que EOL supone una propuesta con licencia GNU, que tiene como uno de sus principios la libertad de modificar y compartir este tipo de software. Agradeceremos, de hecho, cualquier propuesta de mejora a éste y otros respectos.

¿No puede el alumno repetir muchas veces el examen hasta obtener la nota apetecida?

Sí. De hecho, según nuestra experiencia, suele ser una tentación demasiado fuerte. Hablamos de una generación que ha nacido con el ratón en la mano, y conoce de sobra el significado de la flecha *Ir a la página anterior*. Es casi un acto reflejo. Según la importancia que el profesor quiera darle a la prueba en cuestión, puede permitir o no esa práctica. De todos modos, si se examina el *Listado 10*, puede deducirse el resultado en el informe en pdf que posteriormente generará el profesor. Se realiza un ordenamiento previo alfabético por nombre, en base al cual se presentan los resultados. Al no haber más requisitos de ordenamiento, en el caso del alumno que haya repetido varias veces su examen, aparecerán todos los resultados, *por orden de realización*. Es decir, la nota que vale es la primera que aparece en el informe.

Al igual que en el anterior apartado, remitimos a los programadores con más experiencia al principio del movimiento GNU: *se permite la modificación y posterior distribución de la aplicación mejorada*. Esperamos sus aportaciones.

Conclusiones

Nuestra propuesta, EOL, con licencia GNU/UNIX, cumple con todos los requisitos que nos hemos autoimpuesto: dispone de una rutina de autoinstalación, sistema de identificación del profesorado por contraseña, posibilidad de consulta de calificaciones por medio de documento *pdf*... Dentro de una propuesta más grande para el profesorado, orientada a la generación de encuestas online, informes de evaluación y planes de trabajo individualizados (PTI), está a disposición del internauta interesado en *www.informesevaluacion.com*. Siendo coherentes con los términos de esta licencia de software libre, queremos compartir con la comunidad tanto dichas aplicaciones como el enfoque que hemos adoptado a la hora de programar cada una de las subtareas que integran el conjunto.

Somos muy conscientes de que todo se puede perfeccionar, y de hecho, realmente esperamos las aportaciones de aquellos miembros de la comunidad más expertos a los que hayamos conseguido interesar con nuestro proyecto. Creemos, no obstante, que la actual versión de EOL es ya lo bastante estable y fiable como para satisfacer las expectativas del profesor medio que quiera enriquecer su práctica educativa. Entendiendo, lógicamente, que un solo tipo de prueba, por muy automática que sea, nunca puede sustituir al conjunto de las herramientas que integran un proceso evaluador. Es por eso por lo que aún, gracias a Dios, a profesores y maestros no nos han sustituido por ordenadores.

Sobre los autores

María Dolores Nogueras Atance, licenciada en Ciencias Químicas, es profesora de Tecnologías en la actualidad, pero también ha pasado algunos años como profesora de Formación Profesional en Laboratorio. Su irrupción en el mundo informático ha sido algo tardía, y debido sobre todo a la estrecha relación de dicho mundo con la materia que actualmente imparte. Sin embargo, ha sabido retomar el ritmo y pone a prueba y se esfuerza por aprender toda nueva herramienta informática que caiga en sus manos y que pueda tener algo que ver con la educación.

Antonio Gómez García es Ingeniero Técnico Industrial de Formación, y lleva más de diez años dedicando su actividad profesional a la Educación Secundaria y Bachillerato en institutos. Profesor de Tecnologías y de Tecnologías de la Información, ha trabajado como asesor TIC en el Centro de Profesores de Puertollano, y dedica gran parte de su tiempo al software libre y su introducción en el sistema educativo. Desde esa filosofía, ha colaborado ya en varias actividades de formación de padres, profesores y alumnos sobre seguridad en Internet. En la actualidad, es Responsable de Medios Informáticos en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real). Agradecerá cualquier aporte que queráis realizar en *administrador @eduardovalencia.no-ip.org*

En la red

- Informes de evaluación, PTI y exámenes EOL
  http://www.informesevaluacion.com
- Utilidad de simulación de EOL
  http://eduardovalencia.no-ip.org/informes/creaexamenes
- Movimiento por el software libre GNU
  http://www.gnu.org/home.es.html
- Ejemplo de utilización en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real)
  http://eduardovalencia.no-ip.org/examenescalzada/