Autor	Pedro José Guillén Céspedes
E-mail del autor	pjgc@ono.com
Director	José Fernando Cerdán Cartagena
E-mail del director	fernando.cerdan@upct.es

Título del PFC	Aplicación para la gestión de currículums de investigadores

Resumen

Creación de una aplicación que permita crear, almacenar y modificar currículums de investigadores, permitiendo la presentación de estos en diferentes formatos personalizados.

Titulación	Ingeniería Técnica de Telecomunicación, Especialidad en Telemática
Departamento	Tecnología de la información y las comunicaciones
Fecha de presentación	Julio - 2007

0 - Índice

1 - Descripción y objetivos del proyecto

La idea de este proyecto es la creación de una aplicación que permita crear y gestionar currículums de investigadores. Con este sistema se pretende que el usuario pueda crear, modificar o borrar currículums o parte de ellos de manera sencilla. Además permite trabajar con selecciones específicas, pudiendo ocultar los campos por tipo o por fecha sin necesidad de eliminarlos de la base de datos.

1.1 - Objetivos generales proyecto

1- Crear una aplicación fácil de utilizar.
2- Utilizar herramientas de software libre para su programación y ejecución.
3- Hacer una aplicación compatible con la mayoría de sistemas operativos.
4- Orientar la aplicación para uso personal, pero que esté abierta para crear una versión distribuida en un futuro.

1.2 - Objetivos específicos sobre los currículums

1- Utilizar la plantilla para currículums CICYT (Comisión Interministerial de Ciencia y Tecnología) como base para la creación y muestra del currículum.
2- Utilizar un sistema específico para organizar los méritos relevantes.
3- Permitir al usuario seleccionar si quiere que se muestren todas las secciones del currículum o desea hacer una selección manual.
4- Permitir al usuario elegir el orden en el que se muestran las secciones, en caso de que estas existan.
5- Permitir al usuario configurar la forma en la que se muestran las secciones, de manera que pueda elegir el orden en que se mostrarán los campos de cada sección. Permitir además seleccionar qué campos se muestran y cuales no.
6- Volcar el contenido de un currículum en un documento de MS Word (.doc) de forma que se pueda editar a mano cualquier pequeño detalle o enviar fácilmente.
7- Incluir nuevos campos de interés en el currículum, como por ejemplo el sub proyecto, el ámbito o el índice de impacto.
8- Permitir especificar un rango de fechas antes de mostrar un currículum. Solo se mostrarán las entradas que ocurran dentro de ese rango. Permite además mostrar las entradas en orden cronológico ascendente o descendente.
9- Separar los campos según el ámbito (nacional o internacional) y el investigador principal (el autor del currículum u otro distinto).
10- Permitir mostrar los campos en líneas diferentes o todos seguidos separados por puntos. En el último caso, o se deben mostrar los nombres de los campos, solo el contenido.

2 - Bases de datos

2.1 - Introducción

Una base de datos es un conjunto de la información que se almacena en un ordenador de una manera sistemática, de modo que un programa pueda consultarla para contestar a preguntas. La información recuperada en contestación a preguntas se convierte en la información que se puede utilizar para tomar decisiones. El programa que maneja y preguntar una base de datos se conoce como sistema de gestión de base de datos (DBMS - database management system). Las características y el diseño de los sistemas de la base de datos se incluyen en el estudio de las ciencias de la información.

El concepto central de una base de datos es el de un conjunto de información o de fragmentos de conocimiento. Típicamente, para una base de datos dada, hay una descripción estructural del tipo de hechos llevados a cabo en esa base de datos: esta descripción se conoce como esquema. El esquema describe los objetos que se representan en la base de datos y las relaciones entre ellas. Hay diversas maneras de organizar un esquema, es decir, de modelar la estructura de la base de datos: se conocen como modelos de datos. El modelo más común hoy es el modelo relacional, que representa toda la información bajo la forma de múltiples tablas relacionadas que consisten en filas y columnas. Este modelo representa relaciones por el uso de los valores comunes a más de una tabla. Otros modelos tales como el modelo jerárquico y el modelo de la red utilizan una representación más explícita de relaciones.

Muchos profesionales considerarían que una colección de datos constituye una base de datos solamente si tiene ciertas características. Por ejemplo, si los datos se manejan para asegurar su integridad y calidad, si permiten el acceso compartido de una comunidad de usuarios, si tienen un esquema o si apoyan un lenguaje de interrogación. Sin embargo, no hay definición convenida de estas características.

Los sistemas de gestión de base de datos se categorizan generalmente según el modelo de los datos que apoyan: relacional, relacional con objetos, de red y así sucesivamente. El modelo de los datos tenderá a determinar los lenguajes de consulta que están disponibles para tener acceso a la base de datos. La ingeniería interna mucha de un DBMS, sin embargo, es independiente del modelo de los datos y se dedica a factores tales como funcionamiento, concurrencia, integridad y recuperación de faltas del hardware. En estas áreas hay diferencias grandes entre los productos.

Las bases de datos tienen muchos usos, cubriendo virtualmente la gama entera del software. Las bases de datos son el método preferido de almacenaje para las grandes aplicaciones multiusos, donde es necesaria la coordinación entre muchos usuarios. Incluso los usuarios individuales las encuentran convenientes y se basan muchos programas de correo electrónico y organizadores personales en tecnología estándar de la base de datos. Los drivers para bases de datos están disponibles para la mayoría de las plataformas, de modo que el software de aplicación pueda utilizar un interfaz de programación de uso común (API) para recuperar la información almacenada en una base de datos. Dos APIs de base de datos de uso general son JDBC y ODBC. Una base de datos es también un lugar en donde se puede almacenar datos y después ordenarlos fácil y eficientemente.

2.2 - MySQL

2.2.1 - Introducción

MySQL es un sistema de gestión de base de datos, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado lo ofrece bajo la GNU GPL, pero, empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia que les permita ese uso.

Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como el Apache, donde el software es desarrollado por una comunidad pública, y el copyright del código está en poder del autor individual, MySQL está poseído y patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson, y Michael Widenius.

SQL (*Lenguaje de Consulta Estructurado*) fue comercializado por primera vez en 1981 por IBM, el cual fue presentado a ANSI y desde ese entonces ha sido considerado como un estándar para las bases de datos relacionales. Desde 1986, el estándar SQL ha aparecido en diferentes versiones como por ejemplo: SQL:92, SQL:99, SQL:2003. MySQL es una idea originaria de la empresa opensource MySQL AB establecida inicialmente en Suecia en 1995 y cuyos fundadores son David Axmark, Allan Larsson, y Michael "Monty" Widenius. El objetivo que persigue esta empresa consiste en que MySQL cumpla el estándar SQL, pero sin sacrificar velocidad, fiabilidad o usabilidad.

Michael Widenius en la década de los 90 trató de usar mSQL para conectar las tablas usando rutinas de bajo nivel ISAM, sin embargo, mSQL no era rápido y flexible para sus necesidades. Esto lo conllevó a crear una API SQL denominada MySQL para bases de datos muy similar a la de mSQL pero más portable.

Existen varias APIs que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi (via dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby, REALbasic (Mac), FreeBASIC, y Tcl; cada uno de estos utiliza una API específica. También existe un interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL.

MySQL es muy utilizado en aplicaciones web como MediaWiki o Drupal, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

2.2.2 – Ejemplos

```
CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20), species
VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
```

Crea una tabla llamada ‘pet’ con los siguientes campos:
- name, owner, species: admiten cualquier tipo de carácter, pero hasta un máximo de 20.
- sex: admite un único carácter de texto.
- birth, death: admiten fechas en formato standard de MySQL (aaaa-mm-dd)

```
INSERT INTO pet VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL);
```

Introduce en la tabla creada anteriormente los valores 'Puffball', 'Diane', 'hamster', 'f', '1999-03-30' y NULL. Al no indicarse donde hay que introducirlos estos se asignan por orden a los campos existentes. El valor NULL significa que se deja el campo en blanco.

```
UPDATE pet SET birth = '1989-08-31' WHERE name = 'Bowser';
```

Actualiza la tabla ‘pet’. Busca una línea donde el campo name tenga el valor Bowser y para esa línea asigna el valor 1989-08-31 en el campo birth.

```
SELECT * FROM pet WHERE (species = 'cat' AND sex = 'm') OR (species = 'dog' AND
sex = 'f');
```

Muestra todas (indicado por el asterisco *) las columnas de la tabla ‘pet’ que pertenezcan a líneas que cumplan las condiciones de ser a la vez de especie gato y sexo masculino o de especie perro y sexo femenino.

```
SELECT name, species, birth FROM pet WHERE species = 'dog' OR species = 'cat';
```

Parecida a la búsqueda anterior. Esta vez busca en la tabla pet las líneas donde el campo species tenga el valor dog o cat, pero como resultado solamente muestra las columnas name, species y birth.

```
SELECT DISTINCT owner FROM pet;
```

Busca en la tabla pet todos los valores que existen en la columna owner. El valor DISTINC hace que no se muestren los valores repetidos.

```
SELECT name, species, birth FROM pet ORDER BY species, birth DESC;
```

Muestra todos los valores de las columnas name, species y birth contenidos en la tabla pet. Los valores con previamente ordenados, primero por el valor del campo species y posteriormente por el valor de birth de manera descendente.

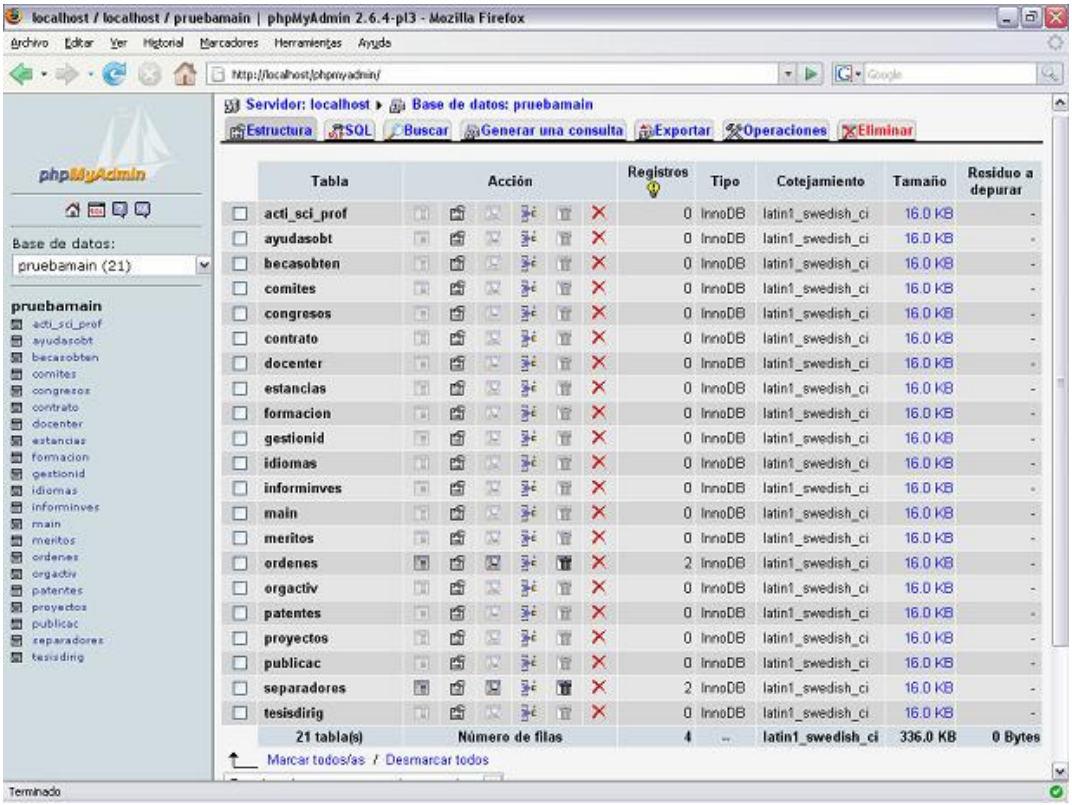
```
SELECT * FROM pet WHERE name LIKE '%fy';
```

Muestra todas las columnas de todas las filas de la tabla pet donde el valor de la columna name acabe en fy, como por ejemplo Fluffy o Buffy. En este caso el caracter % hace las veces de comodín y puede colocarse en cualquier parte de la cadena. Por ejemplo b% mostraría los que empiezan por la letra b mientras que %x% mostraría los que contienen la letra x.

2.2.3 – Uso en la aplicación

El uso de MySQL en este trabajo se reduce a la creación de las tablas, conexión a la base de datos y la manipulación de datos.

La creación de las tablas es externa a la aplicación, ya que fueron creadas previamente mientras se diseño la aplicación y no es necesario modificarlas una vez creadas. En un principio fueron creadas directamente introduciendo el código en la consola de comandos, pero posteriormente se utilizó la aplicación PHP MyAdmin para simplificar el proceso, especialmente en la modificación de tablas que es más propensa a errores si se introducen comandos equivocados.



Base de datos del proyecto visto desde PHP MyAdmin

La conexión a la base de datos y la manipulación de datos se realiza enviando los comandos a las bases de datos como argumentos en código PHP. La cantidad de acciones que se realizan incluyen la creación de nuevas entradas en las tablas, inserción de datos en dichas entradas, modificación de datos existentes y eliminación de entradas.

3 - Lenguajes de programación

3.1 - HTML

3.1.1 - Introducción

El HTML, acrónimo inglés de HyperText Markup Language (lenguaje de marcas hipertextuales), lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores, el HTML se ha convertido en uno de los formatos más populares que existen para la construcción de documentos y también de los más fáciles de aprender.

HTML es una aplicación de SGML conforme al estándar internacional ISO 8879. XHTML es una reformulación de HTML 4 como aplicación XML 1.0, y que supone la base para la evolución estable de este lenguaje. Además XHTML permite la compatibilidad con los agentes de usuario que ya admitían HTML 4 siguiendo un conjunto de reglas.

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser el Bloc de Notas de Windows o cualquier otro editor que admita texto sin formato como Emacs. Existen además, otros programas para la realización de sitios Web o edición de código HTML, como por ejemplo Microsoft FrontPage, el cual tiene un formato básico parecido al resto de los programas de Office. También existe el famoso software de Macromedia (que adquirió la empresa Adobe) llamado Dreamweaver, siendo uno de los más utilizados en el ámbito de diseño y programación Web. El código de esta aplicación fue escrito con PHP Designer 2006.

HTML utiliza etiquetas o marcas, que consisten en breves instrucciones de comienzo y final, mediante las cuales se determinan la forma en la que debe aparecer en su navegador el texto, así como también las imágenes y los demás elementos, en la pantalla del ordenador.

Toda etiqueta se identifica porque está encerrada entre los signos menor que y mayor que (<>), y algunas tienen atributos que pueden tomar algún valor. En general las etiquetas se aplicarán de dos formas especiales:

- Se abren y se cierran, como por ejemplo: negrita que se vería en su navegador como negrita.
- No pueden abrirse y cerrarse, como <hr> que se vería en su navegador como una línea horizontal.

En 1989 existían dos técnicas que permitían vincular documentos electrónicos, por un lado los hipervínculos (links) y por otro lado un poderoso lenguaje de etiquetas denominado SGML. Por entonces un usuario conocedor de ambas opciones, Tim Berners-Lee físico nuclear del Centro Europeo para la Investigación Nuclear da a conocer a la prensa que estaba trabajando en un sistema que permitirá acceder a ficheros en línea, funcionando sobre redes de computadoras o máquinas electrónicas basadas en el protocolo TCP/IP.

Principios de 1990, Tim Berners-Lee define por fin el HTML como un subconjunto del conocido SGML y crea algo más valioso aun, el World Wide Web. En 1991, Tim Berners-Lee crea el primer navegador de HTML que funcionaría en modo texto y para UNIX.

Los trabajos para crear un sucesor del HTML, posteriormente llamado 'HTML+', comenzaron a finales de 1993. El HTML+ se diseñó originalmente para ser un superconjunto del HTML que permitiera evolucionar gradualmente desde el formato HTML anterior. A la primera especificación formal de HTML+ se le dio, por lo tanto, el número de versión 2.0 para distinguirla de esos "estándares no oficiales" previos. Los trabajos sobre HTML+ continuaron, pero nunca se convirtió en un estándar.

El borrador del estándar HTML 3.0 fue propuesto por el recién formado W3C en marzo de 1995. Con él se introdujeron muchas nuevas capacidades, tales como facilidades para crear tablas, hacer que el texto fluyese alrededor de las figuras y mostrar elementos matemáticos complejos. Aunque se diseñó para ser compatible con HTML 2.0, era demasiado complejo para ser implementado con la tecnología de la época y, cuando el borrador del estándar expiró en septiembre de 1995, se abandonó debido a la carencia de apoyos de los fabricantes de navegadores web. El HTML 3.1 nunca llegó a ser propuesto oficialmente, y el estándar siguiente fue el HTML 3.2, que abandonaba la mayoría de las nuevas características del HTML 3.0 y, a cambio, adoptaba muchos elementos desarrollados inicialmente por los navegadores web Netscape y Mosaic. La posibilidad de trabajar con fórmulas matemáticas que se había propuesto en el HTML 3.0 pasó a quedar integrada en un estándar distinto llamado MathML. El HTML 4.0 también adoptó muchos elementos específicos desarrollados inicialmente para un navegador web concreto, pero al mismo tiempo comenzó a limpiar el HTML señalando algunos de ellos como 'desaprobados'.

Ya no va a haber nuevas versiones del HTML. Sin embargo, la herencia del HTML se mantiene en XHTML, que se basa en XML.

3.1.2 - Ejemplos

```
<HTML>
  <HEAD>
    <TITLE>Nombre</TITLE>
  </HEAD>
  <BODY>
    Contenido
  </BODY>
</HTML>
```

La estructura general de todo documento HTML. Para empezar, todo debe estar incluido dentro de las etiquetas de HTML. Dentro de ellas encontramos dos partes principales:
- El encabezado, marcado por las etiquetas HEAD, contiene información sobre el documento actual, como el título (marcado por las etiquetas TITLE), palabras clave que pueden ser de utilidad para motores de búsqueda, y otros datos que no se consideran parte del contenido del documento.
- El cuerpo del documento, delimitado por las etiquetas BODY, contiene el contenido del documento.

```
<TABLE>
  <TR>
    <TD>Contenido</TD>
    <TD>Contenido</TD>
  </TR>
  <TR>
    <TD>Contenido</TD>
    <TD>Contenido</TD>
  </TR>
</TABLE>
```

El modelo de tablas de HTML permite a los autores organizar datos -- textos, texto preformateado, imágenes, vínculos, formularios, campos de formularios, otras tablas, etc. en filas y en columnas de celdas. Dentro de la tabla observamos el elemento TR y TD:
- TR se utiliza para indicar cada una de las filas que contiene la tabla.
- TD se utiliza para cada unidad de datos que contiene la fila y pueden tratarse como si fuesen las diferentes columnas.

```
<A href="http://www.w3.org/">sitio web del W3C</A>
```

El elemento A (anchor) se utiliza para crear los enlaces o hipervinculos con los que podemos desplazarnos a partes del documento o a diferentes páginas. El valor del atributo href indica la URL de destino mientras que el texto entre las etiquetas es el texto que se muestra.

```
<FORM action="http://algunsitio.com/prog/usuarionuevo" method="post">
   Nombre: <INPUT type="text" id="nombre"><BR>

   <INPUT type="checkbox" name="A" value="V"> opt1<BR>
   <INPUT type="checkbox" name="A" value="M"> opt2<BR>

   <SELECT name=”menu”>
      <OPTION value=”1”>texto 1</OPTION>
      <OPTION selected value=”2”>texto 2</OPTION>
      <OPTION value=”3”>texto 3</OPTION>
   </SELECT>

   <TEXTAREA name=”ta”></TEXTAREA>

   <INPUT type="submit" value="Enviar"> <INPUT type="reset">
</FORM>
```

El uso de formularios es una parte muy importante en esta aplicación, tanto que la mayoría de las páginas creadas contienen uno o más. Los formularios se indican con las etiquetas FORM y pueden contener diversos tipos de elementos, como por ejemplo:
- INPUT puede tener diferentes usos según cual sea su tipo, como por ejemplo:
--- Los del tipo TEXT crean líneas de texto para que el usuario escriba en ellas.
--- Los del tipo CHECKBOX se utilizan para crear casillas que podemos marcar o desmarcar.
--- Los del tipo SUBMIT crean el botón que hay que pulsar para enviar el documento a donde vaya a ser procesado (indicado por el valor del campo action que hay en la etiqueta FORM).

--- Los del tipo RESET crean un botón que restaura los valores originales del formulario.
- SELECT crea menús desplegables. Cada una de las opciones posibles incluidas en el menú están marcadas por las etiquetas OPTION. Es posible preseleccionar una de ellas mediante el atributo selected.
- TEXTAREA crea cuadros de texto. Al contrario que los INPUT de tipo text estas cajas pueden ocupar varias líneas, por lo que se usan para grandes cantidades de texto.

3.1.3 – Uso en la aplicación

HTML es el lenguaje base de la aplicación, ya que es el que al final se le pasa al navegador para que lo interprete. Se ha utilizado HTML para la creación de todos los elementos simples, como tablas, botones, listas, etc. Muchas veces no se escribe el código HTML directamente, sino que el código se le pasa como argumento a PHP para que este lo escriba con el mismo resultado, pero con la ventaja de que escribiéndolo indirectamente podemos añadir variables y código especial.

3.2 - PHP

3.2.1 - Introducción

PHP es un lenguaje de programación usado generalmente para la creación de contenido para sitios web. PHP es un acrónimo recurrente que significa "PHP Hypertext Pre-processor" (inicialmente PHP Tools, o, *P*ersonal *H*ome *P*age Tools), y se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web. Últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la biblioteca GTK+.

El fácil uso y la similitud con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores experimentados crear aplicaciones complejas con una curva de aprendizaje muy suave. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas.

Debido al diseño de PHP, también es posible crear aplicaciones con una interfaz gráfica para el usuario (también llamada GUI), utilizando la extensión PHP-GTK. También puede ser usado desde la línea de órdenes, de la misma manera como Perl o Python pueden hacerlo, esta versión de PHP se llama PHP CLI (*Command Line Interface*).

Su interpretación y ejecución se da en el servidor, en el cual se encuentra almacenado el script, y el cliente sólo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página web, generada por un script PHP, el servidor ejecuta el intérprete de PHP, el cual procesa el script solicitado que generará el contenido de manera dinámica, pudiendo modificar el contenido a enviar, y regresa el

resultado al servidor, el cual se encarga de regresárselo al cliente. Además es posible utilizar PHP para generar archivos PDF, Flash, así como imágenes en diferentes formatos, entre otras cosas.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de Aplicaciones web muy robustas.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX (y de ese tipo, como Linux), Windows y Mac OS X, y puede interactuar con los servidores de web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

El modelo PHP puede ser visto como una alternativa al sistema de Microsoft que utiliza ASP.NET/C#/VB.NET, a ColdFusion de la compañía Macromedia, a JSP/Java de Sun Microsystems, y al famoso CGI/Perl. Aunque su creación y desarrollo se da en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un IDE comercial llamado Zend Optimizer.

3.2.2 - Ejemplos

```
<?php codigo_PHP ?>
```

Lo primero que hay que comentar es la secuencia de escape. Es necesario encerrar el código PHP entre señales que indiquen que es PHP y no HTML para que el servidor realice el procesado necesario. Se puede utilizar una misma pareja de marcas para un mismo bloque de texto, pero hay que tener en cuenta que cualquier código que haya dentro y que no sea PHP, como por ejemplo HTML, no solo no se procesará, sino que resultará en error. Para los ejemplos siguientes se omitirán estas marcas.

```
echo "texto";
echo “<A href="http://www.w3.org/">W3C Web site</A>”;
```

La instrucción echo imprime por pantalla el texto introducido. Es posible utilizarla para introducir código HTML en mitad de un bloque PHP con la ventaja de que puede contener variables que pueden ser modificadas cuando se procese la página en el servidor.

```
function double($x) {
    return $x*2;
}
$a = 5;
$b = double($a);
```

PHP permite al usuario definir sus propias funciones como la mayoría de los lenguajes de programación y llamarlas en cualquier momento.

```
if ($a > $b) {
   print "a es mayor que b";
} elseif ($a == $b) {
   print "a es igual que b";
} else {
   print "a es mayor que b";
}
```

PHP permite el uso de gran cantidad de estructuras de control, entre ellas estan if, else, ifelse, while, do...while, for, foreach, break, continue, switch, return...

```
$link = mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) or die ('No conecta.' .
mysql_error());
$db_selected = mysql_select_db (DB_NAME, $link) or die ('No puede usar la tabla.' .
mysql_error());
$query = "SELECT * FROM tabla WHERE xxxx='yyyyy'";
$result = mysql_query ($query) or die ('La consulta falló' .mysql_error());
```

Tres ejemplos de como PHP interactúa con MySQL:
- En el primero se realiza la conexión a MySQL introduciendo la dirección donde se encuentra, el nombre del usuario y su contraseña. Obtenemos un identificador si la conexión se realiza con éxito o un mensaje de error en caso de haber algún error.
- El segundo muestra como utilizamos el identificador principal y el nombre de la base de datos para poder conectarnos a ella. Nos devuelve otro identificador o un mensaje de error según la situación.
- El tercer ejemplo muestra como se realiza una consulta normal mediante PHP. La variable $query almacena el código MySQL que queremos ejecutar y se le pasa a la función mysql_query() para que realice la consulta, la cual se almacena en $result.

3.2.3 – Uso en la aplicación

PHP es uno de los lenguajes más utilizados en la creación de la aplicación. Es un lenguaje que se procesa en el servidor, lo cual lo hace potente y seguro, pero también hace que sea lento de procesar, ya que cada acción necesita enviar la solicitud al servidor y esperar que este la procese y le devuelva la respuesta en código HTML. Uno de los principales usos de PHP en esta aplicación es el de permitir la comunicación entre HTML y MySQL para poder enviar y recibir información de las bases de datos, además de establecer la conexión y comprobar que no hay errores en la comunicación con las bases de datos, ya que estas se encuentran alojadas en el servidor y HTML no tiene suficiente potencia para tratarlas con ellas por si solo. PHP examina los datos antes de ser enviados mediante MySQL y tras recibirlos para traducirlos entre un formato fácil de almacenar en las bases de datos y uno que se le pueda mostrar al usuario de la aplicación.
El uso de PHP va más allá de la comunicación con las bases de datos, ya que al tratarse de un lenguaje de programación permite realizar todo tipo de opciones imposibles para HTML. Como se dijo en la sección de HTML, podemos escribir dicho código indirectamente usando PHP, permitiéndonos utilizar variables en lugar de valores fijos. El uso de variables mediante PHP permite reducir considerablemente el tamaño del programa, ya que varias páginas que eran similares, como la de añadir nuevos datos y la de editarlos se han podido unir en una sola capaz de realizar ambas acciones, lo cual casi reduce a la mitad el tamaño de la aplicación. Mediante el uso de variables podemos

ocultar secciones, por ejemplo en el caso de que solo decidamos mostrar determinados campos o de que estos se encuentren vacíos.

3.3 - JAVASCRIPT

3.3.1 - Introducción

Javascript es el nombre de la fundación de Mozilla del estándar de ECMAScript, un lenguaje de scripts basado en el concepto de la programación basada en prototipos. La lengua es la más conocida para su uso en páginas web, pero también se utiliza para permitir a los objetos contenidos en otras aplicaciones.

A pesar del nombre, Javascript solamente tiene una ligera relación con el lenguaje de programación de Java, la semejanza principal es su deuda común a la sintaxis de C. Semánticamente, la sintaxis de Javascript tiene más en común con el lenguaje de programación SELF.

Javascript es una marca registrada de Sun Microsystems, inc. Fue utilizado bajo de la licencia para la tecnología inventada e implementada por Netscape Communications y entidades actuales tales como la fundación Mozilla.

Javascript es una lengua de scripts basada en prototipos con una sintaxis basada en C. Como C, la lengua no tiene ninguna construcción de la entrada o de la salida por si misma. Donde C confía en bibliotecas estándares de la entrada-salida, el motor del Javascript confía en un *ambiente de anfitrión* en el cual se encuentra. Hay muchos ambientes de anfitrión, entre los cuales las tecnologías de la red son los ejemplos más conocidos. Éstos se examinan primero.

Un uso importante del Javascript basado en redes es escribir las funciones que se incluyen en las páginas HTML y obran recíprocamente con el modelo del objeto del documento (DOM) de la página para realizar las tareas no posibles solamente en HTML. Algunos ejemplos comunes de este uso son.

- Abriendo una ventana nueva pudiendo controlar el tamaño, la posición y la apariencia de esta (es decir si los menús, las barras de herramientas, etc son visibles o no).
- La validación de formularios de web para asegurarse de que los valores serán aceptados antes de que se envíen al servidor.
- Cambiar imágenes cuando el cursor del ratón se mueve sobre ellas: Este efecto es de uso frecuente para llamar la atención del usuario a los enlaces importantes exhibidos como elementos gráficos.

Los interfaces DOM en los navegadores se diferencian y no son iguales siempre a los estándares de W3C DOM. En vez de escribir diversas variantes de una función para cada uno de los muchos navegadores en uso común hoy, es posible, siguiendo cuidadosamente los estándares del nivel 1 o 2 de W3C DOM, proporcionar la funcionalidad requerida de una manera estándar que la mayoría de los navegadores puedan ejecutar correctamente. El cuidado se debe tomar siempre para asegurarse de que la página web sea tolerante a fallos y así que aún así sigue siendo usable por cualquier usuario que:

- Tiene la ejecución del Javascript deshabilitada - por ejemplo como precaución de seguridad.
- Tiene un navegador que no entiende el Javascript - por ejemplo en un PDA o un teléfono móvil
- Se encuentra inhabilitado visualmente o de otra manera, por lo que puede utilizar un navegador inusual, como uno parlante o que tenga habilitada la ampliación extrema del texto.

Los otros ejemplos de Javascript que interactúan con el DOM de una página web se han llaman DHTML y SPA.

Un ejemplo diferente del uso de Javascript en páginas web es hacer llamadas a los servidores de la red y a los servidores de red después de que la página haya cargado, dependiendo de acciones del usuario. Estas llamadas pueden obtener la nueva información, que Javascript puede combinar con DOM de la página existente para que se muestre. Ésta es la base de la programación de Ajax. El patrón del diseño de Javascript PnP fue adoptado gradualmente después de que se usase de forma común Ajax para reducir coste de mantenimiento de Javascript.

Fuera de la red, los intérpretes del Javascript se encuentran en un número de herramientas. Adobe Acrobat y Adobe Reader permiten Javascript en archivos pdf. La plataforma de Mozilla, que es la base de varios navegadores de red, utiliza Javascript para implementar el interfaz de usuario y las transacciones lógicas de varios de sus productos. También en aplicaciones propietarias que carecen interfaces que permitan scripts. Dashboard Widgets en el Apple Mac OS x v10.4 están implementados usando Javascript. La tecnología Active Scripting de Microsoft apoya el Javascript compatible JScript como lengua scripting del sistema operativo. JScript .NET es un lenguaje que es similar a JScript, pero tiene otras características de programación orientadas a objetos. Las aplicaciones del Adobe Creative Suite, incluyendo Photoshop, permiten crear scripts usando Javascript.

El lenguaje de programación de Java, en la versión SE 6 (JDK 1.6) introdujo el paquete `javax.script` que permite que cualquier aplicación de Java lea, interprete y ejecute el código de Javascript en el tiempo de ejecución. El desarrollador de Java puede hacer los objetos y las variables que son parte de la aplicación anfitriona disponible para el código del Javascript usando un objeto de tipo Bindings. Estos aspectos de aplicaciones en uso se pueden acceder y manipular en el tiempo ejecución de Javascript de una manera similar a la manera que las escrituras del lado del cliente tienen acceso al DOM de una página exhibida en un navegador web.

Cada uno de estos usos tiene su propio modelo de objeto que proporciona el acceso al ambiente del anfitrión, permaneciendo la mayoría del núcleo de Javascript idéntico en cada aplicación.

3.3.2 – Ejemplos

```
<script type="text/javascript">
…
</script>
```

Al igual que en PHP es necesario indicar en el código que vamos a insertar código JavaScript, aunque solo se utiliza para la zona en la que están definidas las funciones, ya que podemos llamarlas desde HTML sin necesidad de realizar ningún aviso al navegador.

```
var x=document.getElementById("nombre").value
```

```
var $xdia=document.getElementById("dian");
var $zdia=$xdia.options[$xdia.selectedIndex].value;
```

Muchas veces JavaScript necesita conocer información de los elementos de HTML para poder realizar acciones sobre ellos. En este ejemplo se muestra como puede conseguirlos:
- En el primer caso buscamos en el documento un elemento a partir de su identificador y después conseguimos su valor.
- El segundo caso es un poco más complicado, ya que necesitamos conseguir un valor de uno de los elementos de un menú desplegable. Primero es necesario conseguir el identificador del menú y una vez lo tenemos hay que pedirle a JS que obtenga el valor de la opción que se encuentre marcada.

```
alert ("mensaje");
```

La función alert es útil para informar al usuario en el instante. Cuando se llama a dicha función aparece una ventana con un mensaje y un botón para cerrarla

```
var X = confirm("Texto de la pregunta");
if (X == true){
…
} else {
…
}
```

La función confirm es parecida a alert. En este caso cuando se abre la ventana aparece un texto junto con un botón de aceptar y uno de cancelar. Es posible asignar una acción a cada botón.

3.3.3 – Uso en la aplicación

En un principio se tenía pensado utilizar solo PHP para manejar todo el control, pero la idea fue desechada ya que era demasiado lento cuando solo queremos realizar pequeñas tareas, ya que tiene que ponerse en contacto con el servidor y volver a conseguir la página en cada acción. El lenguaje JavaScript tiene la flexibilidad para realizar dichas tareas rápidamente, ya que es el mismo navegador el que se encarga de procesarlo por lo que no es necesario realizar ninguna transmisión extra, pero como desventaja tiene que

no puede trabajar con la información de la base de datos, ya que esta se encuentra en el servidor y que es poco seguro, ya que cualquier usuario puede examinar el código JavaScript de una página integro desde su navegador, por lo que no debe utilizarse para manejar datos sensibles.

El principal uso de JavaScript en la aplicación es la validación de los datos que introduce el usuario antes de ser enviados al código PHP que los introduce en la base de datos, impidiendo el envío y avisando al usuario en el instante de cuales son los datos que no son validos mediante una pequeña ventana emergente. Se utiliza también para creación de botones con funciones especiales, como el marcar varias casillas pulsando solo una o para bloquear campos en los que no se puede escribir sin que se cumplan ciertas condiciones.

4 - Implementación de los objetivos

4.1 - Bases de datos

Para la implementación de la aplicación se han creado un total de 21 tablas en la base de datos.

Para cada currículum introducido se rellena una línea de la tabla *'main'*, en la que se introducen los datos básicos del usuario y se le asigna un identificador *'id'* automáticamente a ese currículum para distinguirlo de otros. Basándonos en dicho identificador, se pueden enlazar tantas tablas como se deseen de los siguientes tipos para completar la información: *'acti_sci_prof', 'ayudasobt' , 'becasobten' , 'comites' , 'congresos' , 'contrato' , 'docenter' , 'estancias' , 'formacion' , 'gestionid' , 'idiomas' , 'informinves' , 'orgactiv' , 'patentes' , 'proyectos' , 'publicac'* y *'tesisdirig'*.

Estas tablas nombradas contienen un campo para cada uno de los datos que se pueden introducir para completar ese fragmento de información del currículum. Todas esas tablas tienen además dos campos especiales. El primero, *'id'*, es la referencia ya citada a la entrada de la tabla *'main'* a la que pertenecen. El segundo, *'tid'*, es un identificador propio único que ayuda a distinguir entre las diferentes entradas cuando hay varias en la misma tabla con un mismo *'id'*. Todas estas tablas cuentan además con un campo *'coment'* extra en el que se puede escribir cualquier texto que se desee para completar la información en el caso de que los campos normales no sean suficientes o se quiera añadir algún dato relevante.

Además de las tablas citadas, cuando se crea la entrada en *'main'* se crea también una sola entrada en *'meritos'* en la que se almacenarán todos los méritos asociados al currículum en vez de tener una línea independiente para cada uno en una tabla. Dicha entrada cuenta solamente con un campo *'id'* que la enlaza con la entrada en *'main'*, ya que es única para todo el currículum.

Además de todas las tablas nombradas previamente que completan la información de un currículum, existen dos tablas especiales: *'ordenes'* y *'separadores'*. Ambas tienen una estructura parecida, ya que constan de una línea por defecto y una configurable, que el usuario puede modificar para alterar la apariencia de como se muestra el currículum.

4.2 - Nuevas entradas / Editar entradas

En esta sección se explica el funcionamiento de los siguientes archivos: *main.php, actividad.php, ayuda.php, beca.php, comites.php, congreso.php, contrato.php, docencia.php, estancias.php, formacion.php, gestionid.php, idioma.php, informe.php, meritos.php, organizacion.php, patentes.php, proyecto.php, publicacion.php y tesis.php*. De todos ellos, la mayoría son similares, excepto main.php y meritos.php, que son ligeramente diferentes como se explicará más adelante.

Todas estas páginas están divididas en 3 secciones principales: código Javascript, código de gestión del formulario y código del formulario.

Para que se puedan realizar las consultas a MySQL es necesario realizar la conexión con la base de datos. Dicha conexión se realiza tras la parte de Javascript, de manera que las dos siguientes secciones puedan acceder a ella. El código de conexión, que también se utilizará en otras páginas totalmente diferentes, tiene esta forma:

```
// Definimos la información de conexión.
DEFINE (DB_USER, "username");
DEFINE (DB_PASSWORD, "password");
DEFINE (DB_HOST, "localhost");
DEFINE (DB_NAME, "pruebamain");

// Conectamos a la DB
$link = mysql_connect (DB_HOST, DB_USER, DB_PASSWORD);
if (!$link) {
        die ('No conecta:' . mysql_error());
}
$db_selected = mysql_select_db (DB_NAME, $link);
if (!$db_selected) {
        die ('No puede usar la tabla:' . mysql_error());
}
```

Cambiando *username* y *password* por sus valores correspondientes.

Código Javascript:

El código Javascript se encarga principalmente de comprobar, antes de enviar los datos, que la información introducida en el formulario es correcta y se encuentre en un formato adecuado. Además comprueba que los campos obligatorios han sido escritos. En caso de que haya algún problema, una ventana de información nos avisará de cual es el problema. Merece mención especial el trato de las fechas, para el que se ha creado una función que comprueba que el día introducido es compatible con dicho mes y año, cuidando incluso el control sobre los años bisiestos.

```
function validafecha($dia, $mes, $anio){
        // Comprueba que se ha introducido un año
        if ((isNaN($anio)==true)){
                alert ("Es necesario introducir un año");
                retun (false);
        } else if (($dia != '00') && ($mes == '00')){ //Procesamos día
                alert ("El campo mes es requisito para introducir el día");
                return (false);
        }
        // Comprobamos las restricciones de día según el mes
        // Meses 30 días: Abril(04), Junio(06), Septiembre(09), Noviembre(11)
        // Febrero: 28 o 29 días
        if ((($dia == '31') && ($mes == ('04' || '06' || '09' || '11'))) || (($mes == '02')
&& ($dia == ('31' || '30')))){
                alert ("El día no es válido para el mes seleccionado");
                return (false);
        }
        // Comprobamos si es año bisiesto
        // Un año es bisiesto si es divisible por 4, excepto aquellos divisibles por 100
pero no por 400.
        if (($mes == '02') && ($dia == '29')){
                if (($anio % 4) == 0){
```

```
                        if (($anio % 100) == 0){
                                if (($anio % 400) != 0){
                                  alert ("El año introducido no es bisiesto. Corrija el día");
                                   return (false);
                                }
                        }
                        return (true);
                } else {
                        alert ("El año introducido no es bisiesto. Corrija el día");
                        return (false);
                }       }
        return (true);
}
```

En el código Javascript también hay una pequeña función de ayuda al borrado de la tabla. En caso de que elijamos borrarla, dicha función se encarga de redirigirnos a la página de eliminación activando un pequeño formulario que se explica más adelante mediante el método *submit()*.

Código del formulario:

Aunque el código del formulario se encuentra al final del archivo es conveniente explicarlo primero. Cuando el navegador accede a esta parte del código comprueba si la variable *tid* que recibimos mediante el método POST de HTML está activada. En caso de estarlo tenemos que tratar con datos previamente introducidos para editarlos, por lo que habrá que procesar la petición para obtenerlos de la base de datos

```
$query = "SELECT * FROM tabla WHERE tid='$tid'";
```

Donde *tabla* se corresponde a la tabla en la que se almacena la información y el *tid* es el identificador único de dicha fila. Si no está activada vamos a crear una nueva entrada de datos en la tabla, por lo que no es necesario recuperar ninguna información previa. Modificamos además la variable *$ae* según si vamos a añadir información nueva a la tabla o estamos editándola. Esta variable se utilizará después.

El formulario es el Standard de HTML, encerrado en los tags <form> y compuesto principalmente por elementos de tipo <input>, tanto visibles como ocultos, <select> y <textarea>. En caso de que estemos rellenado el formulario por primera vez, todos los campos se encontraran vacíos, pero si estamos editándolo podremos ver el contenido original que obtuvimos previamente de la base de datos. Una vez rellenemos los datos, el botón Aceptar avisa a la sección de Javascript para que realice las comprobaciones y si estas son correctas enviamos el formulario a la misma página para su inserción en la base de datos.

Lo más destacable de está sección es la manera de tratar los menús desplegables del tipo <select> cuando estamos editando, ya que es necesario que aparezca marcada la opción que se introdujo previamente. Exclusivamente para estos casos fue necesaria la creación de una nueva función que mecanizase el proceso, ya que realizar las comprobaciones por otros métodos aumentaba considerablemente el tamaño del código.

```
// Función para rellenar menús desplegables
function desplegable ($ar1, $ar2, $max, $sel){
        for ($buc=0; $buc<$max; $buc=$buc+1){
                if ($ar1[$buc]== $sel) {
                        echo "<option value=$ar1[$buc] selected>$ar2[$buc]</option>";
                } else {
                        echo "<option value=$ar1[$buc]>$ar2[$buc]</option>";
                }
        }
}
```

Los argumentos *$ar1 y $ar2* se corresponden con cadenas previamente introducidas en el código. El primero contiene los valores que se introducirán en la base de datos en caso de que selecciona dicha opción, mientras que el segundo contiene los valores que se muestran por pantalla. Por ejemplo, para el menú desplegable de los meses tenemos estas cadenas:

```
$val_mes = array('00','01','02','03','04','05','06','07','08','09','10','11','12');
$tex_mes = array('--', 'Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio', 'Julio', 'Agosto',
'Septiembre', 'Octubre', 'Noviembre', 'Diciembre');
```

Los otros dos argumentos indican respectivamente la cantidad máxima de elementos que tiene el menú y cual es el elemento previamente seleccionado. En caso de que estemos mostrando el formulario para introducir datos por primera vez, el método escribirá todos los campos como no marcados sin tener que indicarle nada

En caso de que el formulario se muestre para editar datos aparecerá un botón de borrado al final de la página. Dicho botón es en verdad la única parte visible de un segundo formulario que contiene la información que hay que pasarle a la página de borrado para la eliminación total de dichos datos.

Código de gestión:

El código de gestión del formulario se encarga de recoger los datos del formulario e introducirlos en la base de datos. Para ello, cuando se carga la página, se comprueba si el formulario ha sido enviado o no mediante un código parecido a este:

```
if (isset($HTTP_POST_VARS['submit'])){
    // Codigo de gestión.
} else {
    // Código del formulario.
}
```

Lo primero que realiza el servidor es comprobar campo a campo si los datos que ha recibido contienen información. En caso afirmativo los introduce en diferentes variables para su posterior uso, pero si los campos estaban vacíos asigna el valor NULL a las variables. Está parte se encarga también de ensamblar los 3 campos que forman las fechas en un solo campo que pueda ser entendido por MySQL (AAAA-MM-DD). Una

vez introducida toda la información en variables se comprueba por última vez si estamos ante datos nuevos o editados:

En caso de ser nuevos la llamada a SQL tiene esta forma:

```
INSERT INTO tabla VALUES ('$id','$x1','$x2','$x3' ...  ,NULL)
```

Tabla se corresponde con la tabla en la que queremos introducir los datos y las variables *$xn* continen los datos a introducir. La variable *$id* es una referencia al identificador del curriculum al que están siendo enlazados. Está referencia no existe en caso de *main.php*, ya que dicha página crea la información base a la que se enlaza la demás. El campo que se deja a NULL se corresponde con un identificador único en dicha tabla que será asignado automáticamente por MySQL.

En caso de estar editando, la llamada a SQL tiene esta forma:

```
UPDATE tabla SET X1='$x1', X2='$x2', X3='$x3' ... where tid='$tid'
```

Tabla y las variables $*xn* vuelven a ser lo mismo, pero en este caso los valores *Xn* se corresponden con los nombres de los campos en la tabla. El *tid* referencia al identificador único de dicha fila en la tabla y se corresponde con el campo que dejábamos a NULL al introducir los datos por primera vez.

Una vez realizada la consulta, un mensaje por pantalla nos indicará si esta se ha realizado con éxito o si ha ocurrido algún problema en el proceso.

Información adicional:

Como se comentó previamente, *main.php* y *meritos.php* son ligeramente diferentes. Cuando creamos una nueva entrada a través de *main.php* se crea también la entrada que se crearía a través de *meritos.php* si esta tuviese la opción. Esta variación se debe a que por petición del profesor, se prefirió que la información de méritos se introdujese a través de grandes cajas de texto <textarea> en las que se pudiese escribir libremente todos los que se deseasen en vez de crear una nueva instancia (nueva línea en la tabla) para cada mérito. Por esto mismo, la página *meritos.php* no contiene la parte del código que se utiliza para insertar nueva información, sino que solo admite el modo edición. Al haber una única tabla para los méritos, no se puede eliminar la entrada en la tabla por medio de un botón de borrado como en el resto de campos, sino que se borra automáticamente si se decide eliminar el currículum entero.

En las primeras versiones del código, un problema impedía mostrar los saltos de línea en los textos introducidos <textarea>, utilizados para los méritos y para los comentarios, cuando se consultaba el currículum. Dicho problema se debe a que dichos saltos se introducían en la base de datos con su carácter normal, el cual es ignorado completamente por HTML. Para que se respetasen los saltos de línea se realizo un pequeño cambio antes de introducir la información en las variables que se envían en la consulta. Para ello utilizamos la siguiente función:

```
$destino = ereg_replace("\n","<br/>",$origen)
```

Esta función de php busca en una cadena $origen todas las veces que aparece un salto de línea normal "\n", los convierte en saltos de línea HTML "
" y almacena el resultado en $destino. Cuando estamos en modo edición tenemos que volver a cambiarlo a como estaba originalmente realizando la misma función con diferente orden de parametros para que se muestre en los <textarea>.

4.3 - Selector de curriculums

En esta sección se explica el funcionamiento del archivo select.php. Dicha página está compuesta por un formulario encargado de recoger las opciones del usuario que alterarán la manera en la que se muestra el currículum y otro utilizado para añadir una nueva sección a un currículum existente. Todo ello está apoyado por varias funciones Javascript.

Para poder hacer todo esto, lo primero que deber hacer el usuario es seleccionar de la lista de todos los currículum existentes sobre cual desea trabajar. Como la lista depende de la cantidad de currículums existentes, ha sido necesario programarla para que pueda ser creada dinámicamente. Ello lo conseguimos mediante este código:

```
<select name='idcode' id='idcode'>
<?php
while ($row = mysql_fetch_array($result, MYSQL_NUM)){
        echo "<option value='$row[0]'>$row[1] $row[2]</option>\n";
}
?></select>
```

Este código lee la información que hemos extraído previamente de la base de datos y que hemos almacenado en la variable *$row*. Dicha variable es una matriz en la que se almacena, para cada currículum, el identificador, el nombre y los apellidos. El identificador es el que utiliza el código para trabajar con el currículum, mientras que el nombre y los apellidos son solo para generar la lista.

Al selector de currículums le siguen un menú desplegable para el orden cronológico, una casilla para activar la opción del rango de fechas y las cajas de texto para introducir dichas fechas, dos cajas de opciones de tipo RADIO para activar o desactivar los separadores y los órdenes y varias casillas encajadas en una tabla para seleccionar que partes del currículum se deben mostrar. Todo ello son elementos básicos de HTML, por lo que no necesitan ninguna explicación especial. El único punto remarcable de todo ello es la casilla de *'seleccionar todas'* que se encuentra en el selector de secciones. Dicha casilla está enlazada a un fragmento de código Javascript capaz de activar o desactivar todas las casillas con un solo click.

```
function all_none(){
    casilla = new Array ("ch01", "ch02", "ch03", "ch04", "ch05", "ch06", "ch07", "ch08",
"ch09", "ch10", "ch11", "ch12", "ch13", "ch14", "ch15", "ch16", "ch17", "ch18");
    if (document.getElementById("all").checked){     //Si se marca
        for (x=0;x<18;x=x+1){ document.getElementById(casilla[x]).checked=true;}
    } else {       //Si se desmarca
        for (x=0;x<18;x=x+1){ document.getElementById(casilla[x]).checked=false;}
    }
}
```

Este código copia el estado de la casilla a todas la demás, independientemente de si había alguna previamente marcada. Igualmente, si desactivamos la casilla todas las demás se desactivan.

Además de la mencionada previamente, el bloque anterior esta controlado por otras funciones JavaScript. La función *on_off* se encarga de que no sea posible introducir fechas a menos que esté activada la casilla que habilita el rango de fechas mientras que la función *mirafechas* se encarga de que cuando introducimos las fechas, estas estén en orden correcto y que los valores sean validos.

El formulario acaba con 2 botones, que hacen las veces del botón *submit*, pero como un formulario de HTML solo puede tener uno de estos, el proceso equivalente se realiza mediante Javascript. Ambos botones llaman a la misma función *muestra*, pero enviando un argumento distinto. Según el argumento, el código se encarga de enviar la información del formulario a la página para mostrar la información en el navegador *muestra.php* o a la de mostrarlo en un documento de MS Word *muestraword.php*.

```
function muestra($nu){
        var $z=document.getElementById("idcode");
        // Asigna el id del menú desplegable para saber a que currículum hay que
añadir el campo
        document.getElementById("code").value = $z.options[$z.selectedIndex].value;
        var $formu=document.getElementById("verformu");
        if (mirafechas() == true){
                //Antes de enviar comprueba que el rango de fechas es el correcto
                if ($nu == 1){
                        $formu.action="muestra.php";
                } else {
                        $formu.action="muestraword.php";
                }
                $formu.submit();
        } else {
                return (false);
        }
}
```

El segundo formulario es el encargado de añadir nuevas secciones a un formulario existente, el cual ha sido previamente seleccionado al principio del anterior formulario. Cuando pulsamos el botón que hay junto el menú se llama a una función Javascript que se encarga de recoger el ID del currículum del anterior formulario, asigna al nuevo

formulario un valor para su campo *action*, el cual indica a que página se envía la información y por último realiza la transferencia.

```
function changeAction(){
        var $x=document.getElementById("tipo").value;
        var $y=document.getElementById("nuevo");
        var $z=document.getElementById("idcode");
        // Asigna el id del menu desplegable para saber a que curriculum hay que
añadir el campo
        document.getElementById("codex").value = $z.options[$z.selectedIndex].value;

        switch($x){
          case "1":
                $y.action="formacion.php";
                break;

(Se repite para todos los casos)

          case "18":
                $y.action="gestionid.php";
                break ;

        }
        // Envia el formulario
        $y.submit();
}
```

4.4 - Configuración de órdenes y separadores

En esta sección se explica el funcionamiento del archivo config.php, que se encarga de ayudar al usuario a configurar el orden de los campos y la manera en la que están separados. Una vez más, el archivo consta de 3 secciones principales: Código Javascript, código php para tratar la base de datos y código php del formulario.

Lo primero que encontramos en la parte de Javascript son 20 arrays de cadenas de texto. En ellas se encuentra el texto que se utiliza en la opción de previsualizar el orden antes de hacerlo definitivo. Para previsualizarlo llamamos a la función mostrar. En ella tenemos almacenados los valores para todos los casos, por lo que tenemos que seleccionar cual es el caso que nos interesa con un valor numérico que se le pasa a la función al llamarla. Una vez identificado el caso la función recoge la información que hemos introducido, selecciona el array de textos correspondiente y un valor numérico que indica cuantos elementos tiene como máximo dicho campo. Antes de mostrar como queda, la función llama a otra función que se encarga de comprobar que el texto que hemos introducido es correcto, ya que en caso de no serlo es imposible mostrarlo. Una vez comprobado se separan los elementos de la cadena introducida y muestra los valores que se corresponden con los números introducidos en el array de texto.

Como acabamos de comentar, existe una función que se encarga de comprobar que el texto introducido por el usuario es correcto. La función es la siguiente:

```
function comprobar($ca,$nu){
    if ($ca != ''){ //Si no es vacía
        $ar = $ca.split('-');
        $ta = $ar.length;
        for ($x=0; $x<$ta; $x=$x+1){
            if (isNaN($ar[$x])){
                alert ("Los campos deben contener solo números separados por
guiones");
                return (false);
            }
            if ($ar[$x] > $nu){
                alert ("Uno de los campos tiene un número demasiado alto");
                return (false);
            }
        }
    }
    return (true);
}
```

La variable *$ca* contiene el texto que hemos introducido mientras que la variable *$nu* indica el número máximo que puede haber en dicha cadena. La función comprueba que la cadena no esté vacía, separa sus componentes eliminando los guiones y comprueba para cada uno de sus componentes que tienen el formato correcto (números) y que ninguno sea mayor que el valor máximo dado. Existe además una función *mass_comp* que se dedica a llamar a la función *comprobar* para cada uno de los diferentes campos existentes.

Nada más empezar el código php, la página conecta con la base de datos como vimos en otros archivos. Dicha conexión se usa para ambas secciones de php.

El código php del formulario consta de un gran formulario dentro del cual hay una tabla utilizada para ordenar en pantalla la información en lo que parecen varios pequeños formularios para cada tipo de información que se puede añadir a un currículum. Cada uno de esos formularios falsos esta compuesto de una lista en la que se nombran por orden los diferentes campos en el orden por defecto y su número, un menú desplegable que se utiliza para seleccionar el tipo de separador, una caja de texto para recoger la información del usuario en la que indica como configurar el orden y un botón que llama a la función Javascript *mostrar*. En caso de que hubiesemos cambiado algo en una visita anterior, tanto el menú desplegable como la caja de texto mostrarían dicha información. Podemos ver un ejemplo mostrando el que se encarga de la Experiencia de gestión de I+D:

```
<b> Experiencia de gestión de I+D:</b><br/>
<small>
1- Titulo<br/>
2- Tipo de actividad<br/>
3- Fecha<br/>
4- Comentarios<br/>
<?php
        echo "Separador: <select name='or18'>";
        desplegable ($val, $tex, 2, $or[18]);
        echo "</select><br/>";
        echo "</small><input type='text' name='o17' id='o17' value='$row[18]'>";
        echo “<input type='button' onclick='mostrar(17)' name='ver17'
value='Mostrar'>";
?>
```

Al final del verdadero formulario hay dos botones. El botón *Valores iniciales* devuelve las casillas a como se encontraban antes de haber hecho cualquier modificación en la visita actual a la página. El botón *Aceptar* se encarga de llamar primero a la función *mass_comp* para comprobar que todos los datos introducidos son correctos y luego los envía a la parte del código que se encarga de introducirlos en la base de datos.

La sección de código que se encarga de introducir la información en la base de datos es parecida a la de cuando editábamos entradas del currículum en otras páginas. Al igual que en dicho caso, comprueba uno a uno que los diferentes campos tengan algo escrito y los almacena en variables (o almacena el valor NULL en caso de estar vacíos). Posteriormente actualiza la base de datos con dicha información usando la orden UPDATE. Al contrario que en las entradas del currículum, solo existe una línea en la base de datos con los ordenes del usuario, además de una línea con los ordenes por defecto. La página solo modifica la del usuario, mientras que la por defecto siempre es igual. Se almacenan también los valores para los separadores en variables y se introducen de la misma manera que los órdenes, existiendo también una sola entrada modificable y una por defecto.

4.5 - Mostrar en el navegador y conversión a word

Esta sección explica la programación del archivo *muestra.php* y *muestraword.php*. Ambos archivos son muy parecidos, excepto varios detalles que se explicarán más adelante. Ninguno de estos archivos contiene código Javascript. Ambos archivos se encargan del visionado del currículum, uno en el navegador, con la opción de entrar en el modo edición y otro en un documento con extensión .doc.

La primera diferencia entre el código que muestra los datos en el navegador se encuentra en el código que se introduce al principio del documento *muestraword.php* y obliga al navegador a entregarnos los resultados en un documento llamado *curriculum.doc*. Al ser compatible el editor de textos con el código HTML dicha información puede ser utilizada directamente como si de un documento normal se tratase. El código es el siguiente:

```
<?php
header("Content-Description: File Transfer");
header("Content-Type: application/force-download");
header("Content-Disposition: attachment; filename=curriculum.doc");
?>
```

Una de las opciones del visionado requería que los campos se viesen seguidos, separados con solo un punto. En este caso solo debía mostrarse el contenido del campo, no su titulo, por lo que se han creado varios arrays de texto con los títulos de los campos. Existe además un array de texto vacío, el cual se utiliza en caso de que no queramos ver los títulos.

Tras realizar la conexión con la base de datos, antes de realizar las consultas, encontramos la función *mesname*, la cual se encarga de traducir los meses del valor numérico con el que se almacenan en la base de datos a sus nombres. Era necesario introducir los valores con números para que MySQL sea capaz de realizar búsquedas específicas u ordenar los resultados por fecha.

El siguiente paso es conseguir la información sobre el orden de los campos. La página toma siempre de la base de datos la información por defecto de los órdenes y después comprueba si se pidió que se utilizasen ordenes personalizados. En caso afirmativo, recorre una copia del array de órdenes personalizados que toma de la base de datos, rellenando los campos que estén en blanco con los ordenes por defecto. Esto se hace porque es posible que al usuario solo le interese configurar los órdenes en unos pocos campos, por lo que debe respetarse el orden en los demás.

A los órdenes le siguen los separadores, pero esta vez solo comprueba si tiene que utilizar los normales o los personalizados antes de hacer la consulta MySQL, ya que no se pueden dejar campos en blanco. Para simplificar las tablas, se introdujeron los valores con un simple carácter en cada caso, por lo que es necesario crear una equivalencia para el navegador sepa que mostrar. Esto se realiza con el siguiente código:

```
$linea = "<br/>";
$coma = ". ";
for ($x=1;$x<20;$x=$x+1){
        if ($separadores[$x] == 'l'){$separadores[$x] = $linea;}
        else {$separadores[$x] = $coma;}
}
```

El código que se encarga de mostrar el contenido de los currículums sigue una plantilla con pequeñas diferencias según el tipo de sección. Tiene esta forma:

```
$ord_gen = explode("-", $ordenes[20]);
$first = true;
foreach ($ord_gen as $og){

switch ($og){
   case X:
      if (isset($_POST['chY'])){
         //Realizamos la consulta para obtener el contenido de las táblas
         if ($_POST['ferng']){$query = "SELECT * FROM tabla WHERE (id='$code' AND
fecha<='$fcma' AND fecha>='$fcmi') ORDER BY fecha $orfe";}
         else {
            $query = "SELECT * FROM tabla WHERE id='$code' ORDER BY fecha
$orfe";
         }
         $result = mysql_query ($query) or die ('La consulta falló.' .mysql_error());
         $num_rows = mysql_num_rows ($result);
         //Si hay resultados, los imprime por pantalla
         if ($num_rows != 0){
            // Si no es el primer campo que se imprime muestra un separador
            if ($first == true){
               $first = false;
            } else {
               echo "<hr width=50% align=left>";
            }
            echo '<b><font size=\"4\">Título</font></b><br/>';
            //Bucle para el caso de que haya mas de una entrada asociada al ID
            while ($row = mysql_fetch_array($result)){
               echo "<form id='form_X' action='Webname.php' method='post'
ondblclick='submit()'>";
               // Obtiene el separador
               $sepa = $separadores[X];
               if ($sepa == "<br/>"){
                  $n = $nX;
               } else {
                  $n = $vacio;
               }
               //Obtiene los ordenes
               $ord = explode("-", $ordenes[X]);
               foreach ($ord as $z){
                  switch ($z){
                     Código de campo
                  }
               }
               echo "<input type='hidden' value=$row[Z] name='tid'>";
               echo "</form><br/>";
            }
         }
      }
      break;
   }
}
```

El código empieza separando la cadena de órdenes generales en valores individuales y después, mediante un *switch*, busca la parte de código que se corresponde a dicho valor. Lo primero que hace es comprobar si se pidió mostrar dicha sección y en caso afirmativo realiza una de las dos consultas MySQL, dependiendo de si está activado el

rango de fechas o no y por último comprueba que dicha búsqueda ha dado resultado. Antes de empezar a mostrar las secciones comprueba si es la primera de ese tipo que se está mostrando, en cuyo caso imprime por pantalla una línea separadora.

En algunos casos existe un pequeño código que sirve para hacer clasificaciones dentro de un mismo tipo de sección. Podemos encontrar que se separen las entradas según si el autor principal es el titular del curriculum, si los campos tienen relevancia nacional o internacional, etc. El código, que no se incluye en el mostrado anteriormente, simplemente controla cuando empieza cada una de las clasificaciones para mostrar un título que las separe.

La segunda diferencia entre el código de mostrar en el navegador y en word la encontramos en que cuando mostramos en el navegador, cada entrada está dentro de un formulario que nos permitirá interactuar con la entrada. Si se realiza un doble click sobre la entrada en el navegador nos lleva a la página de edición para dicha entrada en la base de datos. Cuando mostramos en word no es posible realizar esto, por lo que se elimina el formulario.

A continuación se repite para la cadena de órdenes de campos de dicha sección lo que ya hicimos con la de órdenes generales. Separamos los componentes y mediante un *switch* buscamos la sección del código que se corresponde. En dichas secciones de código se repite el uso de tres variables. El valor contenido en *$row[X]* es la información sacada de la base de datos. La variable *$n[Y]* contiene el título de dicho campo en caso de que utilicemos como separadores saltos de línea o un *array* vacío en caso contrario. Por último, la variable *$sepa* es el separador elegido, ya sea salto de línea o punto y espacio.

A continuación se muestran los diferentes tipos de código que podemos encontrar para los diferentes campos:

```
if ($row[X] != NULL) {echo "$n[Y]$row[X]$sepa";}
```

Este tipo de código es el más frecuente y se utiliza cuando mostramos un texto que hemos introducido a mano. El código comprueba que dicho campo contiene información, en cuyo caso la imprime. En algunos casos, cuando el campo era obligatorio introducirlo, no realiza la comprobación de contenido, ya que siempre sería afirmativo. En caso de que se utilice este código para un campo de comentarios se elimina la variable *$n[Y]*, ya que no tienen título. En el caso de los campos de un subproyecto en la sección de proyectos se añade una nueva condición además de la de no estar vacío, ya que debe comprobar que para dicho proyecto está habilitada la opción de subproyectos.

```
echo "$n[Y]";
$di = substr($row[X], 8, 2);
$me = substr($row[X], 5, 2);
$an = substr($row[X], 0, 4);
if ($di != '00'){
    echo "$di de ";
}
if ($me != '00'){
    $me = mesname($me);
    echo "$me de ";
}
echo "$an$sepa";
```

Este código es el utilizado para mostrar las fechas. La variable *$di* contiene el día, *$me* el mes y *$an* el año. En MySQL era necesario que la fecha estuviese unida en un solo campo para que se puedan realizar cálculos, por lo que ahora es necesario separarla en los 3 valores que la forman. Dicha separación se realiza con la función de PHP *substr*. Como el único valor obligatorio era el año es necesario comprobar si los valores de mes y día son diferentes de cero antes de mostrarlos.

```
echo "$n[Y]";
if ($row[X] == 'A') {echo "Texto A";}
elseif ($row[X] == 'B') {echo "Texto B";}
elseif ($row[X] == 'C') {echo "Texto C";}
....
echo "$sepa";
```

Este código se utiliza para mostrar la información que introdujimos mediante un menú desplegable con opciones predeterminadas. El código comprueba cual es el valor almacenado en la base de datos y muestra el texto correspondiente.

4.6 - Borrado de entradas

Esta sección explica la programación del archivo *borrar.php*. Este archivo, que tampoco contiene código Javascript, es el encargado de eliminar secciones de currículums o currículums enteros según se le llame desde la página *main.php* o desde otra. Cuando se llama a este archivo le llegan dos variables: *id*, que contiene el identificador único de dicha entrada en la base de datos y *tabla*, que indica que tipo de tabla es la que se utiliza, como por ejemplo *main*, *gestionid*, etc.

El archivo funciona de manera diferente si la tabla es *main* u otra diferente. En caso de no ser *main*, la página ejecuta la orden MySQL que para eliminar de dicha tabla el elemento con ese identificador, el cual se corresponderá con el identificador propio de la entrada en la tabla, no con el que referencia al archivo principal con el cual se relaciona dicha entrada.

```
"DELETE FROM $tabla WHERE tid='$id'"
```

En caso de ser una tabla *main* no es suficiente borrar solo dicha tabla, sino que además debe borrar todas las entradas en todas las tablas que dependen de ella. En este caso, el valor id representa el identificador de la tabla main, el cual todas las entradas de otras tablas tienen como referencia para saber de que tabla provienen. MySQL realizará una busqueda en cada tabla eliminando todas las entradas que referencien a dicho identificador.

```
"DELETE FROM $tabla WHERE id='$id'"
```

Como podemos observar, ahora busca en el campo id, no en el tid.

5 - Ideas para versiones más avanzadas

En esta sección del documento se incluyen diferentes ideas para ampliar el proyecto que quedan fuera de los objetivos iniciales del mismo, pero que pueden resultar interesantes.

5.1 - Versión distribuida

La idea consiste en reprogramar el código de manera que la aplicación pueda ser utilizada por diferentes personas que acceden a un servidor central en lugar de tener cada uno que utilizar una maquina para ello. La versión actual de la aplicación ya lo permite, pero al estar fuera de los objetivos no se han implementado los mecanismos de control ni de seguridad que son necesarios en las aplicaciones de este tipo.

5.2 - Buscador

La idea consiste en la implementación de un buscador que lea todas las bases de datos en busca de algún tipo de información, como por ejemplo en cuantas publicaciones o proyectos ha participado una persona.

5.3 - Crear secciones

La idea más complicada, ya que requiere alterar todo el código o incluso re-escribirlo entero, tal vez con lenguajes diferentes. Esta opción permitiría al usuario poder alterar dinámicamente las plantillas para cada sección o incluso crear secciones nuevas desde cero, lo que conlleva la creación de sus respectivas bases de datos y la introducción de cambios en el código que se encarga de mostrar el formulario. Por supuesto, al poder crear secciones también debe ser posible destruirlas.
En esta versión de la aplicación se han introducido campos de comentarios en cada una de las secciones, por lo que el usuario es capaz de introducir cualquier información extra que desee sin tener que alterar toda la estructura de la aplicación.

5.4 - Mayor personalización de formatos

Esta idea le permite al usuario tener mayor control en la manera en la que se muestran los currículos, pudiendo alterar los formatos de letra, la manera de mostrarse, hacer que el texto tome alguna forma determinada, etc. La opción de cambiar los ordenes y mostrar diferentes separadores es solo un primer paso a lo que sería esta idea. Actualmente, si queremos modificar el currículum en profundidad es necesario hacerlo a mano con el documento que obtenemos en formato *.doc*.

6 - Manual de usuario

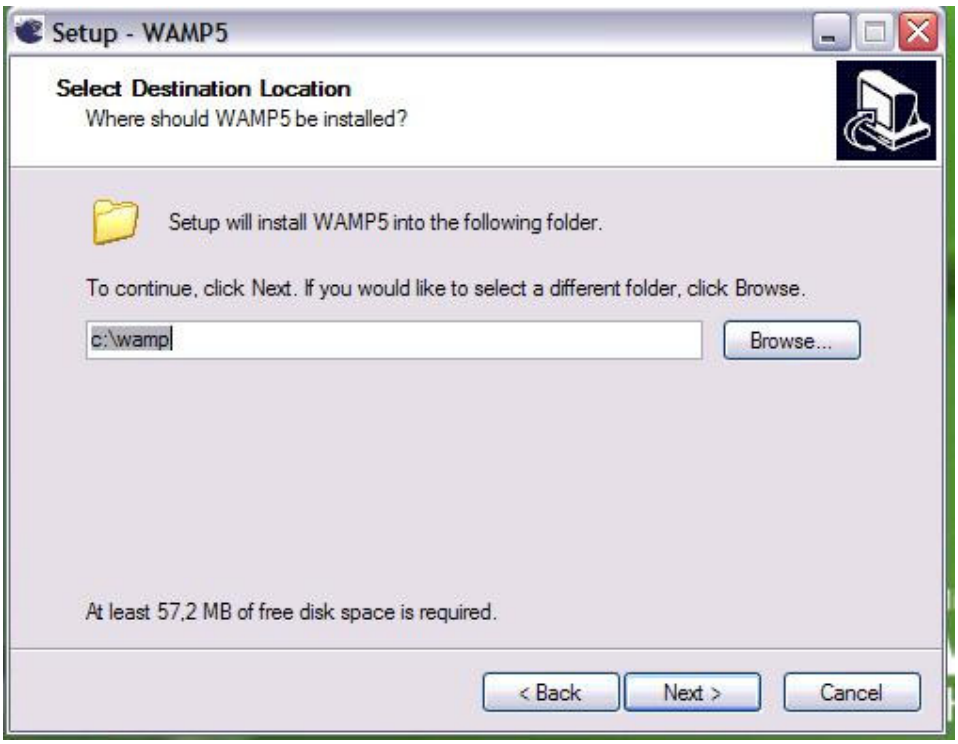
6.1 - Instalación en Windows

Esta instalación supone que la maquina no dispone de servidor Apache con PHP y MySQL. En caso de haber uno instalado solo necesita añadir los archivos php que se encuentran en archivos.exe y las bases de datos de DB.sql en los lugares correctos.

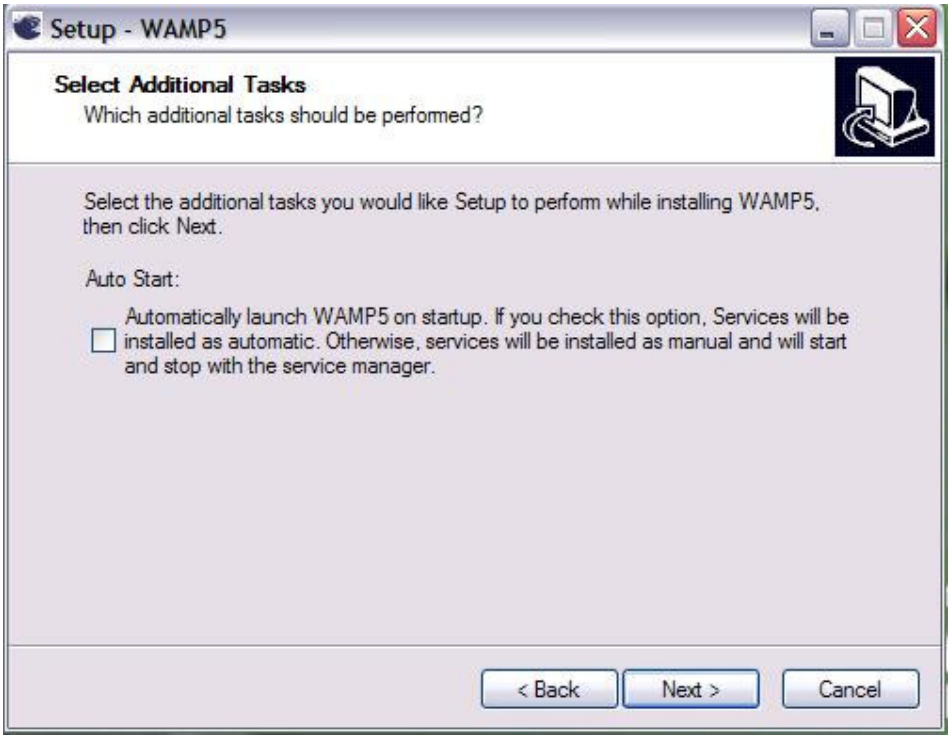
Intalación de WAMP

1- Se ejecuta el archivo wamp_install.exe

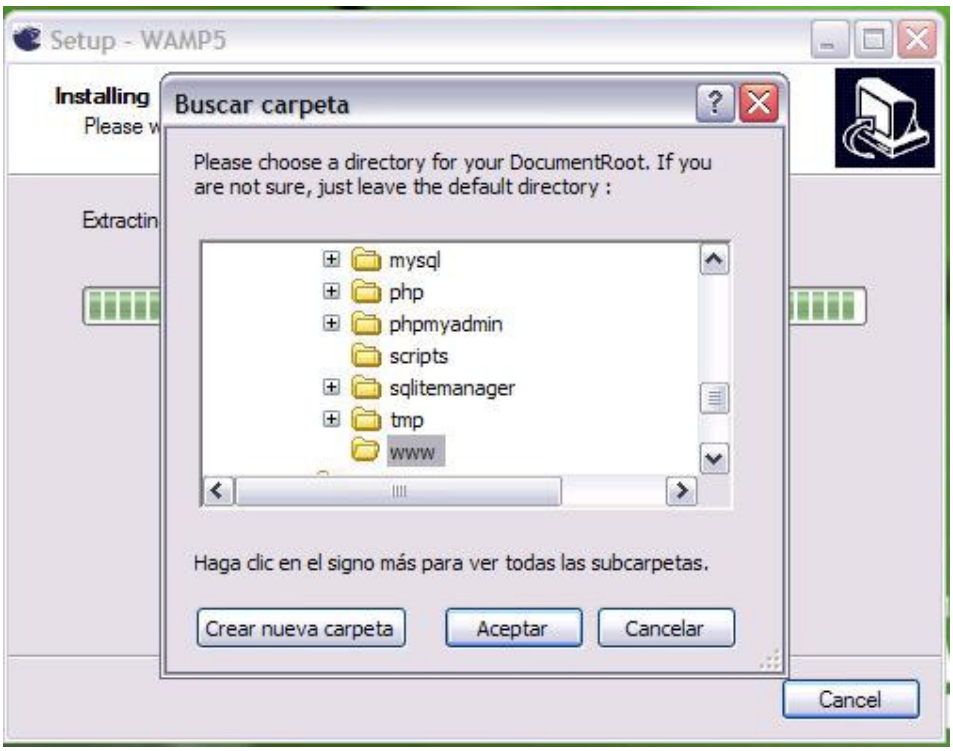
2- Se elige el lugar de instalación deseado (por defecto c:\wamp)



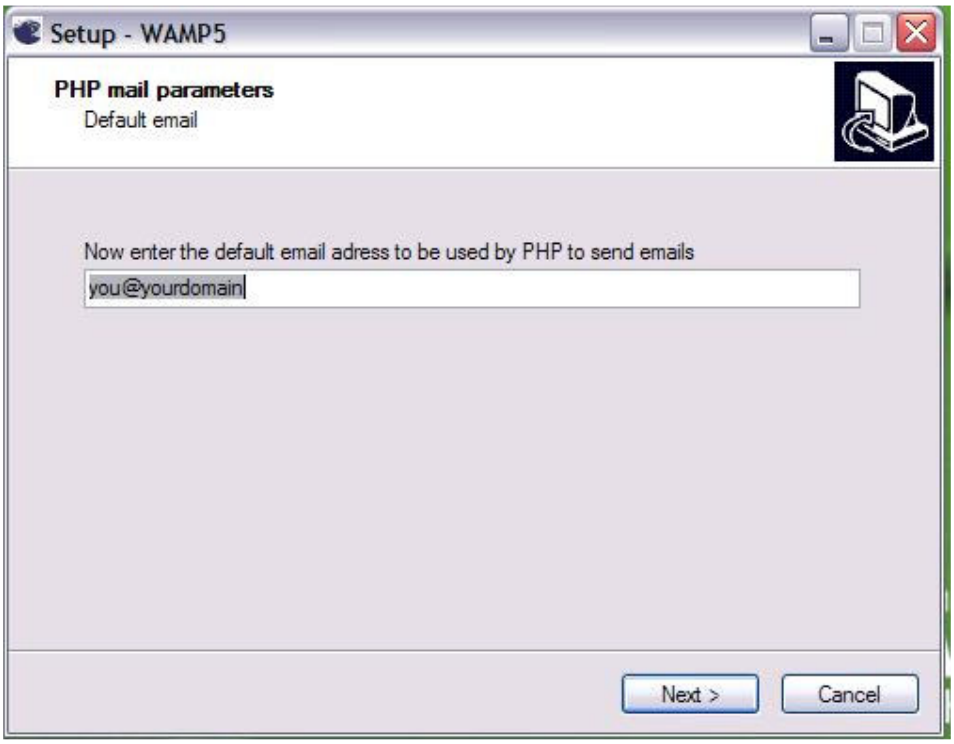
3- Cuando pregunte se le indica que no inicie automáticamente el wamp al conectar el ordenador a menos que se esté seguro de querer hacerlo.



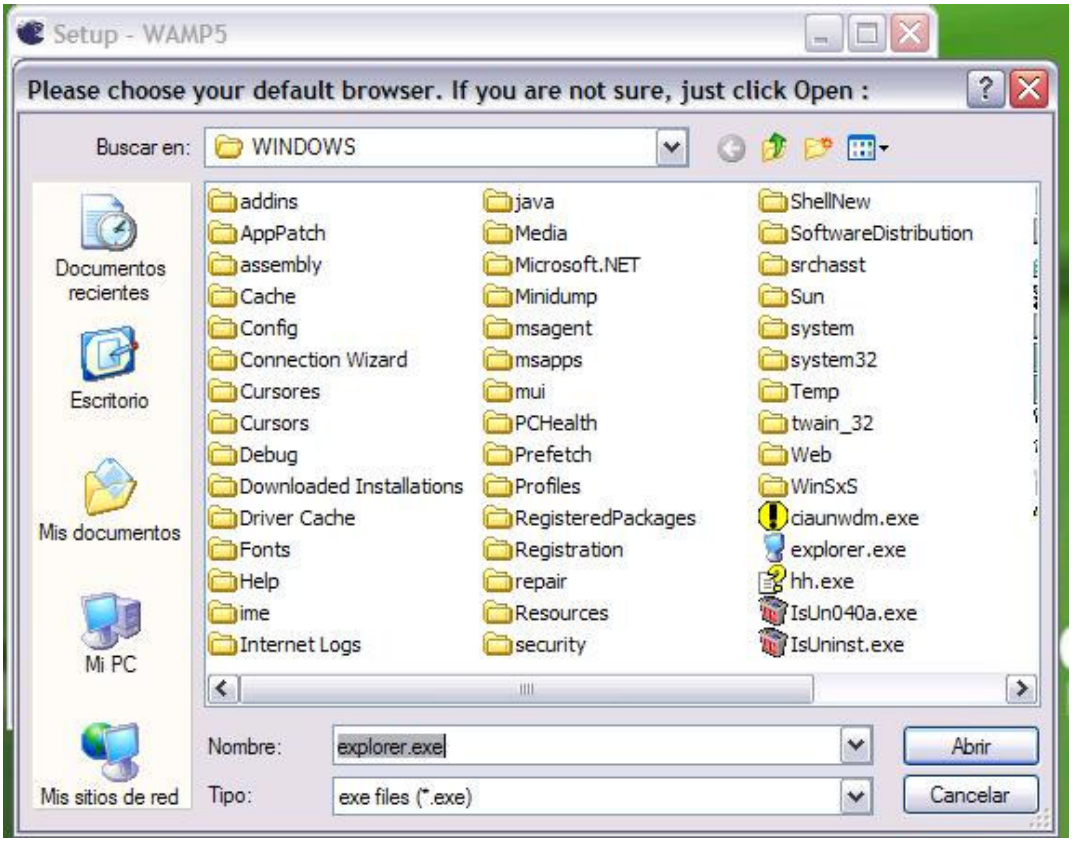
4- Cuando acabe de instalar pregunta cual será el directorio raíz del servidor para los archivos servidos por el servidor. (Por defecto c:\wamp\www)



5- El instalador realiza dos preguntas sobre el servidor de correo que no tienen importancia, así que hay que dejar la respuesta por defecto.



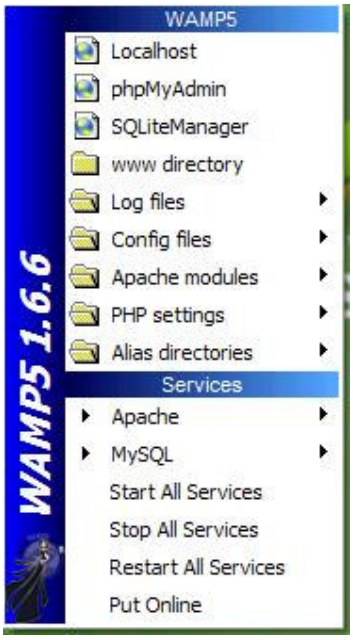
6- Se selecciona un navegador o se deja el por defecto (windows Explorer)



7- Cuando pregunte si se desea ejecutar wamp al acabar hay que marcar la casilla para que lo haga.

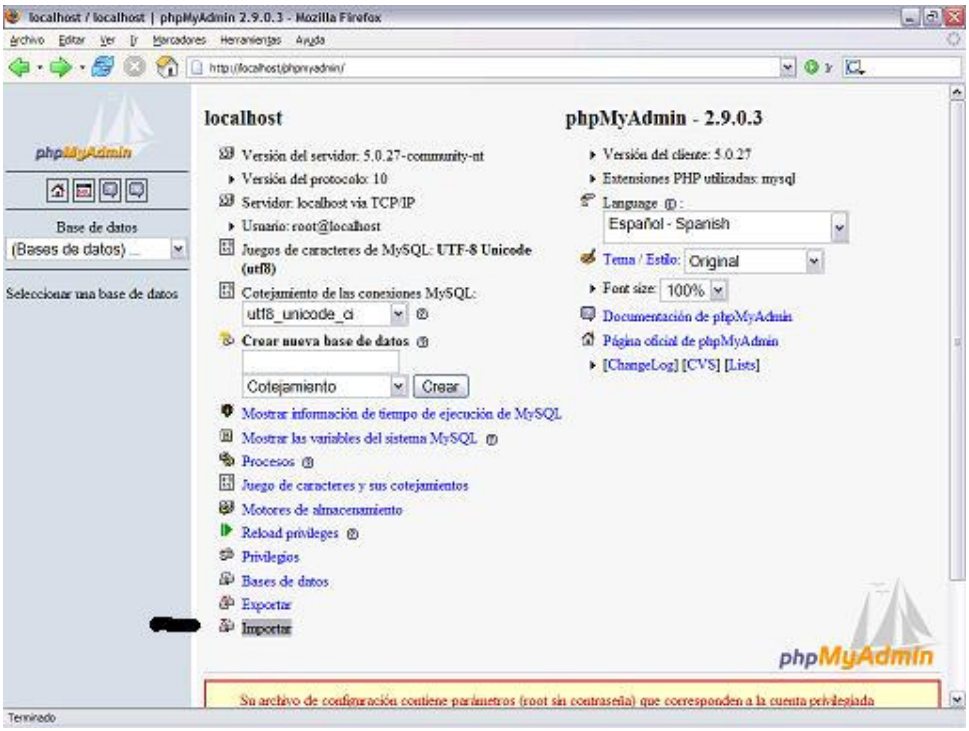
Añadir ficheros del proyecto.

1- En la esquina inferior derecha, junto el reloj de windows, aparece un semicírculo. Si se pulsa una vez con el botón izquierdo aparece el menú del servidor wamp. Pulse sobre *'Start All Services'* para activar el servidor.



2- En el mismo menú pulse sobre *'www directory'* y descomprima el fichero archivos.zip directamente en la carpeta que se abre.

3- Por último elija *'phpMyAdmin'*. Se abrirá una página de configuración de MySQL en el navegador. Marque la opción importar. Con examinar busque el archivo DB.sql y seleccione continuar. Esto crea las bases de datos.



6.2 - Ejecución de la aplicación

1- Tras todos los pasos de la instalación el servidor ya está activado y listo para funcionar. Para acceder a la aplicación hagalo mediante la URL http://localhost/inicio.php o mediante el archivo 'DB Curriculos'.

2- Para apagar el servidor, en el menú seleccione *'Stop All Services'*. Luego haga click con el botón derecho sobre el semicírculo y seleccione *'Exit'*.

3- Para volver a activarlo ejecute el archivo wampserver.exe que estará donde haya decidido instalar el programa (por defecto c:\wamp\) y eleja otra vez *'Start All Services'* en el menú del servidor.

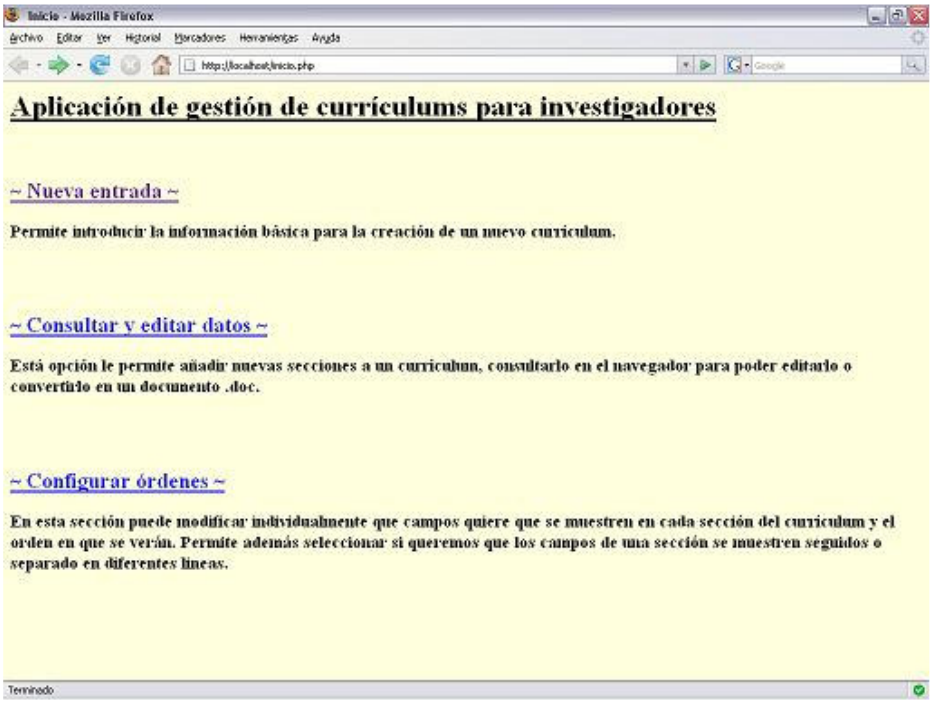
4- Una vez activado el servidor, para acceder a la aplicación, puede hacerlo mediante la URL http://localhost/inicio.php o mediante el archivo 'DB Curriculos'.

Opcional - Por defecto el servidor empieza offline. Permite utilizar todos los servicios como localhost, pero la maquina no permite conexiones externas (se verá un pequeño candado sobre el semicírculo). Con la opción *'Put Online'* del menú del servidor se puede permitir a personas ajenas que accedan al servidor. Será necesario proporcionarles una dirección URL.

Iconos:

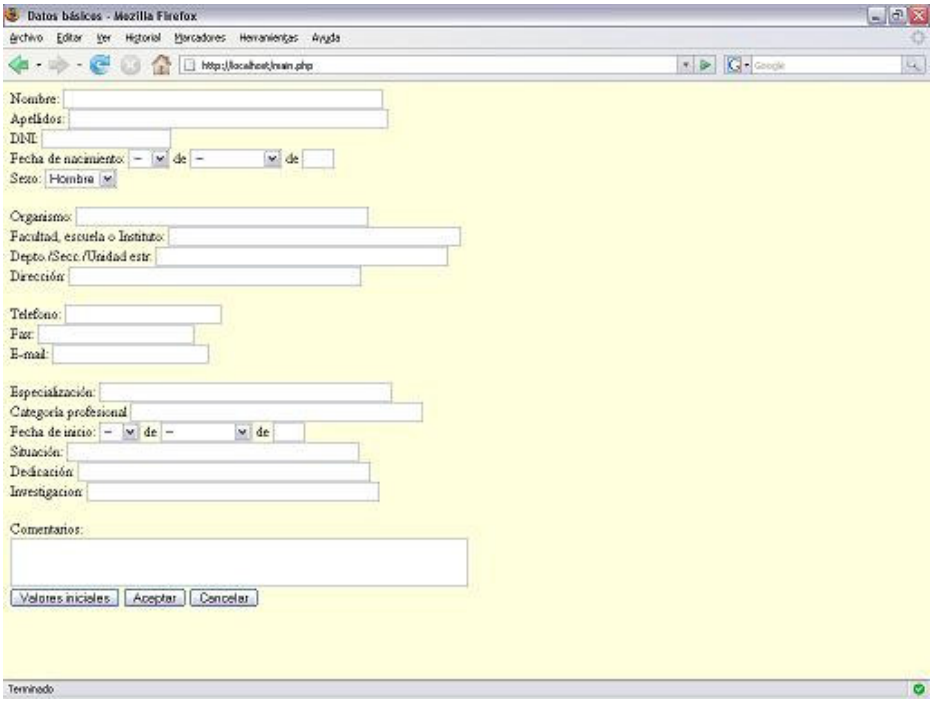
Encendido	Apagado	Offline
		

6.3 - Crear un nuevo currículo



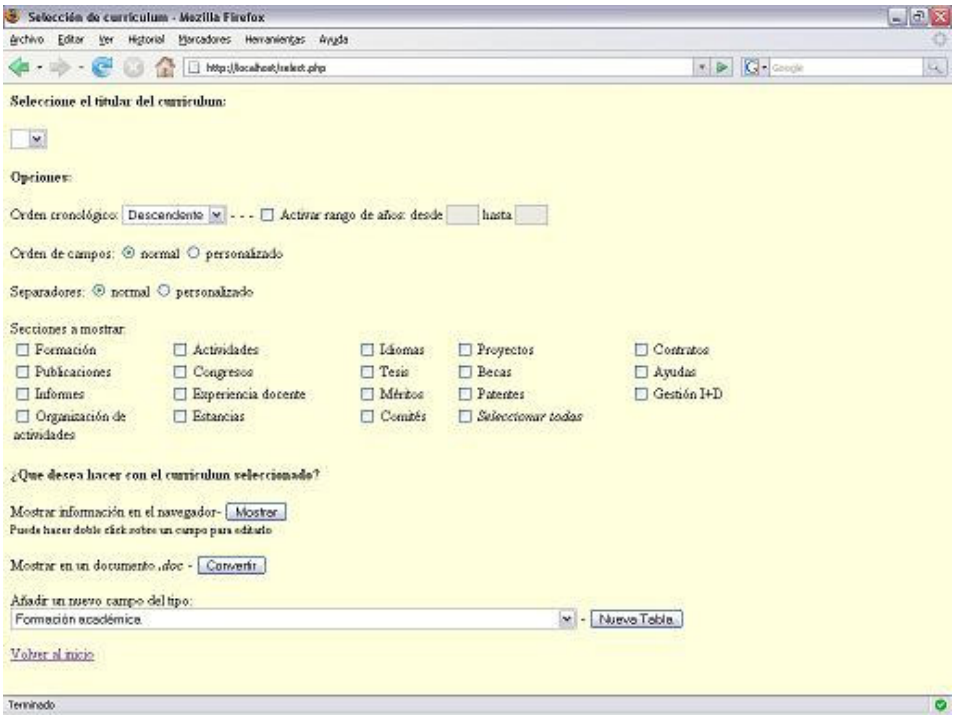
Página inicial de la aplicación

1- Lo primero es crear una entrada con los datos personales básicos. Para ello pulse sobre *'Nueva entrada'* en la página principal. Una vez allí rellene los campos. Es necesario rellenar los campos nombre, apellidos y las fechas.



Página para la creación de un nuevo curriculum

2- Si desea añadir al currículum nuevos campos como tesis dirigidas, participación en proyectos, etc. debe elegir en el menú principal *'Consultar y editar datos'*. En la nueva página primero seleccione quien es el titular del currículum al que desea añadir el campo y después elija del menú desplegable de la parte inferior de la página que tipo de campo quiere añadir. Pulse el botón 'Nueva tabla' y rellene la información para dicha sección. Repita la operación para cada campo que quiera añadir.

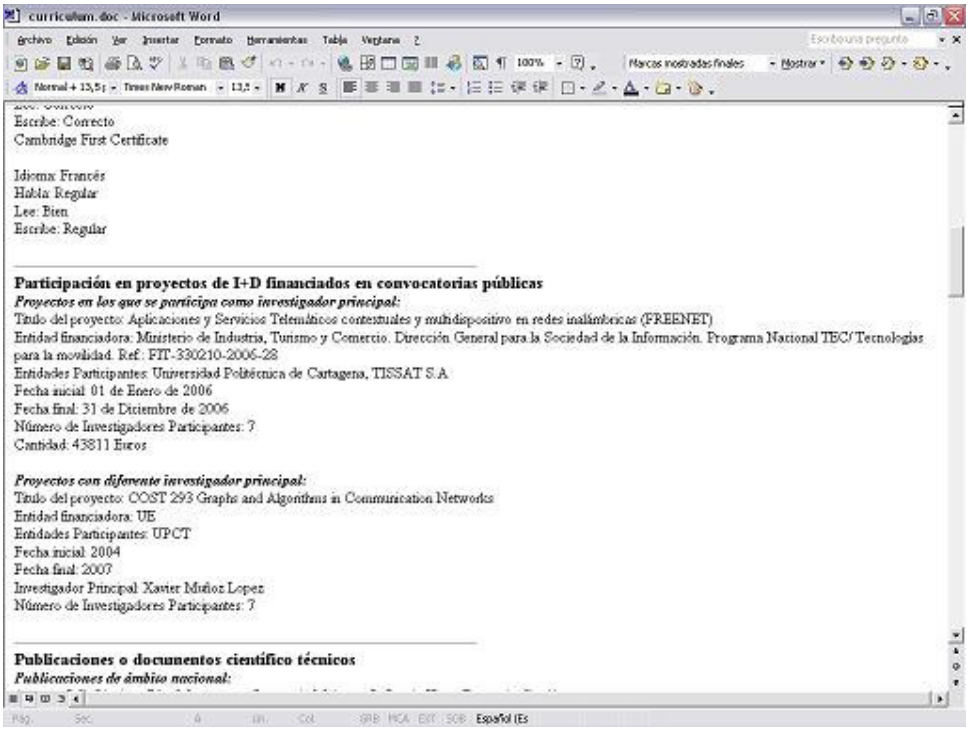


Página de selección de curriculums

6.4 - Mostrar currículum / Convertirlo a formato Word

1- Para mostrar un currículum eleja la opción 'Consultar y editar datos'. Allí seleccione quien es el titular del currículum, que campos se quieren mostrar y las opciones de mostrado. Cuando esté todo listo pulse *'Mostrar'* para que se muestre en el navegador.

2- Si se quiere el currículum en un documento del formato Word (curriculum.doc) tiene que seguir los mismos pasos que en el apartado 1, pero al final pulse sobre el botón *'Convertir'*.



3- Opciones que puede aplicar al currículum:

- *Orden cronológico*: Los campos se muestran del más antiguo al más moderno (ascendente) o a la inversa (descendente). A pesar del orden cronológico, algunas secciones se separan y ordenan con más prioridad automáticamente por otros criterios como el tipo o el ámbito. Dentro de esas sub-selecciones es donde se aplica el orden cronológico.
- *Rango de años*: Solo se muestran los campos que hayan ocurrido entre las fechas introducidas. Es necesario introducir ambas fechas.
- *Orden de campos*: Si elije normal, se mostraran todos los campos en el orden predeterminado. En caso de elegir personalizado solo se mostrarán los campos que hayamos activado y en el orden que haya establecido en la página de configuración.
- *Separadores*: Si lo deja normal la información esta separada en nuevas líneas, mientras que si utiliza la opción personalizado podemos configurarlo para que algunos campos muestren su información toda seguida separada por puntos.
- *Secciones a mostrar*: Marque los campos que desea que se muestren. Si no marca ninguno solo se mostrará la información principal. Aunque marque un campo para que se muestre, si este currículum no tiene asociado ningún campo de ese tipo no se mostrará.

6.5 - Editar y borrar

1- Lo primero que debe hacer es mostrar el currículum en el navegador siguiendo los pasos del apartado anterior. Si quiere editar los datos de alguna sección en particular es importante que la seleccione en las *'Secciones a mostrar'*.

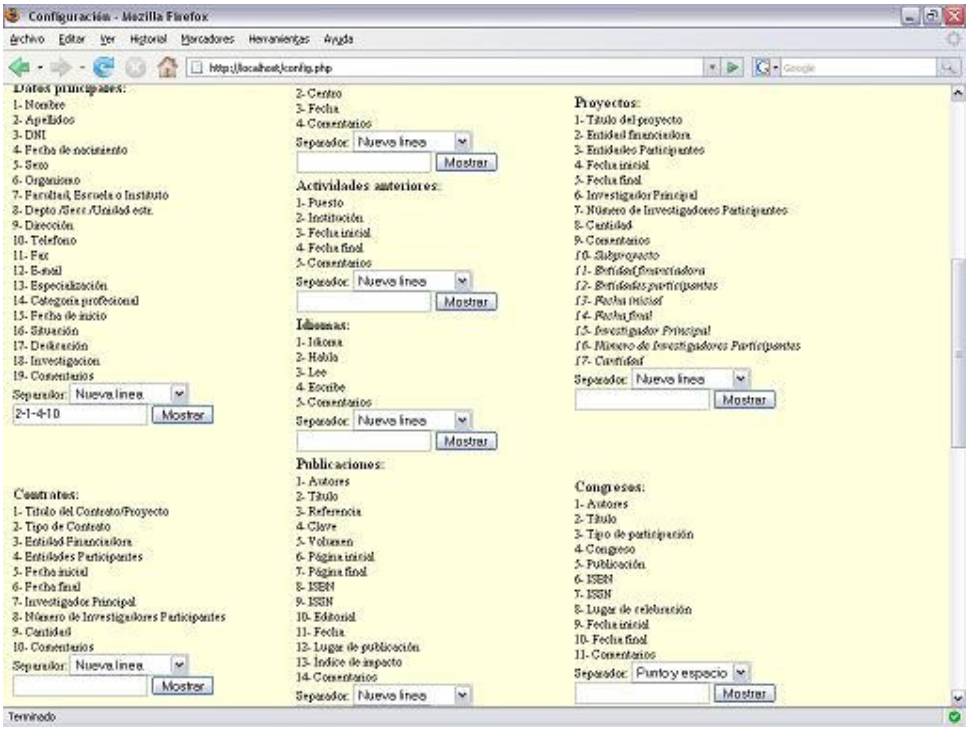
2- Una vez se muestre, haga doble click sobre la sección que quiere editar para entrar al modo edición. Es muy parecido a cuando se creó el campo por primera vez, pero la información que se había insertado anteriormente ya se encuentra escrita. Una vez haya acabado los cambios pulse sobre *'Aceptar'*. Si pulsa sobre *'Valores iniciales'* los datos volverán a estar como en un principio. Una vez haya aceptado no se pueden recuperar los anteriores.

3- Si en vez de editar el campo queremos eliminarlo pulsamos sobre el botón *'Borrar'*.

4- Si quiere eliminar por completo un currículum y todos sus campos debe ir a la página de edición de la información principal y pulsar *'Borrar'*.

6.6 - Configuración

1- Dentro de la página de 'Configurar órdenes' puede ver una muestra del orden por defecto en que se muestran la información en cada campo y puede configurar un orden personalizado.



Página de configuración

2- Si quiere alterar el orden en el que se muestra la información tiene que introducir en la casilla correspondiente el orden deseado. El orden debe introducirse escribiendo los números de cada dato, que puede obtener de la lista que hay sobre la caja de texto, separados solamente por un guión, sin dejar ningún espacio. Por ejemplo 4-7-1-2-3. Si en la lista no escribe el número de algún dato, este será ignorado cuando lo muestre.

3- Cuando esté introducida la sucesión de números, puede pulsar sobre el botón *'Mostrar'* para ver una muestra de como quedarían los campos.

4- Al igual que puede configurar los ordenes también se pueden configurar los separadores. El separador por defecto es 'Nueva línea' y muestra los datos cada uno en una línea diferente. Si elije la opción 'Punto y espacio' los datos se mostrarán todos seguidos, separados por puntos. Esta manera permite hacer los campos más compactos.

5- Una vez haya acabado de configurar los separadores y órdenes personalizados guarde los cambios con el botón *'Aceptar'* en lo más bajo de la página.

6.7 - Información para otros sistemas operativos

Aunque está versión de la aplicación está pensada para usarse bajo sistemas Windows, es posible utilizarla desde otros sistemas operativos. El programa Wamp es específico para Windows, pero se pueden conseguir servidores web para cualquier sistema operativo. Por ejemplo, para sistemas Linux se puede obtener Apache de la web oficial http://httpd.apache.org/ . Será necesario instalar un servidor HTTP, configurarlo e instalar complementos para PHP y MySQL. No es el objetivo de este proyecto enseñar a configurar un servidor en otros sistemas operativos, así que se deja a manos del usuario en caso de que este deseé ejecutar la aplicación en un sistema diferente.

A pesar de lo dicho, al haber sido diseñada la aplicación como una página web, será posible acceder a ella como si de una página normal se tratase desde cualquier sistema operativo que cuente con un navegador siempre y cuando el servidor esté conectado a una red, permita el acceso a los archivos y se proporcione una dirección URL.

7 - Referencias

7.1 - Herramientas utilizadas

- WAMP Server. http://www.wampserver.com/en/
- PHP Designer 2006. http://www.mpsoftware.dk/phpdesigner.php
- Mozilla Firefox. http://www.mozilla-europe.org/es/products/firefox/
- Expansión FireBug para FireFox http://www.getfirebug.com/
- PhpMyAdmin. http://www.phpmyadmin.net/

7.2 - Bibliografía

- **Guia de aprendizaje MySQL** – Larry Ullman – Prentice Hall – ISBN: 84-205-3843-4.

7.3 - Páginas web de interés

- Referencia de MySQL: http://dev.mysql.com/doc/refman/5.0/es/index.html
- Manual de PHP: http://es2.php.net/manual/es/index.php
- Especificación HTML 4.01 http://html.conclase.net/w3c/html401-es/cover.html
- Wikipedia. http://en.wikipedia.org - http://es.wikipedia.org
- Tutoriales JavaScript. http://www.w3schools.com/js/default.asp