# SISTEMAS DE GESTIÓN DOCUMENTAL

### **PRÁCTICAS DE PLWEB**

## PASARELA DE UNA BASE DE DATOS DOCUMENTAL AL WORLD WIDE WEB

La herramienta Personal Librarian nos ha permitido crear bases de datos documentales en un entorno Windows. Esta herramienta permite trabajar en modo cliente servidor, y posibilita de este modo el acceso remoto a una base de datos documental. No obstante, esta no es la mejor forma de trabajar hoy en día.

Cuando se precisa proporcionar acceso remoto a una aplicación, en nuestro caso, una base de datos documental, se suelen utilizar tecnologías relacionadas con el web, de modo que la aplicación se puede ejecutar en un servidor a través de ciertos programas (cgi, jsp, asp, ...) que hacen interactuar al usuario con la aplicación que se quiere hacer disponible al público en modo de trabajo remoto. Un ejemplo gráfico sería el siguiente:



En la práctica que se propone utilizaremos una herramienta llamada PLWeb que permite hacer disponible en el web una base de datos documental creada con la herramienta Personal Librarian. De este modo, utilizaremos la base de datos creada en la sesión anterior de prácticas para hacerla accesible a través del web.

### Introducción

La herramienta PLWeb es básicamente una herramienta de gestión documental combinada con una pasarela al web. Esto significa que con el PLWeb se pueden crear bases de datos compatibles con las creadas con el Personal Librarian, si bien el funcionamiento del PLWeb es bastante más complejo. Por otro lado, la herramienta proporciona unos programas que, situados en un servidor web, permiten trabajar sobre las bases de datos creadas con esta misma aplicación o con el Personal Librarian.

La herramienta se puede utilizar sobre una plataforma Windows, que utilice como servidor web el Internet Information Server. La instalación no es una tarea compleja, pues está bien documentada y no requiere conceptos de administración avanzados, aunque no es el objetivo de esta práctica.

No obstante, la administración de la herramienta sí que es una tarea compleja, pues son multitud de detalles y opciones los que hay que tener en cuenta para mantener el sistema funcionando de forma correcta. También es cierto que una vez configurada la herramienta inicialmente, son pocas las tareas de mantenimiento que hay que realizar, siempre y cuando no se pretenda variar la visualización de las páginas web o las opciones de creación y visualización de bases de datos definidas en la configuración inicial de la herramienta.

Para esta sesión de prácticas contamos con una instalación del PLWeb preparada para que los alumnos introduzcan sus bases de datos en el sistema y realicen las modificaciones oportunas para que la base de datos aparezca en un servidor web. En este enunciado práctico se proporcionará un 'recetario' de los pasos a seguir para conseguir este objetivo.

El alumno no debe perder de vista que crear una pasarela de una base de datos documental al web a través del PLWeb, a pesar de parecer una tarea sencilla, conlleva muchas opciones que no están consideradas en el enunciado que aquí se proporciona, y que el mantenimiento de dicha pasarela puede llegar a ser una tarea muy costosa.

#### **Primeros pasos**

Para entender como funciona la herramienta, podemos acceder a una base de datos documental a través del web en la dirección:

#### http://pokemon.uv.es:80/plweb/index.html

Esta primera página contiene un enlace de entrada al SGD. En nuestra demostración, el acceso no tiene ninguna restricción, pero el sistema se podría configurar para permitir accesos restringidos según las necesidades.

El enlace de entrada ya nos permite observar cuál va a ser el funcionamiento del sistema, es decir, se puede observar que el enlace apunta a /plweb-cgi/fastweb.exe, que es el programa que se va a encargar de hacer todas las acciones que el usuario solicite y de resolver las consultas que formule el usuario. El programa fastweb.cgi (una pasarela, en realidad) acepta el uso de parámetros, de modo que la respuesta del programa variará según los parámetros que le pasemos. En este primer ejemplo, se le pasa el parámetro

TemplateName y el valor views.tmpl. Esto significa que se está solicitando las distintas vistas que ofrece el servidor para poder consultar las bases de datos que contiene.

Por tanto, todas las páginas web que implementen funcionalidades ofrecidas por la base de datos documentales deberán construirse con un formulario (HTML FORM) que llame a la aplicación fastweb.exe (ACTION="/plweb-cgi/fastweb.exe"), y que contenga una serie de elementos de formulario con los nombres y valores adecuados para que la pasarela pueda entender las tareas que se le solicitan.

Al entrar en la página de las vistas, si sólo existiese una vista disponible en el sistema, se saltaría directamente a la ventana de búsquedas, pero en nuestro caso se han utilizado dos vistas para mostrar el funcionamiento de la aplicación y por tanto accedemos a una página de selección de vistas, antes de empezar las búsquedas. Las vistas de que disponemos son la vista por defecto que proporciona PLWeb, y una vista de ejemplo creada para ilustrar las posibilidades de la herramienta.

Desde cada vista se tendrá acceso a distintas bases de datos, y además se podrán utilizar distintos aspectos visuales de las páginas. De este modo, se puede utilizar el mismo servidor y el mismo motor para mantener bases de datos de distintas organizaciones, con diferentes bases de datos y aspectos visuales para la consulta de las mismas.

En las siguientes secciones aprenderemos a modificar los archivos de configuración para definir vistas adecuadas a las necesidades, utilizar bases de datos creadas con Personal Librarian y modificar los aspectos visuales de las pantallas de búsqueda y recuperación de resultados.

#### <u>Tareas a realizar</u>

El objetivo de la práctica es poner la base de datos creada en las sesiones anteriores con el Personal Librarian en el web a través del PLWeb. A modo de resumen, las tareas que se deben realizar en esta práctica son las siguientes:

- 1. Llevar la base de datos al servidor donde reside el PLWeb y colocarla en el directorio adecuado.
- 2. Modificar el fichero de configuración para añadir vistas para cada grupo de trabajo en la página de entrada al servidor.
- 3. Crear templates (plantillas) para cada vista (una por cada grupo de trabajo) de modo que se obtenga un aspecto visual propio a la base de datos de cada grupo.

A continuación se detalla la forma de realizar cada una de estas tareas.

#### Primera prueba: añadir una base de datos a una vista creada

Para facilitar la toma de contacto con la aplicación, se propone inicialmente una tarea sencilla: añadir la base de datos propia de cada grupo a una vista que ya haya sido creada con anterioridad. Para ello, utilizaremos la vista del profesor que tiene por nombre 'ramon'.

Seguiremos, por tanto, los siguientes pasos:

- 1. Crear un directorio en el servidor que contendrá la base de datos de cada grupo. El directorio llevará por nombre la denominación de cada grupo (sgd1, sgd2, ...) y se situará dentro del directorio C:\Archivos de programa\plweb\databases
- 2. Copiar la base de datos de cada grupo dentro del directorio creado (incluyendo todos los archivos: .acc, .adf, .def, .pls, texto fuente, tesauro, etc.). Para realizar esta tarea será necesario utilizar el ftp.
- 3. Modificar el archivo C:\Archivos de programa\plweb\etc\plweb.conf para añadir, en la sección de bases de datos [Databases], los datos correspondientes a la base de datos de cada grupo: NOMBRE, RUTA, y DESCRIPCIÓN, siguiendo el formato de los ejemplos que existen en dicho fichero: name=NOMBRE, location=RUTA, description="DESCRIPCIÓN". (Nota: puesto que es posible que varios grupos trabajen sobre el mismo fichero, es conveniente asegurarse de que los cambios realizados se han almacenado correctamente en el fichero, por ejemplo abriéndolo de nuevo una vez cerrado con los cambios realizados).
- 4. Añadir la base de datos a la vista que se desea que tenga acceso para consulta de la misma. Para ello utilizaremos la vista 'ramon' (definida ya en la sección [Views] del fichero plweb.conf). Será necesario modificar el fichero C:\Archivos de programa\plweb\views\ramon\view.conf y añadir el nombre de la base de datos que se ha utilizado en el punto anterior en la entrada "databases", en formato de lista de bases de datos separados por una coma. (Nota: puesto que es posible que varios grupos trabajen sobre el mismo fichero, es conveniente asegurarse de que los cambios realizados se han almacenado correctamente en el fichero, por ejemplo abriéndolo de nuevo una vez cerrado con los cambios realizados).
- 5. Parar y arrancar el servicio plwebd (<u>hablad con el profesor</u>, puesto que se necesitan permisos especiales de administrador para realizar esta tarea).
- 6. Probar que se tiene acceso a la base de datos desde la vista correspondiente. Esto se consigue entrando en la vista 'ramon' y viendo que la base de datos aparece como un elemento más sobre el que se pueden realizar búsquedas.

Una vez se haya conseguido que la base de datos aparezca como opción de búsqueda en el servidor web, y que se puedan hacer búsquedas sobre la misma, se puede pasar a la siguiente tarea.

### **Los Templates**

La gestión de las vistas definidas en el servidor se realiza a través de los templates, así que es necesario dedicar un tiempo a aprender como funcionan los templates antes de enfrentarnos a la creación de una vista particular.

Un template no es más que una plantilla HTML que se utiliza para dar formato a la información que se puede proporcionar al sistema y/o que el sistema proporciona al

usuario. En la plantilla se definen, por ejemplo, los tipos de letra, los colores e imágenes de las páginas, y los contenidos que van a tener, combinando etiquetas HTML con un lenguaje de macros que proporciona el PLWeb.

Con el lenguaje de macros se pueden incluir cosas como las bases de datos que hay disponibles, los resultados de una consulta, los términos relacionados con un término de una consulta, etc. Con esto se consigue crear páginas HTML con el aspecto que se define en los templates, y con la funcionalidad que queramos añadir a través del lenguaje de macros.

Aprender el lenguaje de macros sería muy costoso y llevaría bastante tiempo, así que para definir los templates cada grupo utilizará un conjunto de templates de ejemplo que se han definido en la vista 'ramon' y que simplifican la tarea de crear una vista particular para cada grupo de trabajo. En cualquier caso, el sistema dispone de un conjunto de templates de ejemplo que son bastante completos y que se encuentran en el directorio C:\Archivos de programa\plweb\views\sample-view\templates, y que tienen la siguiente estructura:



Hay dos tipos de plantillas: las normales, y las de implementación. Se distinguen por el nombre: las segundas terminan por \_IMPL.TMPL. Básicamente, las primeras plantillas incluyen a las segundas con el objetivo de separar elementos de visualización genéricos de específicos. Por ejemplo, en VIEWS.TMPL se incluyen la cabecera y el pie de página de la página web que se va a mostrar, y en medio se incluye la plantilla VIEWS\_IMPL.TMPL, mientras que VIEWS\_IMPL.TMPL contiene la lógica que permite consultar las vistas definidas en el servidor y las muestra en la página web, y además contiene las instrucciones a realizar (HTML FORMS) cuando se selecciona la vista.

Como se puede apreciar, cuando se accede a la herramienta (a la primera página que ofrece el servidor web), la primera plantilla que se carga es views.tmpl, la cual muestra las vistas disponibles y permite seleccionar una para las siguientes acciones. Una vez seleccionada la vista, se pasa a la plantilla de búsquedas. Esta pantalla permite realizar distintos tipos de búsquedas: búsquedas normales, términos relacionados, términos del diccionario y búsqueda difusa.

Según el tipo de búsqueda que se haya elegido, la plantilla PREHIT.TMPL decidirá que plantilla cargar, para mostrar los resultados de cada tipo de búsqueda. La búsqueda normal utilizará, por ejemplo, la plantilla HITLIST.TMPL.

Por último, después de mostrar el hitlist, aún queda un proceso de decisión que consiste en determinar si el documento indexado es de tipo texto o de tipo HTML, con lo que la visualización del documento se realizará con una plantilla distinta, según sea el caso.

Existe además una plantilla para la gestión de errores, que en el gráfico está separada de las demás, pero que se incluye en todas las plantillas de implementación para mostrar información que se ha generado cuando ocurre un error. Par ver como funciona esta plantilla, se puede realizar una búsqueda en el servidor sin especificar la base de datos, o sin especificar ningún término de búsqueda.

#### Creación de la vista particular

Una vez hemos comprendido qué son las plantillas y cómo se utilizan en el PLWeb, vamos a crear y definir una vista para nuestra base de datos, incluyendo las plantillas necesarias para una visualización personalizada.

Para facilitar el proceso, utilizaremos como base para las modificaciones las plantillas de la vista 'ramon', que son una simplificación de las plantillas de ejemplo que hemos visto en el apartado anterior, aunque mantienen la misma estructura, es decir, que aportan una funcionalidad completa sobre la base de datos.

Las tareas que se deben realizar son las siguientes:

- 1. Crear un directorio con el nombre de la vista que se quiere utilizar (por ejemplo, sgd1) en el directorio C:\Archivos de programa\plweb\views.
- 2. Copiar los contenidos del directorio C:\Archivos de programa\plweb\views\ramon al directorio que se ha creado, es decir, se copia la estructura, los ficheros de configuración y los templates para modificarlos con posterioridad.

- 3. Modificar en el nuevo directorio creado el fichero view.conf, considerando las siguientes entradas:
  - a. Name: nombre de la vista (una sola palabra).
  - b. Description: descripción que se desea para la vista, que se muestra en la página web cuando se permite seleccionar la vista de trabajo.
  - c. Databases: lista de bases de datos que se desea que estén disponibles para la vista.
  - d. Mantener el parámetro PARSETEMPLATES=FALSE. Esto hace que se pueda visualizar dinámicamente los cambios hechos en los templates sin tener que parar y arrancar el servidor en cada cambio.
  - e. Mantener el resto de parámetros como están.
- 4. Los otros dos ficheros creados dentro del directorio contienen información de log (log.conf) y de usuarios con permisos para acceder a la vista (viewusrs.conf), y se deben mantener tal y como han sido copiados de la vista 'ramon'.
- 5. Modificar el archivo C:\Archivos de programa\plweb\etc\plweb.conf para añadir, en la sección de vistas [Views], los datos correspondientes a la vista que se desea añadir a la base de datos, que consiste básicamente en poner en una línea nueva el nombre del directorio creado en el paso 1. Tras finalizar este proceso el profesor deberá reiniciar el servidor para reflejar las actualizaciones realizadas: avisad al profesor para que lo haga. (Nota: puesto que es posible que varios grupos trabajen sobre el mismo fichero, es conveniente asegurarse de que los cambios realizados se han almacenado correctamente en el fichero, por ejemplo abriéndolo de nuevo una vez cerrado con los cambios realizados).
- 6. Modificar los templates para dar el aspecto visual deseado a las páginas del servidor.

#### **Resultados**

Puesto que la práctica se ha realizado directamente sobre un servidor, la práctica finalizará cuando se haya conseguido publicar completamente la base de datos documental en el web, se haya definido una vista particular para acceder a dicha base de datos, y se hayan modificado los templates para proporcionar un aspecto visual personalizado para cada base de datos.

La evaluación de la práctica se realizará sobre el resultado obtenido, es decir, el profesor accederá al servidor para comprobar el trabajo que cada grupo haya realizado, y evaluará los resultados obtenidos. Por tanto, los grupos deberán asegurarse de que su trabajo está perfectamente integrado en el sistema y funciona con normalidad en el plazo establecido por el profesor para la realización de la práctica.

### Anexo: Ejemplos de código de macros para implementar funciones del sistema

Función	Uso de una vista en las plantillas
Código	<input name="view" type="hidden" value="{\$VIEW}"/>
Comentario	Cuando se acceden a las plantillas, es necesario indicar al PLWeb en qué vista se está trabajando. De lo contrario, el PLWeb podría decidir utilizar una plantilla de otra vista que se llama igual (de hecho, si no se dice nada, utilizará la vista por defecto). Una vez seleccionada la vista en la primera página, esa vista hay que incluirla en todos los formularios (HTML FORMS) en la forma en la que se indica aquí.

Función	Tratamiento de errores
Código	{\$IF: variable=error, operator=EQ, value=TRUE} {\$INCLUDE: file=error_impl.tmpl}
	{\$ENDIF}
Comentario	Se suele incluir al final de la plantilla, y muestra la página de errores si se ha producido algún error.

Función	Vistas disponibles
Código	{\$IF: variable=NUMVIEWS, operator=EQ, value=0}
e e	[tratamiento de la situación: no hay vistas disponibles]
	{\$ELSE}
	{\$VIEWLIST}
	<form action="http:fastweb.exe" meted="post"></form>
	<input name="TemplateName" type="hidden" value="search.tmpl"/>
	<input name="view" type="radio" value="{\$VIEWNAME}"/> {\$VIEWDESC}
	<input name="submit" type="submit" value="Continuar"/>
	{\$VIEWLIST_END}
Comentario	Se comprueba si existen vistas disponibles. Si no hay vistas, se muestra un
	mensaje de información, y si hay vistas, se crea un formulario que cargará la
	plantilla search.tmpl, y que muestra en forma de radio buttons las descripciones
	de las vistas disponibles. Es necesario poner un input de tipo submit con el
	atributo name=submit.

Función	Enlaces a distintas secciones
Código	{\$TOSEARCHFORM: target=search.tmpl}Búsquedas{\$EOL}
	{\$PREVLIST: target=hitlist.tmpl}Anterior{\$EOL}
	{\$NEXTLIST: target=hitlist.tmpl}Siguiente{\$EOL}
Comentario	Cada línea es un enlace a una sección distinta del servidor, y por orden de aparición son: Página de búsquedas, lista de hits anterior, lista de hits siguiente

Función	Ejecución de una búsqueda de varios tipos
Código	<form action="http:fastweb.exe" enctype="x-www-form-&lt;br&gt;encoded" method="POST"></form>
	<input name="TemplateName" type="hidden" value="prehit.tmpl"/> <input name="view" type="hidden" value="{\$VIEW}"/>
	<pre>{\$DBLIST} <input operator="CONTAINS," type="checkbox" value="\$DBNAME}checked{\$ENDIF}&lt;/th" variable="SELECTEDDBS," {\$if:=""/></pre>
	<input name="query" size="45" type="text" value="{\$QUERY}"/> <input name="query_rule" type="hidden" value="(\$query)"/> <input name="operator" type="hidden" value="OR"/>
	<input name="simplesearch" type="SUBMIT" value="Buscar"/>
	<input name="concept" type="SUBMIT" value="Busqueda conceptual"/>
	<input name="relate" type="SUBMIT" value="Terminos relacionados"/>
	<input name="fuzzy" type="SUBMIT" value="Busqueda Fuzzy"/>
	<input name="dictionary" type="SUBMIT" value="Diccionario"/>
Comentario	Dentro del form, hay que indicar la plantilla que se cargará como resultado de la búsqueda y la vista que se va a utilizar (los dos primeros input hidden). Es importante usar una plantilla que distinga los resultados según el tipo de
	lista de bases de datos donde se puede hacer la búsqueda (son checkbox
	porque se puede seleccionar más de una base de datos para buscar). Luego hay
	que situar un campo de texto, con nombre "query" que contendrá la consulta a
	último, cada tipo de submit permite realizar una búsqueda distinta, en base al
	texto introducido en el campo de búsqueda. Cada boton de submit se distingue
	por el nombre, que es lo que indica al servidor el tipo de busqueda a realizar.

Función	Seleccionar número de resultados para una consulta
Código	<select name="numresults"></select>
	{\$IF: variable=MAXITEMS, operator=EQ, value=10} <option selected=""></option>
	10 {\$ELSE} <option> 10 {\$ENDIF}</option>
	{\$IF: variable=MAXITEMS, operator=EQ, value=25} <option selected=""></option>
	25 {\$ELSE}
	{\$IF: variable=MAXITEMS, operator=EQ, value=0} <option< th=""></option<>
	selected > 25 {\$ELSE} < option > 25 {\$ENDIF}
	{\$ENDIF}
	{\$IF: variable=MAXITEMS, operator=EQ, value=50} <option selected=""></option>
	50 {\$ELSE} <option> 50 {\$ENDIF}</option>
	{\$IF: variable=MAXITEMS, operator=EQ, value=100} <option selected=""></option>
	100 {\$ELSE} <option> 100{\$ENDIF}</option>
Comentario	Se puede determinar el número de registros que se visualizarán por cada página
	tras una consulta. Para ello, solo hay que incluir este select dentro del
	formulario de busqueda. Se pueden modificar los valores predeterminados.

Función	Seleccionar tipo de ordenación para una búsqueda
Código	<select name="sorting"> {\$IF: variable=SORTING, operator=EQ, value=byrelevance}<option selected value="byrelevance"&gt; Relevancia {\$ELSE}<option value="byrelevance"&gt; Relevancia {\$ENDIF} {\$IF: variable=SORTING, operator=EO, value=none}<option selected<="" th=""></option></option </option </select>
	value="none"> Sin orden {\$ELSE} <option value="none"> Sin orden {\$ENDIF} .</option>
Comentario	Se puede determinar el tipo de ordenación con el que se visualizarán los registros recuperados tras una consulta. Para ello, sólo hay que incluir este select dentro del formulario de búsqueda.

Función	Lista de términos relacionados
Código	{\$RELATELIST} {\$IF: variable=QUERY, operator=CONTAINS, value=\$RELATETERM} <font color="red"> {\$RELATETERM}, </font>
	{\$ELSE} {\$DICTTERM},
	{\$ENDIF} {\$RELATELIST_END}
Comentario	Esta macro ofrece una lista de términos relacionados con los de la consulta, marcando en rojo los términos que estaban en la consulta.

Función	Lista de términos fuzzy
Código	{\$FUZZYLIST}
Ũ	{\$IF: variable=QUERY, operator=CONTAINS, value=\$FUZZYTERM}
	<font color="red"> {\$FUZZYTERM}, </font>
	{\$ELSE}
	{\$DICTTERM},
	{\$ENDIF}
	{\$FUZZYLIST_END}
Comentario	Esta macro ofrece una lista de términos semejantes a los de la consulta,
	marcando en rojo los términos que estaban en la consulta.

Función	Lista de términos del diccionario
Código	{\$DICTLIST}
Ū	{\$IF: variable=QUERY, operator=CONTAINS, value=\$DICTTERM}
	<font color="red"> {\$DICTTERM}, </font>
	{\$ELSE}
	{\$DICTTERM},
	{\$ENDIF}
	{\$DICTLIST_END}
Comentario	Esta macro ofrece una lista de términos que se encuentran en el diccionario en
	posiciones cercanas a los términos usados en la consulta, marcando en rojo los
	términos que estaban en la consulta.

Función	Datos útiles de una consulta
Código	{\$HITSFOUND}
	{\$QUERY}
	{\$FIRSTHITRANK}
	{\$LASTHITRANK}
Comentario	Datos de interés para mostrar información de resumen de una consulta. En orden son: el número de hits para la consulta, la consulta, el número de orden del primer hit mostrado, el número de orden del último hit mostrado

Función	Lista de hits para una consulta
Código	{\$HITLIST}
	{\$SCORE: size=5}
	{\$DUCLINK: target=predoc.tmpl}
	{\$TITLE: lines=1, default=**** SIN TITUED **** }{\$EOL} {\$TITLE: lines=2, firstline=2}
	{\$DBNAME: size=12}
	{\$DOCSIZE: size=5}
	{\$HITLIST_END}
Comentario	Esta macro permite listar todos los hits obtenidos para una consulta, hasta alcanzar el límite especificado por el número de resultados en una consulta. En concreto se muestra, por este orden: el valor del ranking para el hit, la plantilla que se debe usar para visualizar el documento completo, titulo del hit, resumen del hit, base de datos en la que se encontró el documento, y tamaño del documento.

Función	Inclusión de variables ocultas
Código	{\$HIDDENVALS}
Comentario	La aplicación fastweb.exe necesita tener información sobre el estado de las acciones del usuario y sus preferencias. Por ello, utiliza una serie de campos 'ocultos' (como la vista que estamos utilizando) que indican los parámetros sin que el usuario sea consciente de ello. Para incluir los parámetros ocultos que provengan de una acción anterior (por ejemplo, la vista que estoy utilizando) es suficiente con incluir esta macro en la página, dentro del form correspondiente. Es importante realizar esta inclusión si no se desea perder información sobre las acciones que está haciendo el usuario y sus preferencias.

Lo que aquí se proporciona es una serie de ejemplos que permitirán mejorar el rendimiento del alumno en la creación de plantillas. En cualquier caso, el manual de administrador del PLWeb contiene en el anexo C una lista completa de macros con su explicación.