Capítulo 2. Tecnologías relacionadas con conversión de interfaces

En el presente capítulo se explicaran las tecnologías relacionadas al ambiente de conversión a desarrollar, así como los lenguajes manejados en su implementación.

2.1 Convertidores

En términos sencillos, un convertidor toma el texto de una página fuente, es decir, el HTML, en el caso de las páginas *web,* y lo reformatea a un texto en un lenguaje objetivo, tal como XML, WML, VoiceXML, UML, RTF, PDF; como se muestra en la Figura 2.1.

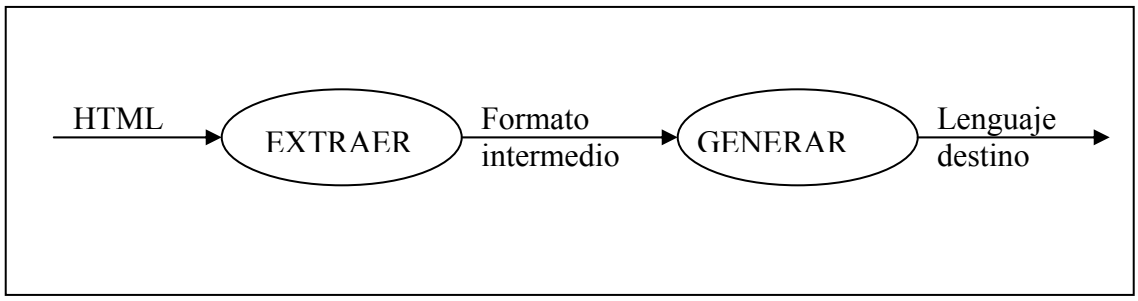

Figura 2.1 Proceso de conversión

El convertidor lleva a cabo la conversión de datos con formato a datos puros, es decir, sin estructura, por lo que se puede aplicar el formato que se desee para poder desplegar esta información en dispositivos diversos, tales como teléfonos móviles y PDAs.

Los convertidores también se conocen como transcodificadores. En otras palabras, la transcodificación es un método para traducir un tipo de código a otro tipo distinto. Los transcodificadores existen para convertir HTML, RTF, XML, entre otros lenguajes, a un

formato diferente, en el caso de la aplicación que describe este documento, a un formato adecuado para su visualización en dispositivos móviles.

2.2 Lenguajes relacionados con la aplicación

PoPS maneja tres lenguajes: inicia con XML, el lenguaje en el cual están hechas las interfaces y que se convierten a WML y XHTML que son los otros 2 lenguajes con los que trabaja enfocándose a los dispositivos móviles. Como se describirá posteriormente, el ambiente de conversión EditMos también tiene a XML como el lenguaje intermedio, siendo HTML el lenguaje de las interfaces de entrada y WML y XHTML los lenguajes de las interfaces de salida. A continuación se describe brevemente cada uno de los lenguajes involucrados en el proyecto.

2.2.1 XML (eXtensible Markup Language)

XML es un lenguaje de marcado similar a HTML, diseñado para describir datos. Mientras que en HTML las etiquetas son predefinidas, en XML el usuario define sus propias etiquetas, por lo cual XML es extensible. XML utiliza un DTD (*Document Type Definition*) o un *esquema* para describir los datos a través de una secuencia de reglas que permiten su manejo. XML está diseñado para ser auto-descriptivo y es un lenguaje de *marcado* donde todo debe ser etiquetado correctamente, lo que genera como resultado *documentos bien formados* [Marchal 1999].

2.2.2 XHTML (eXtensible Hipertext Markup Language)

XHTML fue creado para sustituir HTML (*Hypertext Markup Language*) y es casi idéntico a HTML 4.01 aunque más estricto que HTML al manejar una versión más estructurada. XHTML es una aplicación XML [Martin 2000].

2.2.3 WAP (Wireless Application Protocol) / WML (Wireless Markup Language)

El protocolo WAP es un estándar para servicios de información en terminales móviles como los teléfonos móviles. WML es el lenguaje usado para crear páginas en un navegador WAP, o navegador de los teléfonos celulares. WML está basado en XML y divide el contenido de las páginas en bloques ya que el tamaño reducido de las pantallas de los teléfonos celulares no permite el despliegue completo de la información [Howard 2001].

2.2.4 XSL

XSL es el Lenguaje de Hojas de Estilo Extensible (*Extensible StyleSheet Language*) y también es una aplicación de XML. Mientras XML se concentra en la estructura de la información y no en su apariencia, XSL se encarga de darle formato a los documentos en XML. En la hoja de estilo XSL se especifica el estilo, distribución y paginación de un documento XML o de un archivo de datos. Está formado por plantillas que se comparan con los elementos del documento fuente para producir el documento salida [Martin et al. 2000].

2.2.5 XSLT

XSLT (*XSL Transfomations*) es el lenguaje para transformar documentos XML a otros documentos XML. Diseñado para usarse como parte de XSL, que es un lenguaje de hoja de estilo. Aunado a XSLT, XSL incluye un vocabulario XML para especificar el formato. XSL especifica el estilo de un documento XML utilizando XSLT para describir como se transforma en otro documento XML.

La especificación define la sintaxis y la semántica del lenguaje XSLT. Una transformación expresada en XSLT describe reglas para transformar un árbol fuente (conformado por los elementos del documento original) a un árbol resultado (definido por el usuario) [Kay 2001].

Para objeto de transformaciones de formatos, se compara cada elemento en el documento entrada con la plantilla y se producen los elementos equivalentes.

2.3 Sistemas existentes de conversión de interfaces

El desarrollo de las herramientas de conversión está basado en diversas metodologías. Las herramientas realizan la conversión tomando como entrada datos estructurados en un lenguaje determinado y generan como salida datos sin formato, por lo que los autores de la conversión pueden escoger el formato que se va a aplicar.

Las herramientas de conversión toman menos tiempo de programación ya que permiten la extensibilidad a otros lenguajes, permitiendo obtener un mismo documento en varios lenguajes diferentes entre sí. Las desventajas de estas herramientas son, entre otros, los

errores de conversión; si la página a convertir tiene elementos que el convertidor no reconoce, la conversión falla. Otra desventaja es el costo, ya que una herramienta de conversión comercial es demasiado cara, como las herramientas ofrecidas por compañías como Innodata Isogen [Innodata 2004]. Además, no todas las páginas basadas en audio o gráficos, se pueden convertir a XML, C#, Java ó Visual Basic que son los lenguajes de los dispositivos móviles.

Hablando específicamente de convertidores comerciales ya desarrollados, hay 2 tipos: (1) los convertidores automatizados, como lo son HTML *Tidy* [Raggett 2003], WF4 [Jabbour 2002] y *Norfolk* [Jabbour 2002], que extraen todo el contenido posible de la página en HTML y lo convierten a XML y WML respectivamente; y (2) los convertidores configurables, como *Portal-to-go* [Khare1999], en los que se le indica qué parte del documento se va a convertir, manejándolas a través de un ambiente gráfico que permite realizar la conversión fácil y rápidamente.

Algunos convertidores son gratuitos, otros bajo licencia, pero el punto fundamental es poder concentrar estas herramientas en una sola, analizando entre las existentes qué convertidor es el mejor a integrar, aprovechando las tecnologías ya desarrolladas. Todo con un fin: convertir información estructurada en un lenguaje a otro. Los que se mencionan se enfocan en que el lenguaje final permita su acceso desde dispositivos móviles.

Como se describe en los capítulos siguientes, EditMos integra diversas herramientas existentes en un solo ambiente. Estas herramientas fueron elegidas después de revisar cuidadosamente los desarrollos actuales que podrían contribuir de manera más efectiva a lograr los objetivos del proyecto. Un resultado intermedio del proyecto fue un reporte

técnico que sumariza los trabajos relacionados con conversión de interfaces. El reporte técnico se incluye en el Apéndice A.

2.3.1 Contenido adecuado para la conversión

En general, proveer pequeñas cantidades de información basada en texto, es la característica principal que hace que un sitio *web* sea adecuado para ser convertido a otro lenguaje [Arehart 2000]. Después, necesitamos que la fuente, es decir, la interfaz en lenguaje HTML, sea HTML bien formado. Es decir, que las etiquetas estén bien cerradas y que el documento tengo una estructura coherente (título, contenido, imagen, bordes) porque los navegadores interpretan a su manera el lenguaje HTML. Esto nos lleva a incongruencias en cómo son desplegadas las páginas que encontramos en la red. Por ello la mejor manera para corregir esto es el uso de un "validador" de HTML. Enfocándonos en esto, primero consideraremos que el documento que será convertido está bien estructurado, después de haberlo validado con una herramienta adecuada. Para el proyecto descrito a lo largo de este documento, utilizaremos HTML Tidy, una herramienta gratuita y con la posibilidad de reuso de código, que limpia documentos HTML.

2.3.2 HTML Tidy : Limpiando el HTML

HTML Tidy es una herramienta gratuita, creada por Dave Raggett. Tidy trabaja en etiquetas generadas por editores especializados de HTML y herramientas de conversión. Tidy es capaz de solucionar una gran cantidad de problemas. Cada error encontrado se lista junto con la línea y la columna en la que se encuentra, permitiéndole al usuario ver donde está el problema en las etiquetas [Raggett 2003].

El código fuente continúa disponible bajo una licencia de código abierto, lo cual es un factor importante en la elección de esta herramienta para dar un nuevo formato al documento original.

Tidy presenta las siguientes cualidades:

- Detecta y corrige etiquetas terminales mal cerradas o faltantes.
- Corrige etiquetas finales en un orden equivocado.
- Arregla problemas con énfasis en el encabezado.
- Agrega "/" faltantes al final de las etiquetas.
- Perfecciona listas poniendo las etiquetas faltantes.
- Agrega comillas faltantes en los valores.
- Reporta los atributos desconocidos.
- Detecta las etiquetas con ">" faltantes.

Muchas herramientas generan HTML con exceso de etiquetas tales como las etiquetas "FONT", "BR" y "CENTER". Tidy limpia el documento reemplazando estas etiquetas con propiedades de estilo y reglas usando CSS (*Cascading Style Sheets*). Esto simplifica la lectura del lenguaje de marcado más simple y reduce el tamaño del archivo.

Otra ventaja, tal vez la más sobresaliente, es que al utilizar Tidy, se le puede "enseñar" a la herramienta nuevas etiquetas declarándolas en el archivo de configuración.

Otro punto a favor es que Tidy maneja una librería (API), permitiendo su llamado en aplicaciones propias, habiendo dos versiones: una librería escrita en C y otra en JAVA, los lenguajes más factibles a utilizar debido a su portabilidad y eficiencia. Se puede integrar la herramienta a un sistema existente o que se quiere crear, ya sea integrando la herramienta Tidy o haciendo llamados de las clases en la librería (API) de Tidy. HTML Tidy empezó

como una herramienta de línea de comando, pero hay versiones gráficas disponibles para plataformas Windows y MacOS, así como el API en C o JAVA, la librería a la que se le puede hacer llamados a las clases que la conforman.

Además de depurar documentos en HTML, Tidy permite la conversión al formato XML y XHTML, lo cual se menciona con detalle en la sección 2.4.3.

Habiendo ya escogido una herramienta previa a la conversión con el fin de tener documentos lo mejor estructurados posibles, se puede empezar a considerar qué herramientas utilizar para las conversiones respectivas.

2.3.3 Conversión de HTML a XML/XHTML con HTML Tidy

Durante algún tiempo, el *World Wide Web Consortium* (W3C) ha estado monitoreando los aspectos relacionados con proveer contenido a diferentes tipos de clientes sin tener que hacer diferentes versiones de cada página. Bajo el estandarte de *Mobile Access Group*, muchos de los nuevos estándares, propuestas y borradores de trabajo – como lo son el *eXtensible Markup Language* (XML), que permite la separación de contenido y presentación, las hojas de estilo y el "*Resource Description FrameWork*" (RDF) – están fusionándose con el fin de proveer una plataforma coherente que soportará múltiples clientes de diferentes tipos [Arehart 2000].

El primer paso en el proceso de convertir las páginas en HTML es estructurar el lenguaje de la interfaz y limpiarlo para que un analizador XSLT (transformación XSL): ya sea el *Document Object Model* (DOM) o el *Simple API for XML* (SAX) pueda trabajar con los documentos [Raggett 2003].

Se escogió HTML Tidy ya que permite la fácil conversión de los documentos por medio de dos clases principales que se explicarán a detalle en la sección 4.

Un ejemplo de cómo funciona Tidy se muestra a continuación, donde el Listado 2.1 muestra el documento original y el Listado 2.2 muestra el documento final después de ser procesado por Tidy.

```
<HTML>
  <HEAD>
    <TITLE>Journey to Windsor</TITLE>
  </HEAD>
<BODY>
<TABLE>
  <TR>
    <TD width=15></TD>
    <TD><FONT size="3"face="Helvetica">
      Journey to Windsor<BR>
      Beno&icirc;t Marchal<BR>
      July 2003<BR>
      <BR>
      <A href="mailto:bmarchal@pineapplesoft.com">
         bmarchal@pineapplesoft.com</A>
    </FONT></TD>
  </TR>
</TABLE>
<CENTER><TABLE border=3>
  <TR><TD>
    <A href="pages/dscn0824.HTML">
      <IMG src="thumbnails/dscn0824.jpg" border="0" alt="dscn0824">
    </A><br>
    <FONT size="3" face="Helvetica">
    dscn0824.jpg<br>
    A bright, red mailbox inside the castle. It seems oddly familiar
    in an historic setting.<br>
    Windsor Castle <br>
    &copy; 2003, Beno&icirc;t Marchal
    </FONT>
 </TD></TR>
</TABLE></CENTER>
</BODY>
</HTML>
```

Listado 2.1. Extracto de un archivo HTML como entrada al Tidy

Se aprecian en **negrita** los cambios hechos después de haber procesado el HTML original con Tidy, además de cambiar las etiquetas a minúsculas para futuro manejo del XML. Se agregan las etiquetas necesarias para definir un documento XML, así como se cierran etiquetas vacías como lo son los brincos de línea
.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/XHTML1/DTD/XHTML1-transitional.dtd">
```

```
<HTML XMLns="http://www.w3.org/1999/XHTML">
<head>
<meta name="generator" content=
"HTML Tidy for Mac OS X (vers 1st June 2003), see www.w3.org" />
<title>Journey to Windsor</title>
</head>
<body>
<table>
<tr>
<td width="15"></td>
<td><font size="3" face="Helvetica">Journey to Windsor<br />
Beno&#238;t Marchal<br />
July 2003<br />
<br />
<a href=
"mailto:bmarchal@pineapplesoft.com">bmarchal@pineapplesoft.com</a></font></td>
</tr>
</table>
<center>
<table border="3">
<tr>
<td><a href="pages/dscn0824.HTML"><img src=
"thumbnails/dscn0824.jpg" border="0" alt="dscn0824" /></a><br />
<font size="3" face="Helvetica">dscn0824.jpg<br />
A bright, red mailbox inside the castle. It seems oddly familiar in
an historic setting.<br />
Windsor Castle<br />
&#169; 2003, Beno&#238;t Marchal</font></td>
</tr>
</table>
</center>
</body>
</HTML>
```

Listado 2.2 Extracto de un archivo XML, salida del Tidy.

2.3.4 Conversión de HTML a WML

Para poder convertir un documento HTML a un documento WML, primero se tiene que convertir el documento a XML bien estructurado, como se ilustra en la Figura 2.3, donde se muestra la entrada (HTML) al proceso de conversión, procesando este con Tidy, generando el XHTML o XML para después generar WML a partir del XML. Como XML es el lenguaje intermedio para la conversión, usaremos el lenguaje XML, el cual es la salida de Tidy.

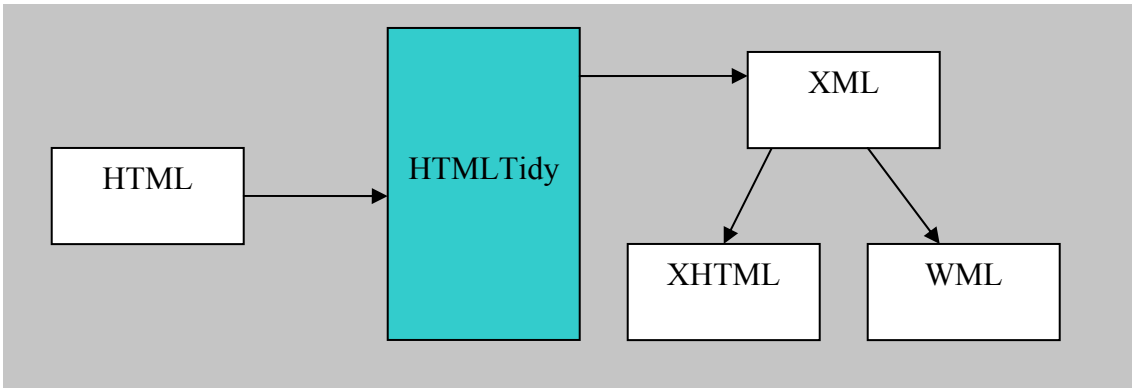

Figura 2.2 Diagrama de HTML Tidy integrado a la herramienta final de conversión.

Una vez generado el documento en XML, el siguiente paso es la conversión de XML a WML. En un inicio se eligió HTML2WML, herramienta de conversión de documentos HTML a WML la cual es el resultado del análisis (ver Apéndice A) del trabajo previo que se menciona a continuación, teniendo una implementación mejorada de éstas [Howard 2001].

1) En la Universidad Estatal de Kansas, David Kapadia es el autor de un proyecto titulado "*Conversión de datos XML a WML* [Kapadia 2000], Kapadia implementa un convertidor usando XSLT, el analizador de XML Xerces y una aplicación JAVA para aplicar la conversión. Los resultados fueron buenos pero el proyecto estuvo muy limitado debido a que la entrada proviene de estructuras de XML ya conocidas. El software tendría que ser reescrito para poder tratar diferentes DTDs o nuevas etiquetas, y por último, la aplicación a HTML nunca fue considerada.

2) Aperghis Maddingue [Aperghis 2002] publicó HTML2WML Versión 0.4.1, un programa registrado bajo de la licencia GNU, la cual es una herramienta de conversión en

tiempo real de HTML a WML escrita en CGI/Perl. Los problemas que se encontraron fueron que la entrada al programa debe ser HTML bien estructurado y la salida generada con esta herramienta está lejos de poder considerarse WML válido, haciéndolo imposible de desplegar en un navegador de WML. Además no permite el uso de *frames*.

3) La Universidad de Durham describe tres aproximaciones de técnicas para procesar XML, pero no específicamente en cuanto a la conversión a WML [Cornelius 2004].

4) LazyWap Versión 0.5 es una herramienta de uso gratuito diseñada en PHP y convierte HTML a WML, creada por un consultor cuso de Internet [Beikov 2001]. Funciona de manera similar al programa en JAVA de Kapadia y a la aplicación en CGI/Perl de Aperghis pero de nuevo no tiene rutinas para manejo de datos más complejos que las simples etiquetas de una página estática.

5) El Centro de Investigación Finlandés propuso métodos respecto al manejo de *frames* y HTML más complejo, pero finalmente su trabajo concluye en la dificultad de convertir HTML mal formado.

Después de detectar los posibles fallos que pudiera tener esta aplicación a pesar de las ventajas mencionadas, se decidió utilizar la misma metodología de PoPS, convertir un archivo XML por medio de las hojas de estilo y un analizador (ver Sección 2.5). Esto permite un mayor manejo del XML, como lo es definir una equivalencia de una etiqueta en HTML en WML o XHTML.

2.4 Herramientas de análisis y manejo de XML

Un analizador de texto (*parser*) es la herramienta XML más básica, pero también es la más importante. Toda aplicación XML está basada en un analizador, las herramientas arriba

mencionadas incluyen un "analizador" para la manipulación de los elementos de los lenguajes de marcado y que se pueda trabajar con éstos. En el caso de Tidy se emplea el analizador DOM mientras que el HTML2WML utiliza SAX en conjunto con hojas de estilo XSL.

2.4.1 DOM

DOM es un API para el *Document Object Model*, el cual está diseñado para proveer una manera de manipulación de datos. Provee la representación de un archivo XML como árbol. DOM también lee un documento XML completo, almacenando toda la información en nodos, por lo que es fácil acceder al documento entero que se encuentra en memoria en toda su longitud como árbol DOM.

DOM es mejor en las siguientes situaciones:

- Cuando se está modificando estructuralmente un documento XML, por ejemplo, al ordenar elementos en un orden particular o al mover algunos elementos de un árbol a otro.
- Cuando se comparte un documento en memoria con otras aplicaciones.

2.4.2 SAX

Un procesador XML con SAX no crea una estructura de datos. En vez de esto, escanea una entrada de un documento XML y genera eventos, como lo es el inicio de un elemento, el fin de elemento, entre otros. La aplicación intercepta estos eventos y hace lo que sea pertinente con respecto a la tarea. SAX es más eficiente que DOM ya que no crea una estructura de datos explícita. Por lo que SAX es mejor en los casos de:

- Cuando se trabaja con un documento grande que no cabe en memoria.

- Cuando se quiere, por ejemplo, contar el número total de elementos en un documento o extraer el contenido de un elemento en específico.

Una aplicación que requiere información acerca de la estructura del documento, como lo es el tipo de elemento, nombres de atributos, y valores de los atributos, puede registrar manejadores (*handlers),* un tipo de llamado de función al procesador XML. El procesador notifica al manejador de eventos, como lo son el inicio de una nueva etiqueta y la existencia de caracteres, así como el fin de etiquetas. A diferencia de usar el API DOM, el cual es recursivo, el proceso entero es un solo paso, las operaciones son llevadas a cabo durante el análisis de los elementos que conforman el documento.

Una vez definidos los lenguajes a utilizar y las herramientas relacionadas al proyecto realizado, se procede a detallar el análisis de requerimientos y el diseño del ambiente de conversión.