Capítulo 3. Herramientas para la construcción de interfaces genéricas

Resumen

Para brindar a los usuarios móviles servicios y páginas Web que se desplieguen correctamente en pantallas pequeñas y se adapten a la limitada capacidad de entrada y de memoria de los dispositivos móviles, se han propuesto varias soluciones. Estas soluciones pueden agruparse en cuatro tendencias:

1. Métodos que resumen el contenido de las páginas Web para que se desplieguen sin problema en pantallas pequeñas
2. Herramientas de conversión, las cuales transforman una interfaz genérica en código escrito en cada uno de los lenguajes de los diferentes dispositivos
3. Aplicaciones que manejan eficientemente los recursos (CPU, memoria y ancho de banda) de los dispositivos computacionales móviles
4. Controles universales

En este capítulo se mencionan brevemente las aplicaciones más representativas de estas cuatro tendencias.

3.1 Métodos para resumir y desplegar páginas Web en dispositivos con pantallas pequeñas

El primer enfoque presenta métodos que resumen el contenido de las páginas Web para que puedan desplegarse correctamente en pantallas pequeñas. A continuación se mencionan brevemente los más representativos.

En [Buyukkokten et al. 2001] se presentan cinco métodos para resumir partes de páginas Web en dispositivos móviles, como PDAs o celulares. Cada página se divide en unidades de texto que pueden ocultarse, desplegarse parcialmente, verse completamente o resumirse. Estos métodos resumen las páginas de diferentes maneras, utilizando técnicas de recuperación de información adaptadas al contexto del World Wide Web. Un método extrae las palabras clave de cada unidad de texto y otro intenta encontrar la oración más significativa y que represente un resumen. Según reportan los autores, las pruebas de usabilidad de este método demostraron que la combinación de palabras clave y resúmenes de una sola oración de texto proporciona un mejor desempeño en tiempos de acceso y

número de operaciones con la pluma comparados con otros esquemas.

Otro sistema divide las páginas Web en fragmentos que puedan adaptarse fácilmente al tamaño reducido de las pantallas de los dispositivos móviles [Kaasinen 2000].

3.2 Herramientas de conversión

3.2.1 Herramientas de conversión de HTML a WML

En el año 2000 se estimaba que existían mil millones de páginas/servicios disponibles en Internet escritas en HTML, y sólo entre 50 000 y 1,500,000 páginas en WML. Para resolver estar diferencia, se crearon herramientas de conversión de HTML a WML. Esta información y la que se presenta a continuación se consultó en [Arehart et al. 2000].

Características. Los convertidores (también llamados transcodificadores) extraen el texto de una página en HTML y lo reformatean a WML, como muestra la Figura 3.1.

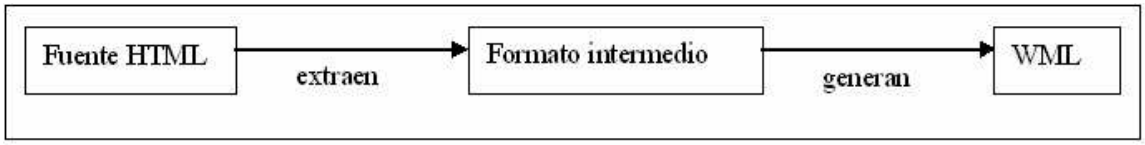

Figura 3.1 Ejemplo de convertidor (adaptado de [Arehart et al. 2000])

Las herramientas realizan la conversión tomando como entrada datos formateados y generando como salida datos sin formato, por lo que los autores de la conversión pueden escoger el formato que se va a aplicar.

La conversión puede ser de dos tipos:

Completamente automatizada: extrae todo el contenido posible de la página, como el mensaje de bienvenida, las ligas, etc. Tanto las ligas como las formas de entrada continúan disponibles para el usuario. Algunas herramientas comerciales de este tipo disponibles actualmente son:

- Phone.come WAP Gateway
- Argo ActiGate

Configurable: extrae partes específicas de la página, como los encabezados nuevos o los precios de mayoreo. La decisión de qué se va a presentar se deja al

desarrollador, pues debe indicar al convertidor qué partes de la página se van a convertir. Ejemplos de convertidores configurables son:

- Oracle's Portal- to-Go
- Spyglass' Prism
- Orchid's Webshaper

Ventajas. Las principales ventajas que presentan las herramientas de conversión con respecto a la construcción completa de páginas WML son:

Velocidad: la conversión toma menos tiempo de programación.

Costo: la velocidad reduce costos.

Independencia del diseño original: el contenido extraído de la página original puede describirse de manera independiente del formato.

La conversión puede extenderse a otros lenguajes: como CompactHTML, HDML, y XHTML, no sólo a WML.

Desventajas. Los convertidores también presentan algunas desventajas que deben tomarse en cuenta:

Errores de conversión: si la página a convertir contiene elementos que el convertidor no reconoce, la conversión falla.

Desempeño: al momento de la ejecución, la conversión de una página toma más tiempo del que tomaría desplegar una página escrita en WML; el convertidor pierde tiempo procesando elementos HTML que son innecesarios para WML.

Costo: una herramienta completa de conversión comercial es muy cara.

No todas las páginas HTML son adecuadas para la conversión: las páginas basadas en audio o en gráficos, como las imágenes del clima y la navegación mediante mapas, no pueden convertirse a WML.

3.2.2 Editores de UIML

3.2.2.1 MyUIML

MyUIML es un ambiente que permite al usuario diseñar y personalizar los componentes de sus interfaces para diversos dispositivos vía Internet. Se basa en UIML.

Características. En este editor el usuario selecciona qué tipo de componente desea

personalizar y los parámetros correspondientes de acuerdo a sus gustos y preferencias. Los componentes disponibles son: campo y área de texto, botón, etiqueta y menú. A partir de las preferencias seleccionadas, MyUIML genera automáticamente el código respectivo para el componente en los siguientes lenguajes: HTML, WML y Java. Esta generación de código la realiza UIML. Los parámetros de los componentes de cada usuario se almacenan para su uso posterior [Sánchez 2002].

Ventajas

- Ambiente gráfico
- Permite la personalización de los componentes de las interfaces de diversos dispositivos
- Disponible vía Web

3.2.2.2 Ambiente visual de programación para la generación de interfaces de usuario independientes del dispositivo

En [Mayora-Ibarra et al. 2003] se presenta una herramienta para diseñar interfaces genéricas mediante la transcodificación automática a múltiples lenguajes objetivo. La herramienta es un ambiente de programación visual que permite arrastrar componentes genéricos creados en UIML y transcodificados a los lenguajes VoiceXML, J2ME, HTML y WML. Esta herramienta aprovecha el lenguaje UIML y los paradigmas de la programación visual para brindar flexibilidad, consistencia y disminuir el tiempo de desarrollo.

Características. Utiliza UIML como el lenguaje fuente para crear elementos genéricos a partir de un vocabulario definido. Sin embargo, usa sólo las primeras cuatro secciones de las cinco que contiene un documento UIML (consultar Apéndice B). En la quinta sección de UIML es donde se debe relacionar cada elemento de la interfaz genérica con su representación en el lenguaje objetivo. Esta propuesta sustituye esta sección con una conversión realizada por el lenguaje de transformación de XML: XSLT.

Arquitectura. La herramienta consta de los siguientes elementos:

- Un ambiente visual de programación para crear las interfaces genéricas en UIML.
- Transcodificadores que convierten la interfaz genérica a tres lenguajes diferentes (WML, VoiceXML, y J2ME). Utilizan XSLT, y un analizador léxico para validar la

sintaxis.

Ventajas

- Ambiente de programación visual
- El ambiente visual cuenta con analizador léxico
- Aprovecha las tecnologías XML al realizar la transcodificación mediante XSLT

Desventajas

- Depende de UIML, pues la interfaz genérica es creada en este lenguaje
- Hasta el momento funciona sólo para 10 elementos
- Garantiza funcionalidad, no usabilidad

3.3 Servicios adaptables para dispositivos computacionales móviles

3.1.1 MOCA

Beck et al. [1999] presentan MOCA, una estructura de servicios adaptables para dispositivos computacionales móviles con memoria limitada, como celulares y PDAs. Para asegurar portabilidad a través de las plataformas heterogéneas de hardware, fue desarrollada en Java.

Características. La estructura de MOCA es abierta y es escalable. Esta estructura permite algunas ventajas:

Los dispositivos se adaptan a su ambiente: mediante descubrimientos y descargas dinámicos de los servicios de dispositivos cercanos.

El software en los dispositivos es extensible: las aplicaciones pueden residir localmente en el dispositivo o ser descargadas de la red inalámbrica de manera transparente.

Las aplicaciones comparten los servicios: MOCA administra eficientemente los recursos de los dispositivos conteniendo diversas aplicaciones en una misma JVM.

Requerimientos. MOCA tiene como objetivos específicos de aplicación dispositivos computacionales móviles que permitan adaptabilidad dinámica, control de rastreo de memoria, seguridad y portabilidad. El tamaño de los componentes básicos de MOCA es de 50K.

Arquitectura. La arquitectura de MOCA está basada en la noción de dos componentes:
Servicio: es un componente de software independiente manejado por MOCA que puede ser activo (con hilos de ejecución) o pasivo. Encapsula una función y proporciona una interfaz específica a la función que realiza. Existen dos tipos de servicios:

local: por ejemplo, servicio de cargado de un archivo.

remoto: por ejemplo, servicio de impresión y servicio de almacenamiento.

Como un servicio puede ser accedido por las aplicaciones o por otros servicios que se ejecutan en un dispositivo, todos los servicios residen en un mismo espacio compartido.

Aplicación: es un programa de Java estándar que declara un método principal y se entrega como uno o más archivos de clases. Las aplicaciones pueden solicitar y usar servicios invocando métodos que proporcionan las interfaces de servicios. MOCA permite que varias aplicaciones se ejecuten en una misma Máquina Virtual de Java (JVM, *Java Virtual Machine*).

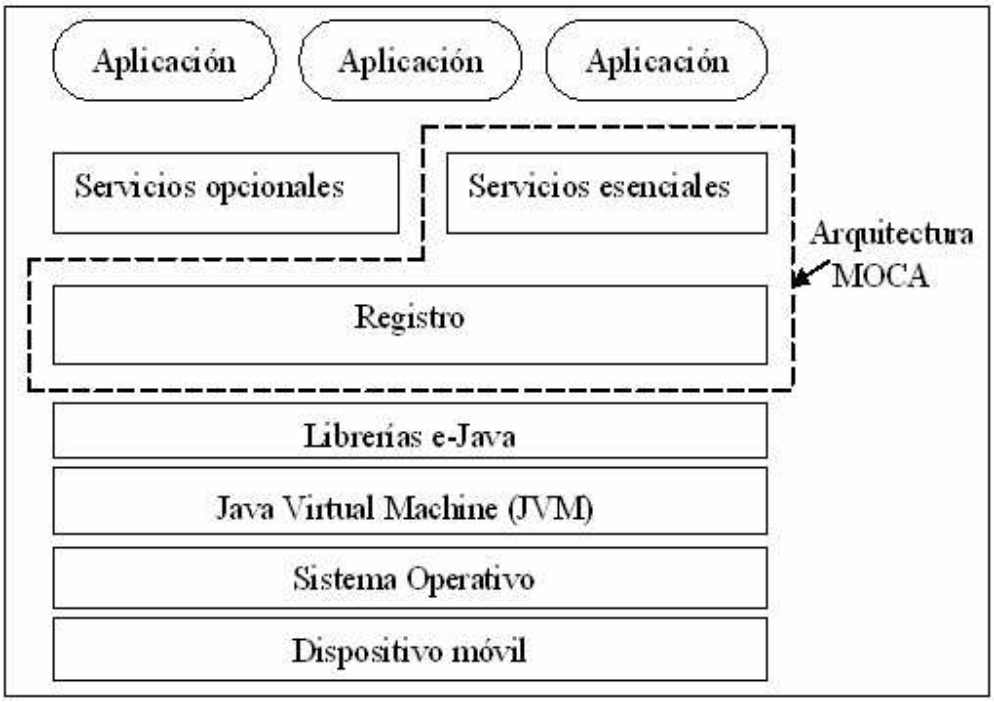

Figura 3.2 Arquitectura MOCA (adaptado de [Beck et al.1999])

Como puede observarse en la Figura 3.2, MOCA se implementa en la parte superior de una JVM.

Los componentes principales de la arquitectura MOCA son:

Registro de servicios. Es el elemento central de la arquitectura MOCA. Realiza dos roles:

- Actúa como un repositorio central de servicios y mantiene el mapeo de la descripción de servicios a las implementaciones correspondientes.
- Encapsula la administración del ciclo de vida de los servicios. En otras palabras, registra, actualiza, y borra el registro de los servicios. Un servicio puede registrarse sin que se descargue en el dispositivo.

Servicios esenciales. Son un subconjunto de servicios que brindan la funcionalidad necesaria para cargar y operar el dispositivo. Sin ellos, MOCA no puede cargar servicios o aplicaciones adicionales.

Servicios opcionales. Pueden residir en el dispositivo o descargarse de la red inalámbrica. Pueden cargarse como respuesta a una petición de una aplicación o descubrirse dinámicamente en un dispositivo cercano. El conjunto de servicios de este tipo es abierto.

Desventajas:

- El desempeño de la plataforma de Java puede ser insuficiente para los dispositivos móviles que cuentan con procesadores lentos.
- No permite la personalización de servicios.

3.4 Controles Universales

3.4.1 Controlador personal universal (*Personal Universal Controller*, PUC)

Los aparatos de oficina y domésticos son cada vez más complejos debido a que ya casi todos ellos contienen algún tipo de computadora. A medida que la complejidad de los aparatos aumenta, las interfaces de los aparatos son más difíciles de usar. Muchas de estas interfaces no presentan opciones para que personas discapacitadas puedan usar los aparatos.

Para resolver este problema varios grupos de investigación, industriales y académicos, están trabajando para simplificar el control de aparatos y servicios creando un control remoto universal. A diferencia de los controles remotos preprogramables disponibles actualmente, estos nuevos controles descargan una especificación del aparato o servicio y la

usan para generar automáticamente una interfaz para el control remoto. Esto promete ser un enfoque útil porque la especificación puede ser lo suficientemente detallada para generar tanto interfaces de voz como gráficas. Sin embargo, la generación de interfaces puede ser difícil.

En [Nichols et al. 2002] sugieren que separar la interfaz del aparato podría solucionar el problema. Proponen un controlador personal universal (*Personal Universal Controller*, PUC), un dispositivo que permite al usuario interactuar con todos los aparatos y servicios de su ambiente.

Características. Un PUC podría tomar varias formas:

- una computadora de mano con interfaz gráfica para una persona discapacitada
- una superficie interactiva de Braille para un invidente
- una diadema que permita síntesis y reconocimiento de voz

Cuando el usuario desee controlar un aparato, el PUC se comunicaría con él, descargaría la especificación con sus funciones y generaría automáticamente una interfaz de control remoto adecuada para el PUC y el usuario. El PUC y el aparato continuarían intercambiando mensajes conforme el usuario manipula la interfaz.

Requerimientos. El sistema PUC debe cumplir con una lista de requerimientos para generar interfaces de alta calidad:

Comunicación bidireccional entre el controlador y el aparato.

Controladores simultáneos: varios controladores deben poder comunicarse con el mismo aparato simultáneamente.

No debe haber información específica de la distribución de los elementos de la interfaz: esto conlleva independencia de modalidad.

Agrupamiento jerárquico mediante árboles: la utilidad de los árboles para agrupar parece ser aceptado universalmente. Casi todos los sistemas actuales usan algún tipo de agrupamiento en árboles.

Variables de estado y comandos representando acciones: las variables de estado y comando son una forma de representar a los elementos manipulables de un aparato de manera concisa.

Dependencia en la información: la información sobre las funciones (si están activas o inactivas) pueden especificarse de manera concisa en términos de valores de las variables de estado.

Ventajas. Este enfoque cuenta con las siguientes ventajas:

- Separa la interfaz de la aplicación.
- Cada PUC crea una interfaz en función de los atributos del aparato.
- Contempla su uso en personas con discapacidades.
- Consistencia de la interfaz a través diferentes aparatos.

Desventajas. Las principales desventajas que presenta este enfoque son:

- Está orientado básicamente a aparatos electrodomésticos.
- No es de gran utilidad para personas sin discapacidad.
- Se encuentra todavía a nivel de investigación.
- Los beneficios de separar la interfaz de la aplicación pueden reducirse si la interfaz generada no cuenta con la calidad suficiente.
- La calidad de la interfaz depende de la lista de requerimientos.

A continuación, el capítulo 4 presenta las tres versiones de espacios personales que se han desarrollado en el contexto del Programa U-DL-A.