2ª Reunión de PFC

Investigación de librerías y de las capacidades de Sonar

López J.D.; Rodriguez M.

22 de julio de 2010

Resumen

En este documento se tratará acerca de un conjunto de tecnologías y librerías que han sido analizados para abordar la realización del Proyecto Fin de Carrera titulado "Cuadro de Mando para la Visualización de la Calidad del Software Integrado con un Entorno de Medición". Para tomar un punto de referencia acerca de los recursos necesarios para implementar metáforas de visualización se ha tomado como referencia el proyecto Sonar [14].

Índice

1. Introducción

El sistema a desarrollar se puede categorizar como una **Aplicación de Internet Enriquecida**, en inglés **R**ich **I**nternet **A**pplication o **RIA**. Una RIA [23], concepto cada vez más común en la era de la Web 2.0, es una aplicación web que, siendo accedida a través de un navegador web estándar, trata de satisfacer las necesidades del usuario como si de una aplicación de escritorio se tratase, combinando así las ventajas de ambos paradigmas (aplicación web y aplicación de escritorio) para maximizar la experiencia del mismo.

Una RIA se difiere de una página web tradicional en el sentido de que no hay una recarga continua de páginas cada vez que el usuario pulsa sobre un enlace. Esto evita producir un alto tráfico de información entre el cliente y el servidor por grande o pequeña que sea la información que se desea actualizar. Además, una RIA posee una gran capacidad multimedia ya que es posible empotrar distintos tipos de objetos (applets, vídeos, sonidos) en ella sin necesidad de utilizar programas externos para su visualización.

Una vez establecidas las características que debe cumplir una RIA y estudiadas las capacidades de visualización que posee Sonar [14], se propone un conjunto de tecnologías y librerías que permitirán implementar un sistema que satisfaga los requisitos de usuario correspondientes.

2. Tecnologías

La capa de conocimiento o reglas de negocio del sistema estarán desarrolladas en Java, por lo que se propone JSP (Java Server Pages) como tecnología principal para el desarrollo de la aplicación web. Se hará uso de otras tecnologías comunes entre la Web 2.0 como CSS, JavaScript y Ajax. No obstante, con un propósito investigador, se propone también el uso de HTML 5, Struts 2 e Hibernate.

2.1. HTML 5

HTML 5 (HyperText Markup Language, versión 5) [22] es la quinta revisión de HTML. Esta nueva versión incorpora de forma nativa, es decir, sin depender de programas externos o *plugins* del navegador, reproducción de audio y vídeo, edición de documentros, *drag&drop*, un elemento *canvas* sobre el que se pueden pintar gráficos en 2 y 3 dimensiones, etc. Esta última característica hace que sea posible el uso de WebGL (ver sección 3.2.6).

2.2. Struts 2

Struts 2 es la nueva versión del *framework* de desarrollo web Java Apache Struts. Struts 2 está basado en el patrón MVC (Modelo-Vista-Controlador), una arquitectura que busca reducir el acoplamiento dividiendo las responsabilidades en 3 capas bien diferenciadas:

- El modelo, que hace referencia a los datos que maneja la aplicación y las reglas de negocio que operan sobre ellos.
- La vista, encargada de generar la interfaz con la que la aplicación interacciona con el usuario.
- El controlador, que comunica la vista y el modelo respondiendo a eventos generados por el usuario en la vista, invocando cambios en el modelo, y devolviendo a la vista la información del modelo necesaria para que pueda generar la respuesta adecuada para el usuario.

No obstante, también es posible diseñar una aplicación web mediante el patrón MVC utilizando páginas JSP y servlets. En este caso, las páginas JSP tomarían el rol de la vista, los servlets el del controlador y los POJO (Plan Old Java Object) el del modelo. El modelo, a su vez, suele estar dividido en dos subcapas siguiendo el patrón DAO (Data Access Object).

En la aplicación web, Struts 2 se empleará en el desarrollo de determinados casos de uso extraídos de la especificación de requisitos de usuario, identificando dichos casos de uso como *acciones* de Struts.

En [15] se encuentra una lista con todos los plugins de los que dispone este framework: jQuery, Hibernate, jFree Chart, JUnit, YUI, GWT, etc.

2.3. Hibernate

Hibernate es un *framework* de persistencia, libre y bajo licencia LGPL, que permite establecer un *mapeo objeto-relacional* para la plataforma Java y sistemas basados en SQL, es decir, permite relacionar una clase Java con una tabla de una base de datos para abstraer al programador del lenguaje SQL y haciendo que la aplicación sea portable entre cualquier sistema gestor de bases de datos compatible con SQL.

Este framework, además de utilizarse para gestionar la persistencia del núcleo de Medusas (ver Figura 1), se utilizará para gestionar la persistencia de la aplicación web: gestión de usuarios, estadísticas de uso, historiales, etc.

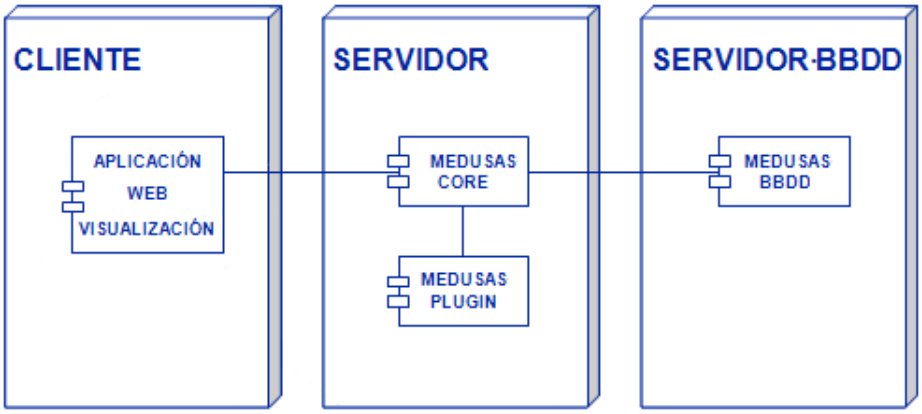

Figura 1: Arquitectura del proyecto Medusas

3. Librerías

Las librerías que se han analizado para el desarrollo de la aplicación web se han dividido en dos categorías: librerías para el desarrollo web y librerías para visualización.

La primera categoría se compone de librerías que darán soporte para dotar la aplicación web características similares a la de una aplicación de escritorio, maximizando así la experiencia del usuario. La segunda categoría se compone de librerías que permiten generar distintos tipos de gráficos y diagramas para poder representar los distintos aspectos de calidad que se estudiarán a lo largo de la elaboración del Proyecto Fin de Carrera.

3.1. Desarrollo Web

Una aplicación web tiene dos contextos bien diferenciados: el lado del cliente (client-side) y el lado del servidor (server-side). Para poder dotar una aplicación web de las características de

una aplicación de escritorio es necesario trabajar desde el lado del cliente. Para ello, se dispone de tecnologías como CSS, Ajax y JavaScript que nos permiten cambiar dinámicamente el contenido que el usuario recibe en su navegador web a través del DOM[1].

3.1.1. DWR: Direct Web Remoting

Versión: 2.0 stable
Licencia: Apache Software License v2 [18]
Página Web: [3]

DWR, bajo el eslogan de "Easy Ajax for Java", es una solución muy utilizada en el mundo empresarial para generar código JavaScript que, ejecutándose en el cliente, permite que los navegadores web hagan peticiones a clases Java que se ejecutan en el servidor como si éstas se ejecutasen en local. Además, con *Reverse Ajax*, es posible que el código Java que se está ejecutando en el servidor utilice un API para modificar el contenido y estilo de los clientes, estableciendo así comunicación en ambos sentidos: del cliente al servidor y del servidor al cliente.

Por último, es posible integrar DWR con tecnologías como Dojo (client-side) y Struts e Hibernate (server-side).

3.1.2. Dojo

Versión: 1.5.0 (20 de Julio de 2010)
Licencia: BSD
Página Web: [2]

Dojo es un **framework para JavaScript** que permite a los desarrolladores dotar de funcionalidades y efectos dinámicos (eventos, *drag and drop*, redimensionar paneles, etc.) a las aplicaciones web que utilizan la tecnología Ajax. Se compone de tres grandes paquetes de propósitos muy genéricos:

- Dojo, que contiene el núcleo del framework para proporcionar las funcionalidades básicas.
- Dijit, que contiene todos los *widgets* básicos necesarios para implementar una buena interfaz gráfica de usuario: *sliders*, barras de progreso, barras de menú, pestañas, etc.
- Dojox, que consiste en un conjunto de extensiones avanzadas para Dojo entre las que podemos encontrar *widgets* complejos (calendarios, etc.), librerías para generar y manipular gráficas y diagramas (dojox.charting), APIs para interactuar con servicios de Google, etc.

Numerosas empresas como Google, IBM, OpenLaszlo y AOL contribuyen al soporte de este proyecto, aunque actualmente la documentación disponible es incompleta y limitada.

3.1.3. jQuery

Versión: 1.4.2
Licencia: MIT o GPLv2
Página Web: [10]

jQuery es un **framework para JavaScript**, similar a Dojo, pero en vez de estar organizado en un núcleo y grandes paquetes de propósito general, su arquitectura se basa en

[1]DOM, acrónimo de Document Object Model, es un API a través del cuál se puede modificar dinámicamente el contenido, estructura y estilo de los documentos HTML y XML.

pequeños *plugins* muy específicos. Este framework está muy extendido entre la comunidad de desarrolladores de aplicaciones web y contiene grandes fuentes de documentación.

Cabe destacar **jQuery UI** [11], que se ejecuta sobre jQuery y provee al programador de un mayor nivel de abstracción en cuanto a interacciones con la interfaz gráfica de usuario, animaciones, *widgets* complejos, etc.

3.2. Visualización

Se han analizado numerosas librerías de visualización [21] tales como jQuery Visualize [12], Flot [4], Axiis [1], Style Chart [16], JFree Chart [7] y Google Chart API [5], entre otros. Finalmente se han seleccionado para profundizar en su estudio las que se enumeran a continuación. El criterio de selección se fundamenta en las siguientes características:

- Generación dinámica de diagramas.
- Posibilidad de elección entre una gran colección de modelos (diagramas de barras, sectores, radar, etc.).
- Permitir la interacción por parte del usuario.
- Visualización atractiva.
- Fácil integración en la aplicación web, desarrollada en Java.
- Gratuito y, preferiblemente, libre.
- Una licencia conforme a nuestros requisitos.

Para satisfacer dichos requisitos se han tenido en cuenta librerías específicas para la generación de diagramas y librerías de propósito general para gráficos 2D y 3D.

3.2.1. DojoX Charting

DojoX Charting es una extensión para Dojo (ver sección 3.1.2) que permite la generación de gráficas y diagramas (ver Figura 2) para su posterior integración en páginas web. Además, soporta zoom, *tooltips* y permite resaltar e interaccionar con elementos del diagrama para ofrecer una mayor experiencia de usuario, capturando los eventos correspondientes y asignándoles manejadores.

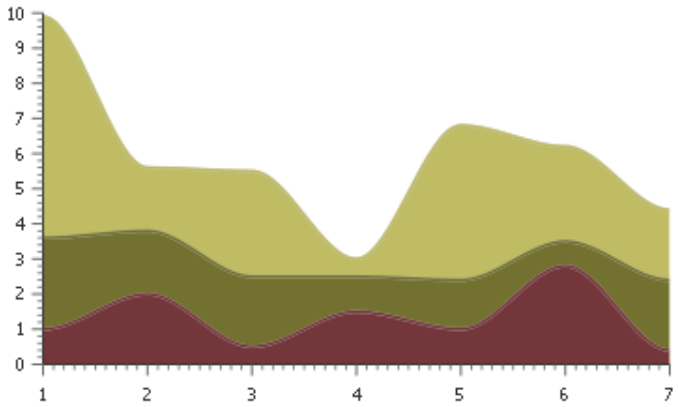

Figura 2:

3.2.2. Google Visualization API

Versión: 16 de Junio de 2010
Licencia: [19]
Página Web: [6]

Es una librería perteneciente a Google. Entre sus características destacan las siguientes:

- Gran cantidad de documentación.
- Buen soporte y actividad por parte del equipo de desarrollo.
- Basado en JavaScript.
- Permite a los desarrolladores crear nuevos modelos de visualización y compartirlos con la comunidad.
- Soporta interacción por parte del usuario gracias a una fuerte captura y manipulación de eventos.
- Permite ordenar, filtrar y manipular los datos.
- Los orígenes de datos pueden ser la propia página o páginas externas, por ejemplo Google Docs.
- Proporciona una gran variedad de modelos: líneas, barras verticales y horizontales, sectores, áreas, árboles, **geo-mapas** (ver Figura 3), mapas interactivos que utilizan el API de Google Maps, tablas, **nubes de palabras** (ver Figura 4), etc.
- El código de visualización no puede ser almacenado localmente. Se debe acceder a *http://www.google.com/jsapi* para utilizar las visualizaciones.

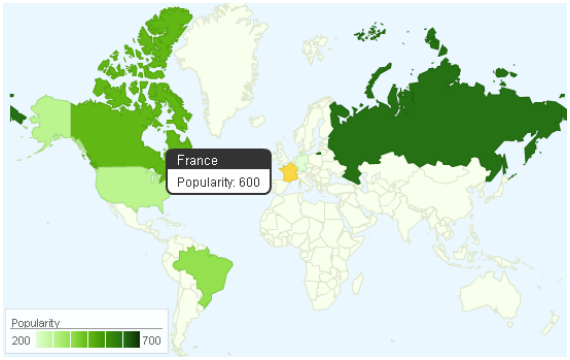

Figura 3: Geomapa de Google Visualization API

Figura 4: Nube de palabras de Google Visualization API

3.2.3. The JavaScript InfoVis Toolkit

Versión: 2.0.0a (28 de Junio de 2010)
Licencia: Creative Commons 3
Página Web: [8]

Características:

- Desarrollada completamente en JavaScript.
- Soporta la captura y manipulación de eventos.

- Soporta zoom y animaciones.
- Permite que el usuario mueva los diagramas o sus elementos.
- Soporte para dispositivos móviles.
- Proporciona una gran variedad de modelos: líneas, barras verticales y horizontales, sectores, áreas, árboles. **No soporta diagramas de tipo radar**.
- Gran potencial en cuanto a la manipulación de grafos.

3.2.4. Open Flash Chart

Versión: 2 (28 de Julio de 2009)
Licencia: LGPL [20]
Página Web: [13]

Características:

- Libre y gratuito.
- Se basa en flash para generar los diagramas.
- Incluye un plug-in para ser integrado con numerosos lenguajes, entre ellos Java y el framework Struts 2.
- Captura y manipulación de eventos.
- Grandes factores de configuración y personalización: capturar eventos, barras de carga, menús (ver Figura 6), tooltips, etc.
- Proporciona una gran variedad de modelos: líneas, barras verticales y horizontales, sectores, áreas, **radar** (ver Figura 5), etc.
- Permite incluir varios diagramas en una misma página.
- Los diagramas pueden ser exportados a imágenes estáticas.

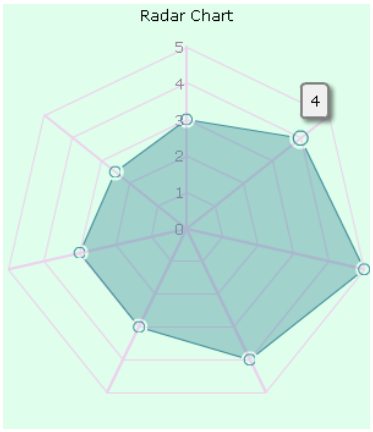

Figura 5: Diagrama de tipo “radar” con Open Flash Chart

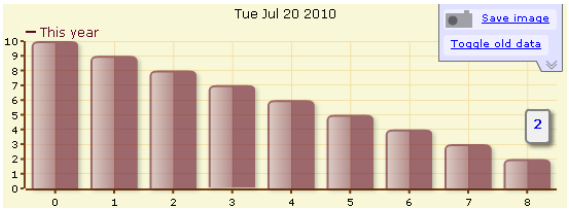

Figura 6: Diagrama con menú con Open Flash Chart

3.2.5. JOGL: JavaOpenGL

Versión: 1.1.1a
Licencia: BSD
Página Web: [9]

JOGL es un *binding* de Java para el API de OpenGL (JSR-231), que permite desarrollar aplicaciones que manejen gráficos 3D. JOGL implementa todas las funciones de OpenGL 1.3 - 3.0, 3.1, ES 1.x y ES 2.x, y se integra con AWT y Swing.

En el proyecto se utilizará para crear pequeños componentes integrables en la aplicación web, que permitirán representar en 3D las metáforas de los distintos aspectos de calidad del software.

3.2.6. WebGL: OpenGL ES 2.0 for the Web

Página Web: [17]

WebGL [17] es una especificación estándar para un API basada en OpenGL ES 2.0, que permite desplegar gráficos en 3D a través del elemento *canvas* de HTML 5.

Esta tecnología se emplearía en el desarrollo del proyecto en sustitución de Java OpenGL. No obstante, se observan ciertas ventajas y desventajas que habría que considerar junto con el coste que podría conllevar la migración de una tecnología a otra:

- WebGL dibuja los gráficos sobre el elemento *canvas* que proporciona HTML 5. Por el contrario, para integrar Java OpenGL en la web sería necesario hacer uso de *applets*.
- WebGL se encuentra en una fase muy experimental y sólo se encuentra implementado en versiones *nightly* de navegadores como Mozilla Firefox, Google Chrome y Safari. Por otro lado, Java OpenGL dispone de versiones estables.
- Tanto WebGL como Java OpenGL respetan al máximo la especificación propuesta por la versión de OpenGL ES 2.0.

Referencias

[1] Página principal de Axiis. http://www.axiis.org/.

[2] Página principal de Dojo Toolkit. http://www.dojotoolkit.org/.

[3] Página principal de DWR: Easy Ajax for Java. http://directwebremoting.org/dwr/index.html.

[4] Página principal de Flot. http://code.google.com/p/flot/.

[5] Página principal de Google Chart API. http://code.google.com/apis/chart/index.html.

[6] Página principal de Google Visualization API. http://code.google.com/intl/es-ES/apis/visualization/interactive_charts.html.

[7] Página principal de JFree Chart. http://www.jfree.org/.

[8] Página principal de JavaScript InfoVis Toolkit. http://thejit.org/.

[9] Página principal de Java OpenGL. http://jogamp.org/jogl/www/.

[10] Página principal de jQuery. http://jquery.com/.

[11] Página principal de jQuery UI. http://jqueryui.com/.

[12] Página principal de jQuery Visualize. http://www.filamentgroup.com/lab/update_to_jquery_visualize_accessible_charts_with_html5_from_designing_with.

[13] Página principal de Open Flash Chart v2. http://teethgrinder.co.uk/open-flash-chart-2/.

[14] Página principal de Sonar. http://www.sonarsource.org/.

[15] Apache Struts 2 Plugin Registry. https://cwiki.apache.org/S2PLUGINS/home.html.

[16] Página principal de Style Chart. http://chart.inetsoft.com/.

[17] Página principal de WebGL. http://www.khronos.org/webgl/.

[18] Apache Software License v2. http://www.apache.org/licenses/LICENSE-2.0.html.

[19] Términos de uso de Google Visualization API. http://code.google.com/intl/es-ES/apis/visualization/terms.html.

[20] GNU Lesser General Public License v3. http://www.gnu.org/licenses/lgpl.html.

[21] 22 Awesome Visualizatin Libraries: Charts and Diagrams. http://www.onextrapixel.com/2009/11/24/22-awesome-visualization-libraries-charts-and-diagrams/.

[22] Definición HTML 5: HyperText Markup Language, versión 5. http://en.wikipedia.org/wiki/HTML5.

[23] Definición RIA: Rich Internet Applications. http://es.wikipedia.org/wiki/Rich_Internet_Applications.