OpenUSS: Un sistema libre de manejo del aprendizaje

Arturo Morales López

Universidad Pedagógica Nacional – Unidad Ajusco, Carretera al Ajusco #24, Col. Héroes de Padierna, México, D.F., 14200.
México
et@upn.mx

Resumen. Actualmente la mayoría de los sistemas de manejo del aprendizaje se basan en tecnologías de desarrollo web. OpenUSS es un sistema libre de manejo del aprendizaje que incorpora lo último en tecnología Java del lado del servidor así como una serie mejores prácticas orientadas a tener un desarrollo estructuralmente limpio y de fácil extensión.

Palabras clave: OpenUSS, GPL, J2EE, EJOSA.

1 Introducción

OpenUSS[1] es un sistema libre (en el sentido GPL[2]) de manejo del aprendizaje que proporciona un mecanismo simple de manejo de documentos así como canales de comunicación entre los diversos actores del proceso de aprendizaje y enseñanza asistido por computadora a través del web.

OpenUSS es desarrollado con un diseño modular utilizando lo último en tecnología J2EE[3]. J2EE (Java 2 Enterprise Edition) es un conjunto de especificaciones de Sun microsystems para proveer soluciones Java del lado del servidor que cualquier proveedor independiente de software puede implementar. Estas especificaciones dictan como deben ser los ambientes de ejecución (contenedores) y los componentes que satisfacen las necesidades básicas de una aplicación web:

1.-Servlets: objetos basados en un mecanismo de petición y respuesta que nos ofrece un punto de entrada y salida a nuestra aplicación.
2.-Enterprise Java Beans o EJB's: componentes destinados a modelar los procesos y entidades de nuestro dominio.
3.- Java Server Pages o JSP's: Nos permiten generar interfaces web de usuario de manera muy sencilla.

Los contenedores que albergan estos componentes son responsables de manejar los servicios de bajo nivel dentro de la aplicación como la persistencia, transacciones, seguridad, etc. Sin embargo el uso de estos componentes y contenedores no nos dice como atacar las dificultades de tipo estructural que

podemos encontrar: ¿Qué objeto es responsable de obtener la información de la base de datos?, ¿Qué objeto es responsable de localizar y crear las interfaces a mis componentes remotos?. La respuesta a este tipo de preguntas se resuelve usando la experiencia previa de muchos desarrolladores que proponen una serie de mejores prácticas cuando se trata de resolver estos problemas.

2 Metodología.

Para poder implementar esta aplicación haciendo uso eficiente de los componentes y contenedores anteriormente mencionados recurrimos a EJOSA [4].

2.1 EJOSA (Enterpsise Java Open Source Architecture) es una arquitectura libre basada en mejores prácticas que resuelve las dificultades estructurales comúnmente encontrados en el desarrollo de aplicaciones distribuidas de varias capas. EJOSA hace uso de Enhydra y XMLC[5] para generar los Servlets que dan vida a la capa de la presentación mientras que JOnAS soporta los EJB's usados para implementar los procesos y entidades de nuestro modelo. Dicho modelo a su vez puede permanecer alojado en cualquier base de datos relacional que tenga un manejador Java.

2.2 Usamos XMLC para generar de manera automática el DOM[5] de los objetos de nuestra interfaz de usuario a partir de documentos XML/HTML.

2.3 Los procesos y entidades fueron implementados usando EJB's aplicando una serie de patrones de diseño para resolver problemas específicos de aplicaciones Web. Estos patrones ya han sido implementados a nivel de especificación dentro de EJOSA lo cual nos deja las manos libres para entrar de lleno a la implementación de los procesos ocurrentes dentro de nuestro dominio.

3 Experiencia

Como resultado tenemos una aplicación modular, altamente escalable y capaz de distribuirse a través de varias instancias. Actualmente cuenta con más de 10 000 usuarios alrededor del mundo divididos principalmente en dos instancias: Universidad de Munster y Universidad Pedagógica Nacional [6].

Conclusiones

Este artículo propone dar una idea general del uso una arquitectura libre basada en las últimas tecnologías del desarrollo web. El poder y simplicidad de esta arquitectura es de suma importancia tanto para el desarrollo de LMS's (sistemas de manejo del aprendizaje) como para el desarrollo de cualquier aplicación distribuida de varias capas, aun más tomando en consideración que esta arquitectura próximamente implementará el uso de C-JDBC[7], tecnología que nos permite albergar nuestro modelo en cúmulos de bases de datos. Sin embargo cabe mencionar que el trabajo en la capa de presentación puede ser considerablemente reducido utilizando un modelo de desarrollo orientado a componentes visuales como Barracuda MVC[8].

Referencias

[1] OpenUSS. 2000. OpenUSS. http://openuss.sourceforge.net/ . (2003).
[2] GPL, GNU. ¡GNU No es Unix! - el Proyecto GNU y la Fundación para el Software Libre (FSF). http://www.gnu.org/home.es.html . (2003).
[3] E. Armstrong, S. Bodoff et al. 2003. The J2EE 1.4 tutorial. http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html. (2003).
[4] EJOSA. 2000. EJOSA – Enterprise Java OpenSource. http://ejosa.sourceforge.net . (2003).
[5] W3C, DOM. W3C Document Object Model. http://www.w3.org/DOM/ . (2003).
[6] OpenUSS, Reference implementation. 2002. Reference implementation. http://openuss.sourceforge.net/openuss/referenceimpl.html . (2003).
[7] Objectweb, CJDBC. C-JDBC Home Page. http://c-jdbc.objectweb.org/ . (2003).
[8] Barracuda, MVC, XMLC. Barracuda Presentation Framework - 1.2.0 (Basic Docs). http://barracudamvc.org/Barracuda/index.html . (2003).