
ARQUITECTURA PARA SISTEMAS DE EDUCACIÓN BASADA EN WEB USANDO PROGRAMACIÓN ORIENTADA A COMPONENTES

*Arquitetura para os sistemas de educação baseado
em WEB orientado aos componentes*

**Rubén Peredo Valderrama^a, Alejandro Canales Cruz^b,
Leandro Balladares^c, Alain Menchaca Resendiz^d**

^a Maestro en Ciencias, Laboratorio de Inteligencia Artificial del Centro de Investigación en Computación, Profesor Investigador, IPN, México, DF - México, e-mail: peredo@cic.ipn.mx

^b Doctor en Ciencias, Universidad Iberoamericana, Profesor Investigador, IPN, México DF - México, e-mail: acc@cic.ipn.mx

^c Maestro en Ciencias, Centro de Investigación en Computación, Profesor Investigador, IPN, México DF - México, e-mail: ballad@cic.ipn.mx

^d Candidato a Maestro en Ciencias, Centro de Investigación en Computación, Estudiante, IPN, México DF - México, e-mail: alainb06@sagitario.cic.ipn.mx

Resumen

En este trabajo, se presenta una nueva Arquitectura Orientada a Componentes y Agentes (AOCA) para sistemas de educación basada en Web (Web-Based Education, WBE por sus siglas en inglés). Esta arquitectura se basa en la especificación del modelo IEEE 1484 (Learning Technology System Architecture, LTSA por sus siglas en inglés), utilizando el patrón de desarrollo que se ha denominado Intelligent Reusable Learning Components Object Oriented (IRLCOO). Los IRLCOO son un tipo especial de Sharable Content

Object (SCO) compatible con Sharable Content Object Reference Model (SCORM). El nuevo modelo AOCA se mezcla con metadatos (SCORM 2004, RDF, XML), para ser usados por el motor de inferencias basado en reglas conocido como Jena y además, se utiliza Joseki como servidor Web para la implementación de una plataforma semántica. La arquitectura que se describe en este trabajo se usa para desarrollar herramientas CASE para la autoría para contenidos y evaluaciones, orientadas a ofrecer interoperabilidad a nivel de aplicación bajo la filosofía de Servicios.

Palabras-clave: Educación basada WBE; Web semántica; IRLCOO; SCORM.

Resumo

Neste trabalho, apresentamos uma nova Arquitetura Orientada a Componentes e Agentes (AOCA), para sistemas de Educação Baseado em Web (Web-Based Education, WBE por suas siglas em inglês). Esta arquitetura está baseada na especificação do modelo IEEE 1484 (Learning Technology System Architecture, LTSA por suas siglas em inglês), usando o padrão de desenvolvimento que foi denominado Intelligent Reusable Learning Components Object Oriented (IRLCOO). Os IRLCOO são um tipo especial de Sharable Content Object (SCO) compatível com Sharable Content Object Reference Model (SCORM). O novo modelo AOCA mistura metadados (SCORM 2004, RDF, XML), para ser usado pelo motor de inferências baseado em regras conhecido como Jena e ademais é usado Joseki como servidor Web para a implementação de uma plataforma semântica. A arquitetura que é descrita neste trabalho é usada para desenvolver ferramentas CASE para desenvolvimento de conteúdos e avaliações, orientado para oferecer interoperabilidade de aplicação sob a filosofia de Serviços Web.

Palavras-chave: Educação baseada em web; Web semântica; IRLCOO; SCORM.

La especificación IEEE LTSA, lanzada por el grupo de trabajo de la IEEE “1484 Comité de Estándares de Tecnologías de Aprendizaje (*Learning Technology Standards Committee*, LTSC por sus siglas en inglés)”, cubre un amplio

rango de sistemas, comúnmente conocidos como tecnologías de aprendizaje, entrenamiento basado en computadora, instrucción asistida por computadora, tutoría inteligente, tecnología de educación y entrenamiento, etc. (IEEE, 2001).

En general este estándar es aceptado dentro de la comunidad de aprendizaje *Web*, siendo el propósito de este trabajo mostrar una nueva Arquitectura Orientada a Componentes y Agentes (AOCA), para el desarrollo de sistemas para WBE basados en la especificación IEEE 1484 LTSA. La idea surge como una respuesta a la necesidad de pasar de la generación de materiales educativos hechos de forma artesanal, hacia una producción en serie de estos materiales, resolviendo los problemas técnicos del profesor a través del modelo de componentes, simplificando la creación y publicación de los cursos dentro de sistemas para WBE. En los aspectos tecnológicos se ha buscado que los materiales bajo el patrón de componentes cumplan con: calidad, accesibilidad, flexibilidad, reuso, adaptabilidad, interoperabilidad, reducción de tiempos de entrega y reducción de los altos costos. Mientras que al mismo tiempo se ha tratado de cubrir el aspecto de implementación de una metodología pedagógica dentro de los cursos que ayuden al estudiante dentro de los ambientes para WBE.

Los IRLCOOs son parte fundamental de la nueva arquitectura en conjunción con estándares abiertos tales como XML (XML, 2003), para conformar vocabularios que sirven como sistemas codificadores de información que aseguran que los materiales educativos sean interoperables con diferentes sistemas LMS como: *Global IMS Learning Consortium* (IMS, 2005), *Advanced Distributed Learning* (ADL), y ADL SCORM (ADL, 2004). El trabajo presentado esta basado en los IRLCOOs desarrollados por Peredo et al. (2005). Los IRLCOOs representan una clase de contenido de aprendizaje caracterizado por: multimedia, alta interactividad, intensa retroalimentación. Todo soportado por medio de la Interfaz de Programación de Aplicaciones (*Interface Programming Application*, API por sus siglas en inglés), implementada para la funcionalidad de la aplicación.

IEEE 1484 LTSA

El comité de estándares de tecnología de aprendizaje 1484 de la IEEE propuso una arquitectura que se pretende sea un estándar para todos los sistemas de tecnologías de aprendizaje. Cada una de las capas se describe a continuación:

1. las interacciones del ambiente y de los estudiantes: concerniente a la adquisición, transferencia, intercambio, formulación, etc., del conocimiento y/o información a través de la interacción con el ambiente;

2. características de diseño relacionadas al estudiante: concernientes a los efectos que tienen sobre el diseño de los sistemas de tecnología de aprendizaje;
3. sistema de componentes: describe la arquitectura basada en componentes, identificado los procesos que intervienen en el sistema y se centran en el estudiante;
4. implementación de perspectivas y prioridades: describe los sistemas tecnológicos de aprendizaje desde una amplia variedad de perspectivas por referencia a subconjuntos de la capa de componentes del sistema;
5. interoperabilidad y funcionalidad de los componentes – codificación, APIs, protocolos: describen los componentes genéricos *plug and play* (interoperables) e interfaces de una arquitectura tecnológica de aprendizaje basada en las tecnologías de la información.

Básicamente AOCA propone mejoras en la capa 3 de la especificación IEEE 1484 LTSA que se muestra en la siguiente Figura 1.



FIGURA 1 - Componentes del sistema

Arquitectura orientada a componentes y agentes

Entre uno de los puntos clave de la ingeniería de software esta la meta de desarrollar software de calidad. Con esto en mente, los componentes son ampliamente vistos por los ingenieros de software como una de las principales tecnologías para resolver la denominada crisis del software. La nueva revolución del software pasa por la ingeniería de software basada en componentes. Entre las razones fundamentales que sustentan la relevancia de la Programación Orientada a Componentes (*Component Oriented Programming*, COP por sus siglas en inglés),

están: un alto nivel de abstracción ofrecido por el paradigma y las tendencias actuales para conformar librerías de componentes reusables de autoreo, que soporten el desarrollo de aplicaciones para diferentes dominios. Wang; Qian (2005) muestra las tres principales metas de la COP: dominio de la complejidad, manejo del cambio y reusabilidad.

Existen diferentes definiciones de lo que es un componente de software, una de las más ampliamente aceptadas es la de Szyperski que lo define de la siguiente manera: “un componente de software es una unidad de composición con interfaces especificadas contractualmente y dependencias explícitas de contexto. Un componente de software puede ser desplegado de manera independiente y es objeto de composición por terceras partes” (SZYPERSKI, 2002).

La arquitectura propuesta está basada en la capa 3 de la especificación IEEE 1484 LTSA. La arquitectura modificada consiste de cuatro procesos: Entidad del estudiante, Evaluación, Coach y Entrega. Se tienen dos bases de datos: registros del estudiante y recursos de aprendizaje. Además de catorce flujos de información.

El proceso del Coach debe ser dividido dentro de dos subprocesos: Coach y Coach virtual. Esta decisión se debe a las necesidades individuales de los estudiantes, algunas deben de ser atendidas por el Coach de manera manual, mientras que otras como: secuencia de navegación de los contenidos, secuenciación de las evaluaciones, estatus de actividades, estatus de tareas, etc., deben ser capturadas de manera automáticamente, siendo responsabilidad del Coach virtual la captura de las métricas mediante los componentes IRLCOOs de los contenidos, evaluaciones, FAQ (*Frequently Asked Questions*), foros de discusión, Chat y herramientas colaborativas.

La operación de los Componentes del sistema es la siguiente: los estilos de aprendizaje, estrategias, métodos, etc., son negociados entre el estudiante y los otros participantes, comunicando sus preferencias de aprendizaje. A continuación la información del estudiante (comportamiento dentro del curso, trayectoria, nomadidad, etc.) se almacenan dentro de los registros del estudiante, esto es una nueva propuesta nuestra dentro de la arquitectura de la IEEE. Los estudiantes son observados y evaluados dentro del contexto de componentes IRLCOOs multimedia interactivos. Las evaluaciones a lo largo del curso producen una evaluación, que constituye información acerca del estudiante. La información del estudiante colectada a lo largo del curso por medio de los componentes IRLCOOs (captura información del teclado del estudiante, elecciones del estudiante mediante el ratón, respuestas de voz, respuestas escritas, etc., en general información del estudiante a lo largo del curso), son almacenadas dentro la base de datos en los registros del estudiante. El Coach revisa las evaluaciones del estudiante e información del estudiante, tales como las preferencias, registro del desempeño del estudiante, y revisión de los objetivos de aprendizaje. El Coach virtual propuesto

por los autores de este trabajo revisa la actuación del estudiante y graba su información personalizada vía los componentes IRLCOOs, haciendo una captura automática de las métricas y haciendo modificaciones dinámicas de la secuencia del curso en función de los resultados (personalización del curso de acuerdo a las verdaderas necesidades del estudiante) basándose en el diseño del proceso de aprendizaje. El Coach y Coach virtual buscan los recursos de aprendizaje, vía consulta a la BD de los Recursos de aprendizaje, regresando un catalogo de información con contenido de aprendizaje adecuado a las especificaciones del Coach. El proceso de Entrega tiene los localizadores de los recursos y hace una solicitud a la base de datos de los recursos de aprendizaje extrayendo los recursos de aprendizaje (e.g., URLs). El proceso de entrega extrae los contenidos de aprendizaje multimedia interactivos para el estudiante, adaptándolos a las verdaderas necesidades del estudiante.

Plataforma IRLCOO

Los IRLCOO fueron desarrollados con un integrador de medios llamado Flash de Macromedia (MACROMEDIA, 2005). Este integrador de medios tiene un lenguaje de OOP denominado ActionScript 3 (MACROMEDIA, 2005). Una característica fundamental de los componentes es la carga en tiempo de ejecución de los medios (Archivos Flash Video, Archivos MP3, Archivos SWF y Archivos JPG), además de ofrecer un ambiente adaptivo y programable que se adapta a las necesidades de los estudiantes. Tomamos como objetivo generar una librería multimedia de componentes IRLCOO para sistemas de WBE teniendo en mente la separación del contenido de la navegación, para su posterior reutilización. Mientras que el *Run Time Environment* de la ADL permite separar los contenidos de la navegación, el costo que hay que pagar con este esquema es alto, al necesitar nuestras instituciones educativas un LMS compatible, además de que los materiales solo son accesibles vía *Web*. Mientras que con nuestro esquema basado en componentes es posible producir materiales educativos duales, que pueden ser accedidos vía *Web*, y pudiendo ser utilizados los componentes IRLCOO como unidades de composición para aplicaciones de escritorio, usándolos como componentes ActiveX, distribuibles vía CD o DVD. Los componentes usan diferentes niveles dentro del Flash Player, permitiendo conformar una estructura jerárquica. Esta estructura permite utilizar reutilización binaria de componentes, creando componentes especializados más pequeños, reusables, y adecuados para agregarse dentro de componentes más complejos en tiempo de ejecución. Los componentes IRLCOOs tienen interconstruidos una API para la comunicación con el sistema LMS y el MAS.

Los IRLCOO son meta etiquetados con la función de proporcionar información adicional para identificar los componentes y determinar especificaciones características. El meta etiquetado es hecho vía un subsistema que permite generar etiquetas SCORM 2004 y *Resource Description Framework* (RDF-XML) (RDF, 2007), posibilitando cierto grado de inferencias sobre los materiales por medio de una plataforma semántica.

Consumo de servicios web de los IRLCOO

ActionScript 3.0 tiene el componente *WebServiceConnector* para poder consumir los denominados Servicios Web (SW) desde los componentes IRLCOOs. Dicho componente posibilita el acceso a los métodos remotos ofrecidos por el LMS a través del protocolo SOAP (*Simple Object Access Protocol*). Posibilitando a los SW la habilidad de aceptar argumentos y retornar un resultado al script. El componente *WebServiceConnector* se usa para hacer múltiples llamadas a la misma funcionalidad dentro del LMS. Los componentes descubren e invocan SW usando SOAP y un servidor UDDI (*Universal Description, Discovery, and Integration*), vía *middleware*. Estableciendo una capa entre el cliente de SW y el servidor. Posibilitando la opción de escribir clientes más inteligentes y dinámicos, al reducir la necesidad de dependencia dentro de los clientes de código duro, substituyéndolo con una sección amarilla de servicios.

Arquitectura orientada a servicio

AOCA esta basada en SOA (*Service Oriented Architecture*). SOA provee servicios y contenidos gracias al uso de la segunda generación para SW, SOAP, XML, Lenguaje de Definición de Servicios *Web* (*Web Services Definition Language*, WSDL por sus siglas en inglés) y UDDI. La ventaja de usar SOA reside en el desarrollo de servicios, que proveen interoperabilidad a nivel de aplicación. Esto se lleva a cabo debido a que los servicios ocultan los detalles del ambiente de ejecución, a través de interfaces que tienen todas las aplicaciones. Estos servicios son llamados SW y tienen la habilidad de emitir y consumir mensajes de datos XML, sin importar la plataforma, *middleware*, sistema operativo o hardware (ERL, 2004).

Marcado semántico

Las ontologías sirven para estandarizar y proveer interpretaciones para los materiales *Web*. Para hacer a los materiales entendibles para la computadora. Los recursos *Web* deben de ser meta-etiquetados de manera semántica. Tales

recursos posibilitan el razonamiento acerca de su contenido y servicios avanzados de consulta. Soportando la creación y mantenimiento de la ontología, ayudando a mapear entre diferentes ontologías. El marcado semántico es creado por un subsistema denominado *semantic_markup* de una manera automática. La Figura 2 muestra los componentes IRLCOOs meta-etiquetados.

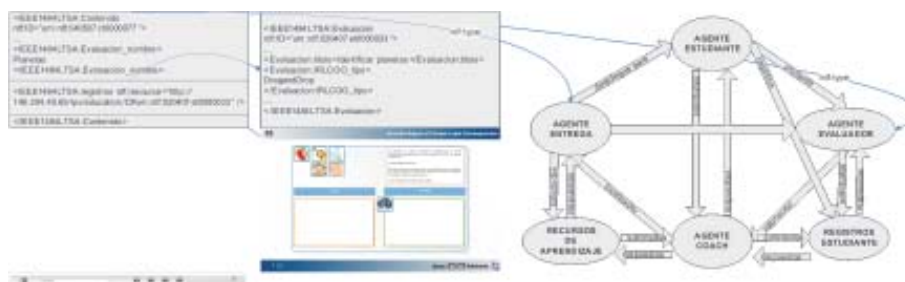


FIGURA 2 - Componentes IRLCOO con meta-etiquetas

Comunicación LMS con los componentes IRLCOO

El modelo de comunicación usa un modo asíncrono dentro de un Ambiente en Tiempo de Ejecución y enlaza al API de comunicación del LMS de ADL (ADL, 2004), usando como tecnologías de implementación AJAX (*Asynchronous JavaScript And XML*) (CRANE, 2006), y Struts (HOLMES, 2004). La comunicación del LMS de ADL se logra mediante su API, mediante un applet que sirve de puerto de comunicación incrustado y oculto en la página de los contenidos educativos. El API usa JavaScript y consiste de una colección de métodos estándar para comunicar los clientes con el LMS.

Los componentes IRLCOOs logran la comunicación con el API de comunicación del LMS de ADL mediante el comando *fscommand*, que posibilita la llamada a funciones JavaScript. Se utiliza el *framework* Struts para la parte de implementación del patrón de diseño Modelo-Vista-Controlador (*Model-View Controller*, MVC por sus siglas en inglés) de Java. Permitiendo independencia del modelo, la vista y el controlador. Este *framework* permite dar mantenibilidad a los sistemas, mejor desempeño y reusabilidad. La Figura 3 muestra el esquema de comunicación de los componentes IRLCOO con el LMS de ADL vía AJAX y Struts. Se muestra que la comunicación con el sistema que comienza cuando los componentes IRLCOO generan un evento mediante el comando *fscommand*, este comando permite a los componentes lograr comunicación con el navegador *Web* mediante JavaScript, al llamar a una función denominada: *FileName_DoFSCommand(command,args)*, la cual maneja el mensaje emitido desde el componente IRLCOO vía *fscommand*.

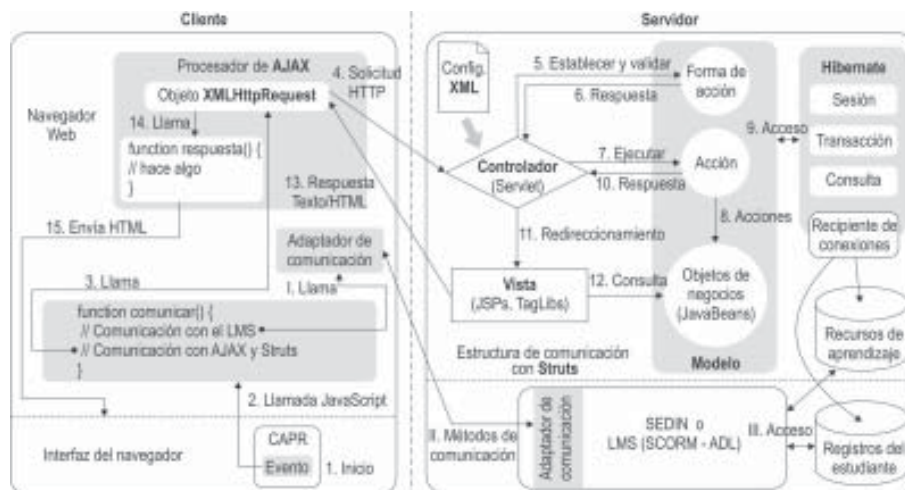


FIGURA 3 - Modelo de comunicación entre los IRLCOOs, LMS y los frameworks AJAX y Struts

La comunicación de AJAX y el *framework* Struts comienza cuando se invoca el método AJAX-Struts. Se crea una instancia del objeto *XMLHttpRequest*. Usando el método *open()*, la llamada es configurada, colocando el URL correspondiente con el método HTTP, típicamente GET o POST. La solicitud es disparada vía una llamada al método *send()*.

Este modelo de comunicación mejora el modelo para sistemas de WBE basados en el LMS de ADL, porque provee mejores capacidades de comunicación, interacción, interoperabilidad, seguridad y reusabilidad entre diferentes tecnologías. AJAX/Struts nos permiten generar un canal de comunicación bidireccional hacia el LMS basado en ADL.

Capa de persistencia de mapeado objeto-relacional

Hibernate es un ambicioso proyecto que tiene como meta solucionar el problema de manejo de persistencia de datos en Java. Dejando a los desarrolladores libres para concentrarse en los problemas del negocio. Hibernate es una solución no intrusiva. Casi todas las aplicaciones requieren datos persistentes, esta es una parte fundamental en el desarrollo de aplicaciones. Si un sistema no preserva datos de los usuarios estos tienden a ser de poca utilidad. La persistencia implica que los objetos pueden sobrevivir al finalizar el proceso de la aplicación, almacenándolos y recreándolos posteriormente. *Object/Relational Mapping* (ORM) es el nombre dado a la solución automatizada para resolver el problema de incompatibilidad (PEAK, 2006). La aplicación esta

construida con ORM en la mayoría de los subsistemas. La Figura 4 muestra el paquete/clase HIBERNATE.INTERPRETE para nuestro sistema FAQ (*Frequently Asked Questions*).

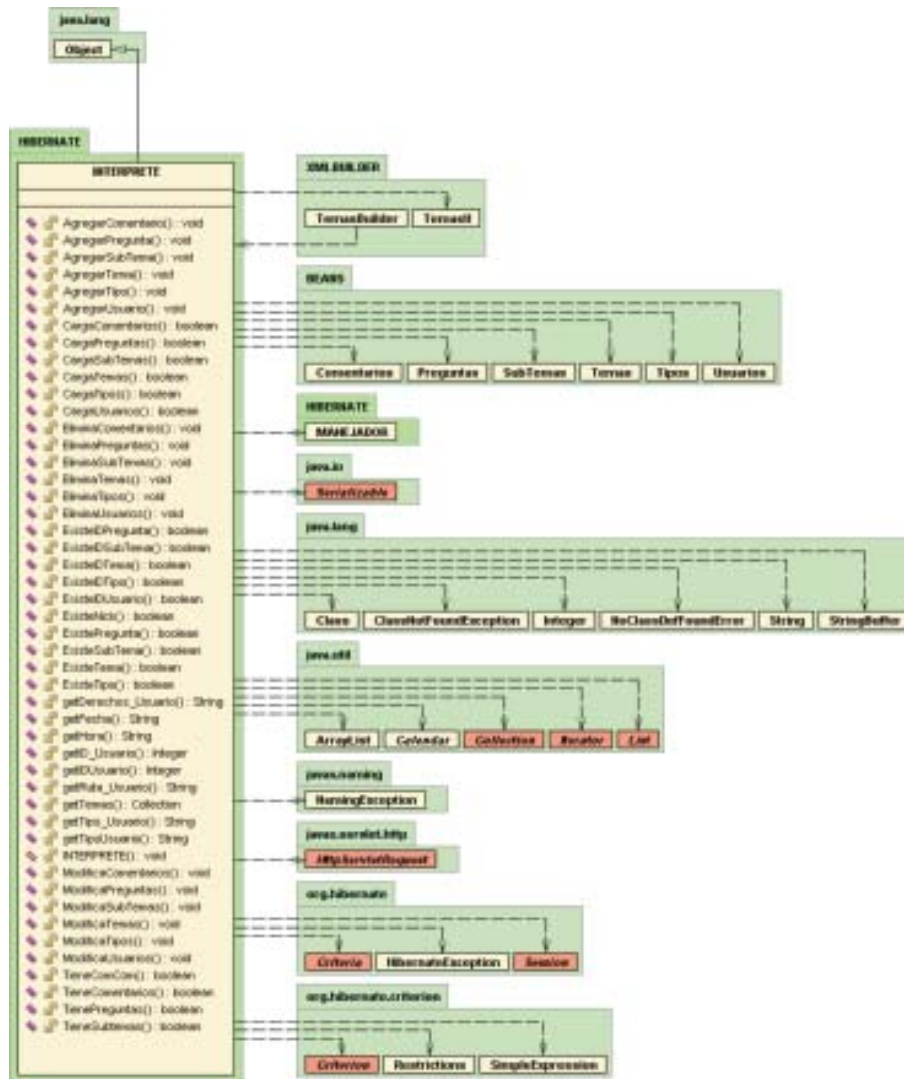


FIGURA 4 - Clase INTERPRETE para el subsistema FAQ

SiDeC y sistema de evaluación

En respuesta al problema del desarrollo de contenidos como es su elevada complejidad, se propone desarrollar una herramienta de Desarrollo de Aplicaciones Rápidas (*Rapid Application Development*, RAD por sus siglas en inglés) llamada SiDeC (Sistema de Desarrollo de eCursos), esta herramienta esta basada en los átomos de configuración descritos, denominados IRLCOOs. El SiDeC se basa en la arquitectura propuesta denominada AOCA, con la finalidad de automatizar el proceso de elaboración de contenidos multimedios complejos a los autores, además de posibilitar el manejo de múltiples meta-etiquetados para posibilitar inferencia sobre los materiales, estructurando y empaquetando el contenido multimedia en base a los IRLCOOs.

El SiDeC se usa para construir courseware basado en los componentes IRLCOOs. El proceso de elaboración depende de la estrategia pedagógica que se este utilizando, el sistema guía al profesor por una serie de formularios, donde puede ir llenando los contenidos y eligiendo los componentes IRLCOOs que mejor se adapten a sus necesidades de su curso, además de ir subiendo los multimedios asociados a los componentes utilizados. En este momento las estrategias pedagógicas implementadas están basadas en los Mapas Conceptuales (MC) y Aprendizaje Basado en Problemas (ABP), pero ya se esta trabajando en otras estrategias como son: Método de Casos, Método de Proyectos, etc.

La inclusión de elementos cognitivos, tales como los MC obedecen al patrón de diseño instruccional para el desarrollo de los cursos. Así, los cursos no únicamente tendrán cuestiones teóricas o prácticas, ya que incluirán un modelo mental acerca de los procesos mentales individuales. Los MC son esquemas que estructuran conceptos con el propósito de ayudar a los estudiantes a maximizar la adquisición de conocimiento. Un MC es una técnica gráfica usada durante el proceso de enseñanza-aprendizaje, entre otras formas instruccionales y estrategias de aprendizaje, y como un recurso esquemático o mapa de navegación.

La herramienta soporta la utilización de IRLCOOs para conformar cursos en línea. Los entregables son compatibles con la especificación de los modelos de meta-etiquetado SCORM 1.2 (contenido de agregación, secuenciación y navegación, y el ambiente en tiempo de ejecución) (ADL, 2004). Los meta-datos representan la descripción de los componentes y sus contenidos. El subsistema de meta-datos provee las herramientas necesarias para agregar el meta-etiquetado de los materiales de una manera sencilla conforme a los estándares del repositorio IMS/IEEE, esto se logra mediante el Modelo de Objeto Documento (*Document Object Model*, DOM por sus siglas en inglés) y Lenguaje de Transformaciones de Hojas de Estilo Extensible (*Extensible Stylesheet Language Transformations*, XLST por sus siglas en inglés). Se utilizaron para sus

implementaciones los *frameworks* denominados Xerces (XERCES, 2005) y Xalan (XALAN, 2005), mientras Xerces es un *parser* XML pero que no cuenta con métodos para hacer persistente el documento construido, Xalan es un *framework* que implementa XLST que nos permite hacer persistente los documentos construidos con Xerces. Estos dos *frameworks* de manera conjunta permiten conformar el subsistema de meta-etiquetado de la herramienta SiDeC.

El SiDeC propone una estructura de curso basada en la idea de un objeto de aprendizaje compuesto como una colección de Átomos de Aprendizaje Reutilizables (AAR) y Átomos de Información Reusable (AIR) (USKOV, 2003). Estos átomos son agrupados de forma conjunta para enseñar una tarea común basada en un objetivo de aprendizaje. Un AAR es una pieza atómica de aprendizaje que es construido por encima de un objetivo de aprendizaje. Cada uno de los AAR puede ser clasificado como: concepto, hecho, proceso o procedimiento. Los AARs proveen información del comportamiento del estudiante dentro del curso, p.e., trayectoria, tiempos y evaluaciones. Esta información se almacena dentro la BD de Registros del estudiante.

Los AIR por otro lado son piezas de información meta-etiquetadas que son construidas sobre los objetos. Esta puede contener siete diferentes tipos de información sobre los contenidos, tales como: revisión, introducción, importancia, objetivos, prerrequisitos, escenario, y entorno.

El sistema de evaluación para WBE esta diseñado bajo la misma arquitectura usada para SiDeC. La funcionalidad del sistema evaluador yace sobre el análisis del perfil del estudiante, el cual se construye a lo largo de las experiencias de aprendizaje. El perfil se basa en métricas que se extraen a lo largo de sus experiencias con el LMS. Estas métricas son almacenadas dentro de los registros del estudiante. La generación de nuevas secuencias de cursos es una función de los resultados obtenidos, además de contar con un nivel de adaptación.

El sistema de evaluación combina IRLCOOs especiales de evaluación, meta-etiquetado y una plataforma de agentes. La *Web Semántica* ofrece crear sistemas asistenciales para los usuarios humanos con la idea de facilitar sus actividades en línea. La *Web Semántica* ofrece muchas ventajas tales como: reducción de la complejidad, estandarización de las funcionalidades y atributos, definición de un conjunto de APIs para el despliegue de la plataforma (PASSIN, 2004).

Resumiendo el funcionamiento de los componentes del sistema de evaluación se muestran en la Figura 5. Básicamente el proceso del sistema de evaluación se lleva a cabo a través de dos fases. La primera fase es soportada por el LMS, y esta enfocada a presentar el curso y su estructura. Todas las acciones son registradas por medio de componentes de contenido IRLCOO que capturan las métricas de los estudiantes. La evaluación es hecha por componentes IRLCOO de evaluación y en algunos casos por IRLCOO de simulación. Estos procesos se despliegan por el LMS en conjunto con el *middleware* desarrollado basado en la versión empresarial de Java.

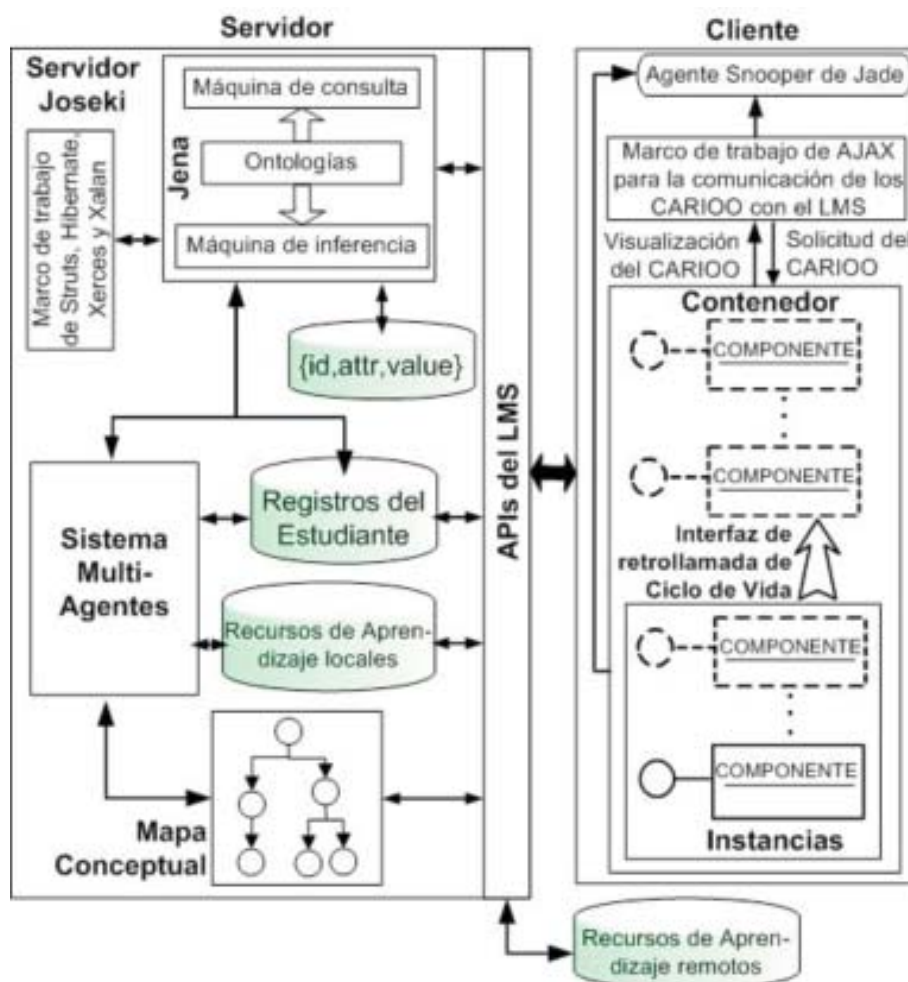


FIGURA 5 - Plataforma Web semántica para WBE

La segunda fase hace un análisis de los registros del estudiante, esta fase se lleva a cabo por el servidor basado el *middleware* del sistema MAS desarrollado en JADE. La plataforma de agentes posee siete agentes: *Snooper*, *Buffer*, *Learner*, *Evaluation*, *Delivering*, *Coach*, e *Info*. La idea fundamenta es automatizar el análisis del estudiante a través del Coach/Coach virtual, dando resultados parciales que pueden ser útiles para la retroalimentación hacia el estudiante. Estos agentes son implementados a través de programas Java Beans, que son incrustados dentro de la aplicación corriendo en ambos lados del sistema (Cliente/Servidor). El agente *Snooper* funciona como un disparador por medio de *INFORM performative*, que activa el sistema MAS del lado del servidor. Este agente es desplegado dentro de una página *Java Server Page* (JSP) como un Java

Bean. Una vez que la lección ha finalizado, la interfaz gráfica del usuario activa al agente *Snooper* y envía las métricas de evaluación (usando *Agents Communications Language*) para ser analizadas por la contraparte del sistema MAS del lado del servidor. El agente *Buffer* maneja la conexión y los mensajes desde el cliente. Ambas tareas usan el agente *Buffer* y envían información al agente *Coach*. Entonces el agente *Coach* solicita los registros del estudiante como son: preferencias de aprendizaje, trayectoria, información monitoreada, etc. El agente *Coach* analiza la información para determinar las verdaderas necesidades del estudiante. Solicitando el agente *Coach* los recursos de aprendizaje al agente *Delivering*, y este despliega los contenidos de aprendizaje propuestos por el agente *Coach* al estudiante. El agente *Delivering* envía los contenidos y evaluaciones de aprendizaje al agente *Learner* para su presentación. Estos agentes posibilitan la secuenciación dinámica al cambiar la secuencia del curso y las evaluaciones, mediante el cambio al archivo IMS Manifest. La secuencia esta definida por la estrategia basada en CM o ABP empleada, y el esquema de secuenciación de SCORM.

Una vez que la información es recibida (secuencia, tipo de IRLCOO, localización, etc.), esta se representa como una cadena, la cuál se construye dinámicamente en base del motor de inferencias conocido como Jena (2006) y servidor Joseki (2006), generando retroalimentación “*inteligente*” y dinámica.

La Figura 6 ilustra una vista integral del sistema descrito en este trabajo. En términos generales, la arquitectura de software es dividida dentro de cuatro capas: aplicación, agentes y componentes, base de datos y servidor.

PLATAFORMA WEB SEMÁNTICA

La arquitectura de la plataforma *Web Semántica*, está constituida por tres partes principales, como se muestra en la Figura 6. A continuación se describen brevemente estas tres piezas:

1. El *motor de consultas* que recibe las solicitudes y respuestas revisando el contenido de las bases de datos que fueron llenadas por el agente *Info* y el motor de inferencias.
2. El *manejador de base de datos* es la columna vertebral de todo el sistema. Recibe los hechos del agente *Info*, intercambia hechos como entradas y salidas con el motor de inferencia, y provee hechos al motor de consulta.
3. El *motor de inferencias* utiliza hechos y ontologías para derivar conocimiento adicional basado en hechos que únicamente está implicado. Este conocimiento es provisto por cada uno los estudiantes de forma implícita.

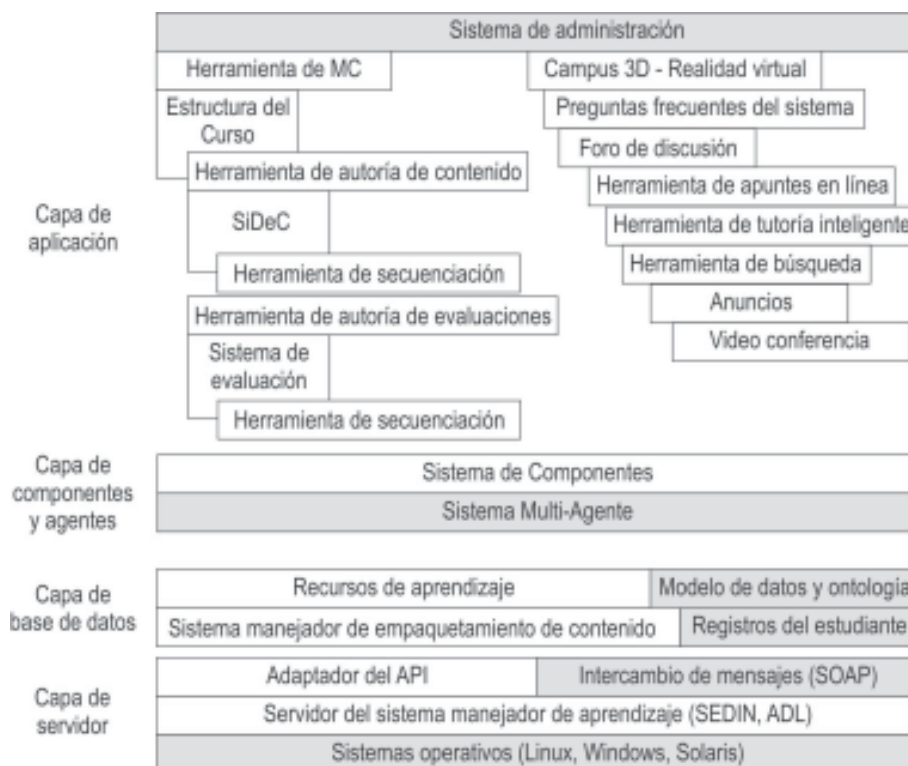


FIGURA 6 - Arquitectura del sistema mediante un patrón de capas

Se usa a las ontologías desarrolladas para compartir y reusar el conocimiento. El agente *Info* las usa para extraer hechos, usando el motor de inferencias para derivar conocimiento adicional implícito de los estudiantes y las posibles acciones a seguir.

Se uso JENA como motor de inferencias, que es un *framework* basado en Java para construir aplicaciones *Web* Semánticas. JENA provee un ambiente de desarrollo para: RDF, RDFS, OWL, SPARQL e incluye un motor de inferencias basado en reglas (ANTONIOU, 2004). Se utilizó JOSEKI como servidor *Web*, soportando: HTTP, SOAP, protocolo SPARQL, y RDF SPARQL.

CONCLUSIONES

Los sistemas de autoreo y evaluación descritos en este trabajo están basados en una arquitectura adaptativa e inteligente para el paradigma de Educación Basada en *Web*. Esta arquitectura está integrada por MAS, componentes IRLCOOs y una plataforma *Web Semántica*. La propuesta se enfoca a resolver problemas de: reusabilidad, accesibilidad, durabilidad, e interoperabilidad de contenidos de aprendizaje, los cuales son construidos en base a los componentes IRLCOO, siendo los principales elementos de construcción de material educativo para la entrega de materiales educativos de contenido y evaluación vía la plataforma.

El modelo de comunicación implementado se basa en el API de comunicación de ADL, pero además se le han agregado tecnologías que mejoran la comunicación como son: AJAX, Struts, Hibernate, IRLCOO, *Web Services* en conjunto con un servidor UDDI y *Web Semántica*. Todo lo anterior provee nuevas capacidades para el desarrollo de sistemas de WBE. El SiDeC y el sistema evaluador fueron desarrollados bajo el modelo AOCA para ayudar en la automatización y reducción de la complejidad del proceso de desarrollo de contenidos de aprendizaje.

El esquema ADL maneja secuenciación dinámica, composición, separación de contenido y navegación en el ambiente RTE. Mientras que nuestra arquitectura además de las ventajas de ADL adiciona las capacidades para reutilizar los componentes IRLCOOs para aplicaciones de escritorio y *Web*, posibilitando la creación de herramientas CASE que reutilicen los componentes de contenido y evaluación generados. Finalmente la arquitectura IEEE 1484 LTSA fue modificada, ya que los componentes permiten registrar ciertas métricas de los estudiantes de forma automática sin pasar por el proceso del Coach presencial.

REFERÊNCIAS

ADL. **Advanced distributed learning initiative**. 2007. Disponible en: <<http://www.adlnet.org>> Accesado en: 3 Enero 2004.

ANTONIOU, G.; VAN HARMELEN, F. **A semantic web primer**. USA: The MIT Press, 2004.

CRANE, D.; PASCARELLO, E.; JAMES, D. **Ajax in action**. Greenwich, USA: Manning Publications, 2006.

ERL, Thomas. **Service-oriented architecture: a field guide to integrating XML and web services**. USA: Prentice Hall, 2004.

IEEE. **IEEE 1484.1 LTSA**. New York: 2001. Disponible en: <<http://www.ieeeltsc.org/standards/1484-1-2003>>. Accesado en: 2 Abril 2007.

IMS. **Global IMS consortium**. Texas USA: IMS, 2008. Disponible en: <<http://www.imsproject.org>>. Accesado en: 3 Junio 2005.

HOLMES, J. **Struts: the complete reference**. California, USA: Mc Graw Hill – Osborne Publications, 2004.

JENA. **Jena: a semantic web framework for java**. Disponible en: <<http://jena.sourceforge.net/>>. Accesado en: 1 Abril 2006.

JOSEKI. **Joseki: a SPARQL server for Jena**. Disponible en: <<http://www.joseki.org/>>. Accesado en: 7 Abril 2006.

MACROMEDIA. **Macromedia inc.** Disponible en: <<http://www.macromedia.com>>. Accesado en: 30 Agosto 2005.

PASSIN, T. **Explorer's guide to semantic web**. 2004. Greenwich, USA: Manning Publications. 2004.

PEAK, P.; HEUDECKER, N. **Hibernate quickly**. Greenwich, USA: Manning Publications, 2006

PEREDO, R.; BALLADARES, L.; SHEREMETOV, L. Development of intelligent reusable learning objects for web-based education systems. **Expert Systems With Applications**, Oxford, v. 28, n. 2, p. 273-283, Feb. 2005.

RDF. **RDF specification**. 2002. Disponible en: <<http://www.w3.org/RDF/default.htm>>. Accesado en: 17 Febrero 2007.

SZYPERSKI, C. **Component software: beyond object-oriented programming**. New York, USA: Addison-Wesley, 2002.

USKOV, V.; USKOV M. Reusable learning objects approach to Web-Based Education. **International Journal of Computer and Applications**, Canada, v. 25, n. 3, 2003.

XALAN. **Xalan Java also implements the javax**. 2005. Version 2.0. Disponible en: <<http://xml.apache.org/xalan-j/>>. Accesado en: 9 Enero 2005.

XERCES. **XML Schema**: provides validation capabilities that weren't available with Document Type Definitions. Xerces-Java 2.0. 2005. Disponible en: <<http://xerces.apache.org/xerces-j/>>. Accesado en: 7 Febrero 2005.

XML. **Extensible Markup Language (XML)**. 2003. Disponible en: <<http://www.w3.org/XML>>. Accesado en: 21 Julio 2003.

WANG, Andy; QIAN, Kai. **Component-oriented programming**. Georgia, USA: John Wiley & Sons, Inc., 2005.

Recebido: 18/02/2008

Received: 02/18/2008

Aprovado: 25/03/2008

Approved: 03/25/2008