# Laboratorio Virtual de Programación Java basado en el paradigma de Educación Basada en Web

**R. Peredo Valderrama, J. C. Caravantes Ramírez, A. Canales Cruz**
**Laboratorio de Inteligencia Artificial, Centro de Investigación en Computación, IPN, México, D.F.**
**Av. Juan de Dios Bátiz Esq. Othón de Mendizabal S/N**
**Col. Nueva Industrial Vallejo, 07738, México, D.F.**
**peredo@cic.ipn.mx, caravanjc@yahoo.com, alejandro_canales@cuaed.unam.mx**

**RESUMEN**

El presente trabajo expone un sistema de Laboratorio Virtual de Programación Java (LVPJ), que posibilita realizar prácticas de forma remota, y elaboración de materiales educativos didácticos para prácticas de alta calidad, basadas en el paradigma de Educación Basada en *Web* (*Web-Based Education*, WBE por sus siglas en inglés). El desarrollo del Laboratorio Virtual de programación se realizo utilizando tecnologías de última generación, como son: Struts, Hibernate, ActionScript 3.0, JADE y AJAX. Se utilizaron componentes multimedia denominados: Componentes de Aprendizaje Reusables Inteligentes Orientados a Objetos (*Intelligent Reusable Learning Components Object Oriented*, IRLCOO por sus siglas en inglés), implementando una Aplicación Rica en Internet (*Rich Internet Application*, RIA por sus siglas en inglés). El núcleo de este sistema es un compilador en línea para código Java, posibilitando compilar bloques de código y generar los archivos compilados correspondientes, así como visualizar los resultados por medio del navegador *Web*. El sistema cuenta con una Interfaz Gráfica de Usuario (*Graphical User Interface*, GUI por sus siglas en inglés) avanzada con la finalidad de reducir la complejidad. Los alumnos pueden evaluar sus conocimientos mediante un sistema de evaluación y una retroalimentación dinámica basada en un Sistema Multi-Agente (*Multi-Agent System*, MAS por sus siglas en inglés).

**Palabras Claves:** Laboratorio Virtual, WBE, Componentes, Agentes y RIA

## 1. INTRODUCCIÓN

La enseñanza virtual se plantea como una solución al problema de crecimiento exponencial de la sociedad actual, teniendo un conjunto de desafíos y una búsqueda de nuevas soluciones pedagógicas/tecnológicas dentro del proceso enseñanza/aprendizaje, donde las Tecnologías de la Información avanzadas están tomando un papel cada día más importante. Una de las definiciones más aceptadas que define el paradigma de educación virtual es la siguiente:

— "*El uso de computadoras y comunicaciones en diferentes escenarios de aprendizaje* [1]."

El desarrollo de laboratorios virtuales para el paradigma WBE es una forma de generar experiencias enriquecedoras de aprendizaje donde quiera y cuando quiera, siempre que se cuente con una conexión a la *Web*, con la finalidad de proveer a los estudiantes con experiencias lo más cercano posible a la realidad. De aquí la necesidad de contar con un laboratorio virtual, que ayude al estudiante a practicar y así mejorar sus resultados en el mundo real. Ya que en algunos casos muchas instituciones lamentablemente no cuentan con los recursos humanos y computacionales para proveer de prácticas tradicionales en los laboratorios a los estudiantes, privándoles de estas experiencias enriquecedoras en muchos sentidos. En las prácticas tradicionales el instructor tiene que explicar todos los pasos de la práctica detalladamente, incluyendo en algunos casos las medidas de seguridad pertinentes. Para posteriormente pasar a la configuración y operación paso por paso, tomando demasiado tiempo. En muchos casos siguiendo la metodología anteriormente descrita no se obtienen los resultados esperados, debido a la mala configuración del equipo, errores en el seguimiento de la práctica, falta de compresión de la misma, tiempo insuficiente para dar un seguimiento personalizado a cada estudiante o equipo de trabajo, entre otras [2]. El uso del paradigma de WBE ofrece dos ventajas importantes a tomarse en cuenta en la implementación de los laboratorios virtuales que a continuación se mencionan:

— El estudiante puede avanzar a su propio ritmo.
— Personalización de los materiales de estudio de acuerdo a sus necesidades en tiempo de ejecución.

El paradigma WBE ofrece oportunidades y desafíos para todos los involucrados. Uno de los aspectos importantes que ha extendido su uso de forma importante ha sido la reducción de costos por licenciamiento, ya que tanto los servidores que manejan el Protocolo de Transporte de HiperTexto (*Hyper Text Transport Protocol*, HTTP por sus siglas en inglés), como los navegadores *Web*, clientes por excelencia usados en su arquitectura son gratuitos. Las principales líneas de investigación a nivel internacional de la WBE son: reusabilidad, accesibilidad, durabilidad e interoperabilidad de materiales didácticos y su uso en ambientes virtuales de aprendizaje. Siendo las principales iniciativas a nivel internacional las siguientes: Iniciativa de Conocimiento Abierta (*Open Initiative Knowledge*, OKI por sus siglas en inglés), Aprendizaje Distribuido Avanzado (*Advanced Distributed Learning*, ADL por sus siglas en inglés) y Consorcio de Aprendizaje Global (*Global Learning Consortium*, GLC por sus siglas en inglés). Los líderes de los dos últimos proyectos han trabajan de forma vinculada en la producción de recursos de aprendizaje innovadores [3-5].

En este trabajo se presenta el desarrollo de un LVPJ en línea para el paradigma de WBE y los componentes de software. Los componentes utilizados los denominamos Componentes de

Aprendizaje Reusables Inteligentes Orientados a Objetos (*Intelligent Reusable Learning Components Object Oriented*, IRLCOO por sus siglas en inglés) [6], que consumen Servicios *Web* (*Web Service*, WS por sus siglas en inglés), implementados con ActionScript 3.0 [7], para conformar una plataforma RIA y utilizar un Sistema Multi Agentes (*Multi-Agent System*, MAS por sus siglas en inglés), para crear un subsistema asistencial de apoyo implementado con el *framework Java Agent DEvelopment framework* (JADE) versión 3.6 [8]. El sistema permite elaborar material didáctico interactivo para prácticas de programación Java para WBE, pero posibilitando su uso en otros modelos de enseñanza. Las principales ventajas que ofrece el sistema son: reducción de la complejidad de elaboración de prácticas de LVPJ al automatizar los procesos, reuso de los materiales educativos interactivos, mantenibilidad del sistema, y subsistema de apoyo inteligente basado en agentes.

## 2. COMPONENTES DE INTERFAZ Y EVALUACIÓN IRLCOO PARA EL LVPJ

### 2.1 Introducción a los componentes de software

La arquitectura de software ha ido incrementando su reconocimiento como un elemento fundamental para construir grandes y complejos sistemas de software. La Programación Orientada a Componentes (*Component Oriented Programming*, COP por sus siglas en inglés) ha ido ganando adeptos en los últimos años de forma importante. Esto ha permitido retomar la idea de reuso de los componentes de software, teniendo como base de éxito la creación de nuevas y variadas tecnologías de componentes. El paradigma COP ha permitido que las líneas de producción de software basada en componentes reciban nueva atención. A principios de los 90's, Microsoft planteo el Modelo Objeto Componente (*Component Object Model*, COM por sus siglas en inglés) [9], como una manera de construir documentos complejos, desarrollando la tecnología denominada Objeto Enlazado y Embebido (*Object Linking Embebed*, OLE por sus siglas en inglés) [10], que fue posteriormente aplicada a Intercambio de Datos Dinámico (*Dynamic Data Exchange*, DDE por sus siglas en inglés) [11], y en el desarrollo de componentes de Visual Basic, esta tecnología permanece hasta el día de hoy pero bajo otra denominación por cuestiones comerciales, llamándose actualmente ActiveX [10]. Durante 1996 el Grupo de Manejo de Objeto (*Object Management Group*, OMG por sus siglas en inglés) definió el modelo de componentes CORBA (*Common Object Request Broker Architecture*) [12], con la finalidad de construir grandes aplicaciones distribuidas. Sun Microsystems desarrollo la tecnología denominada Java *Beans*, que básicamente definía su modelo de componentes para la construcción de componentes Java, la primera y segunda versión no tuvo gran aceptación pero con la liberación de la versión tres se ha retomado el camino hacia la aceptación de diferentes comunidades de desarrollo. Resumiendo la comunidad de ingeniería de software basada en componentes de software se encuentra trabajando de forma muy activa. En trabajos anteriores se ha mencionado una buena definición de lo que es un componente de software [13-15].

### 2.2 Una definición de componente de software

Como se comento al final del punto anterior en trabajos anteriores se ha mencionado la definición más importante de lo que es un componente de software [13-15], esta definición es ampliamente aceptada, debido a que proviene del que muchos consideran el padre del modelo orientado a componentes de software Clemens Szyperski. Pero debido a su importancia dentro del trabajo se mencionará a continuación:

— "Un componente de software es una unidad de composición con interfaces especificadas contractualmente y dependencias de contexto explícito. Un componente de software puede ser entregado independientemente y es tema de composición para terceras partes [15]."

Esta definición toma una gran importancia en el desarrollo del proyecto debido a que es una aplicación basada en componentes de software, obteniendo los beneficios de los mismos. El LVPJ reduce la complejidad de elaboración de prácticas de programación Java para un ambiente WBE, aumenta la reusabilidad de los materiales educativos didácticos desarrollados, permitiendo reutilizarlos en diferentes contextos, y ofreciendo una buena mantenibilidad de los mismos, ya que permite actualizarlos de forma sencilla simplemente respetando la interfaz anterior, pero posibilitando agregar nuevas funcionalidades manteniendo la compatibilidad con componentes anteriores. Además de posibilitar la agregación de nuevos componentes de forma transparente para el profesor, ya que es posible modificar el contenedor y agregar nuevos componentes sin necesidad de modificar todo el proyecto y los materiales educativos didácticos.

### 2.3 IRLCOO versión 1.0

Los Objetos de Aprendizaje Reusables (*Reusable Learning Objects*, RLO por sus siglas en inglés) son una de las piezas centrales de los sistemas de WBE actuales, para el desarrollo de contenidos educacionales, por lo cual es fundamental su desarrollo. Nuestros primeros intentos de implementación de RLO fueron usando Flash de Macromedia y su lenguaje de programación denominado ActionScript 1.0, siendo usados como herramienta de desarrollo para generar un núcleo básico de los componentes de contenido en la forma de Componentes de Aprendizaje Reusables (*Reusable Learning Components*, RLC por sus siglas en inglés). Los componentes Flash son interpretados por un cliente denominado *Flash Player* (FP), que es ampliamente utilizando por medio de un *plug-in* en los diferentes navegadores *Web*. Estos componentes desarrollados permitieron manejar multimedios, además de cargar ciertos tipos de medios en tiempo de ejecución, posibilitando tener un ambiente programable y adaptivo a las necesidades del estudiante en tiempo de ejecución. Otro aspecto importante a señalar es la carga de los componentes en diferentes niveles, permitiendo implementar la separación de los contenidos de la navegación, esto es importante porque permite la reutilización de los componentes. ADL hace la separación de contenidos de la navegación por medio de su Ambiente en Tiempo de Ejecución (*Run Time Enviroment*, RTE por sus siglas en inglés), teniendo como problema principal que los materiales educativos tienen que ser montados en el RTE, además los contenidos deben de cumplir con un conjunto de reglas para lograr la separación contenido/navegación. Flash cuenta con una biblioteca de los denominado Clips Inteligentes (*Smart Clips*, SM por sus siglas en inglés) para los elementos de aprendizaje denominados Interacciones de Aprendizaje (*Learning Interactions*, LI por sus siglas en inglés), siendo la idea original generar una librería multimedia de RLC para Entrenamiento Basado en Computadora (*Computer Based Training*, CBT por sus siglas en inglés), pero más tarde esta idea ha evolucionando

hacia el paradigma WBE. En los primeros diseños la separación entre contenidos/navegación de los componentes era usan diferentes niveles dentro del FP, permitiendo generar componentes especializados, pequeños, reusables, y capaces de integrarlos en tiempo de ejecución dentro de un componente que se denomina contenedor [16].

### 2.4 IRLCOO versión 2.0

La liberación de ActionScript versión 2.0 posibilitó redefinir los RLC desde un modelo procedimental, hacia una primera implementación de un modelo de Programación Orientada a Objetos (*Object Oriented Programming*, OOP por sus siglas en inglés), lo que se reflejó en el rediseño e implementación de los componentes, redefiniéndolos a Componentes de Aprendizaje Reusables Orientados a Objetos (*Reusable Learning Components Object Oriented*, RLCOO por sus siglas en inglés). Pasando a implementar una Interfaz de Programación de Aplicaciones (*Application Programming Interface*, API por sus siglas en inglés), para un conjunto de atributos y métodos, enfocándose a resolver los problemas comunes más importantes como son los siguientes: API - ADL de comunicación con el Sistema Manejador de Aprendizaje (*Learning Management System*, LMS por sus siglas en inglés), API de comunicación MAS, API de carga dinámica de Objetos de Contenido Compartido (*Sharable Content Object*, SCO por sus siglas en inglés), y API de composición SCO [6].

El siguiente paso en su momento fue la implementación de inteligencia dentro de los componentes RLCOO, redefiniendo un subconjunto a RLCOO inteligentes (*Intelligent Reusable Learning Components Object Oriented*, IRLCOO por sus siglas en inglés), constituyendo componentes avanzados, agregando retroalimentación hacia los usuarios en tiempo de ejecución y reconfiguración dinámica, conformando las nuevas APIs de retroalimentación y reconfiguración dinámica. Uno de los primeros intentos de agregar inteligencia a los componentes IRLCOO fue por medio de una red neuronal del tipo Memoria Asociativa Bidireccional (*Bidirectional Asociative Memory*, BAM por sus siglas en inglés), conformando la API - BAM [16]. En las subsiguientes versiones se ha optado por la utilización de agentes de software, en la forma de un MAS del lado del servidor [17].

### 2.5 (I)RLCOO con ActionScript 3.0

Con la liberación de ActionScript 3.0 se ha tenido que hacer una reestructuración completa de los componentes (I)RLCOO, debido a que este lenguaje ha tenido una completa reingeniería para su total transformación en un lenguaje orientado a objetos, con la finalidad de resolver problemas comunes para el desarrollo de aplicaciones de Flash y Flex [18]. Esto con las finalidad de estructurar código como lo hacen la mayoría de los lenguajes para OOP, haciéndolo fácil de usar y de aprender. Soportando abstracción, herencia, y polimorfismo, siendo estas las características soportadas por un lenguaje para OOP, además de posibilitar el uso de patrones de diseño de software. A continuación se mencionan las principales reestructuraciones de los componentes (I)RLCOO con ActionScritpt 3.0.

**2.5.1 API de comunicación LMS 2.0: u**na de las API que sufrió más cambios fue la comunicación con la API del LMS – ADL [4]. En las versiones 1 y 2 de los componentes (I)RLCOO la comunicación hacia el mundo exterior ((I)RLCOO –> navegador *Web* –> Applet –>LMS, se realizaba por medio de la función de ActionScript *fscommand* que permitía la comunicación de los componentes con el navegador mediante funciones de JavaScript. La función ya no se utiliza en la última versión de ActionScript, siendo sustituida por la clase *ExternalInterface*. Esta comunicación de los componentes (I)RLCOO se realizaba por medio de JavaScript, debido a que es la manera de comunicarse del LMS - ADL usado JavaScript para comunicarse con el cliente, por medio de un Applet incrustado en el cliente para servir como un API-adaptador de comunicación con el LMS - ADL, por lo cual toma especial importancia para la comunicación de nuestros componentes (I)RLCOO.

La clase *ExternalInterface* permite llamar a funciones JavaScript desde los componentes (I)RLCOO, con la clase la función JavaScript puede retornar un valor a los componentes. El llamado a las funciones JavaScript desde los componentes se hace por medio del método estático *ExternalInterface.call*("LMSSetValue", "cmi.core.score.raw" +score), donde el anterior código establece el *score* del estudiante en el LMS - ADL. Para la invocación de las funcionalidades de los componentes (I)RLCOO desde JavaScript, primero se procede a registrar la función dentro de los componentes, para posteriormente llamar a la función de JavaScript vía el objeto Flash. El registro de la función ActionScript se puede realizar por medio del método estático *ExternalInterface.addCallback()*. Una vez registrado el procedimiento para llamarlo desde JavaScript, se llama a la funcionalidad deseada como un método del objeto Flash referenciando el ID del objeto ActiveX. La siguiente línea muestra el llamado a la funcionalidad *LMSGetValue()*, para obtener la calificación del estudiante en un proceso de evaluación.

```
document.getElementById("flashObjectID").LMSGetV
alue();
```

Por lo anterior se tuvo que hacer una reestructuración de este API - ADL de comunicación con el LMS y los componentes (I)RLCOO.

**2.5.2 Rediseño de la arquitectura de los componentes IRLCOO con ActionScript 3.0:** ActionScript 3.0 plantea una arquitectura de despliegue que afecta a los componentes IRLCOO denominada: arquitectura de despliegue. Un término que describe una jerarquía de objetos visuales y sus relaciones. Otro punto fundamental es el manejo de eventos en ActionScript 3.0 respecto a la versión anterior. Algunos eventos se podían poner en botones e instancias de los *MovieClip*, pero esto ocasionaba conflictos, ya que estos eventos aislados modificaban el estado de la aplicación. En ActionScript 3.0 se usan escuchadores de eventos para disparar los métodos. El código siguiente muestra la clase *XMLDisplay*, que es la encargada de cargar las preguntas del examen, como se puede ver, ya se tiene un soporte de paquetes *cic.laboratoriovirtual.utility.xml*, donde se tienen organizadas las clases en bloques funcionales. Posteriormente viene la sección de importación de bibliotecas, para pasar a continuación a la sección de definición de la clase *XMLDisplay*, la cual extiende de *MovieClip*. La carga de la información de los archivos XML (*eXtensible Markup Language*) se realizaba en las versiones anteriores mediante un objeto denominado *XML*, ahora se hace por medio de un par de objetos *URLLoader* y *URLRequest*, instanciando el primero y agregándole un escuchador de eventos cuando se haya completado

*onXMLLoaded*. Enseguida se invoca el método *load* del *URLLoader* con un objeto *URLRequest*, disparando el evento y ejecutando el método *onXMLLoaded*, ejecutando una acción personalizada y realizando varias acciones personalizadas dependiendo del tipo de pregunta a utilizarse en el momento de cargar el contenedor con examen/preguntas/navegación. Además la clase *XMLDisplay* en esta versión permite definir los niveles de accesibilidad de la clase.

El código siguiente muestra la nueva estructura de paquetes/clases del LVPJ, además del uso de los escuchadores para disparar los métodos. Con lo que se demuestra el uso del nuevo modelo de programación en ActionScript versión 3.0, donde se ve un lenguaje de OOP más maduro y consistente, parecido a Java, C++, C#, entre otros.

```
package cic.laboratoriovirtual.utility.xml
{
import flash.display.*;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.xml.*;
import flash.events.*;
public class XMLDisplay extends MovieClip
{
public function XMLDisplay() {
var _loader:URLLoader = new URLLoader();
_loader.addEventListener(Event.COMPLETE,
onXMLLoaded);
_loader.load(new
URLRequest(''examen/pregunta.xml'')); }
public function
onXMLLoaded(event:Event):void {
var loader:URLLoader =
URLLoader(event.currentTarget);
var _xml:XML = new XML(loader.data);
...
}}}
```

**2.5.3 Componentes (I)RLCOO de interfaz del LVPJ:** la idea general del LVPJ, es desarrollar prácticas de laboratorio de programación Java interactivas como materiales educativos didácticos para *Web*, con la opción de posibilitar a los profesores y alumnos la opción de compilar en línea. El sistema cuenta con tres usuarios principales: administradores, profesores y estudiantes. Donde el rol del administrador son altas, bajas y cambios de los usuarios del sistema. El profesor es el encargado de desarrollar las prácticas de laboratorio de programación Java interactivas como materiales educativos didácticos, con compilación en línea. Los estudiantes pueden hacer uso de los materiales educativos didácticos, resolver prácticas del Laboratorio Java con compilación en línea, tanto para sus prácticas como para sus evaluaciones. La Figura 1 muestra el ambiente del profesor, donde este ha invocado al componente de codificación, para la captura de las prácticas de laboratorio de los estudiantes, el sistema permite ir ingresando el código y cuenta con un sistema de ayuda que permite auxiliar al profesor en la codificación, que al ingresar el código en el sistema se va mostrando los objetos, métodos y atributos, este sistema fue implementado usando AJAX (*Asynchronous JavaScript And XML*), por lo que se estableció un sistema de comunicación bidireccional, la ventaja fundamental de AJAX es que no es necesario refrescar la página, esta es básicamente un tipo de aplicación que se denomina de autocompletado. La Figura 2 muestra el ambiente del profesor mediante los componentes IRCLOO de falso/verdadero, en conjunto con los componentes de codificación y navegación dentro del contenedor y que a su vez conforman una de las evaluaciones de falso/verdadero del sistema.

## 3. ARQUITECTURA DEL LVPJ

### 3.1 Arquitectura IEEE 1484 - LTSA

La Arquitectura de Sistemas Tecnológicos de Aprendizaje (*Learning Technology Systems Architecture*, LTSA por sus siglas en inglés), provee una referencia de trabajo para comprender los sistemas de aprendizaje a distancia existentes y futuros, buscando la reusabilidad, interoperabilidad y portabilidad del LMS. LTSA se planteo para Sistemas de Manejo de Contenido de Aprendizaje (*Learning Content Management System*, LCMS por sus siglas en inglés). La arquitectura es flexible para adaptarse a las nuevas tecnologías y sistemas de aprendizaje. Además de mostrar los elementos fundamentales de todo el sistema de aprendizaje, por lo menos para los próximos 10 años. Esta arquitectura está enmarcada dentro del proyecto 1484 de la IEEE, constituyendo un estándar para el aprendizaje a distancia propuesto por el Comité de Estándares Tecnológicos de Aprendizaje (*Learning Technology Standards Committee*, LTSC por sus siglas en inglés) de la IEEE *Computer Society*. Esta arquitectura tiene el soporte de varias trascendentales instituciones, el modelo permiten definir: análisis, diseño e implementación de alto nivel, de las tecnologías para el aprendizaje y entrenamiento basado en computadora, sistemas de apoyo para la presentación de materiales educativos electrónicos, enseñanza asistida por computadora, tutoriales inteligentes, tecnologías de entrenamiento y educación [19-21].

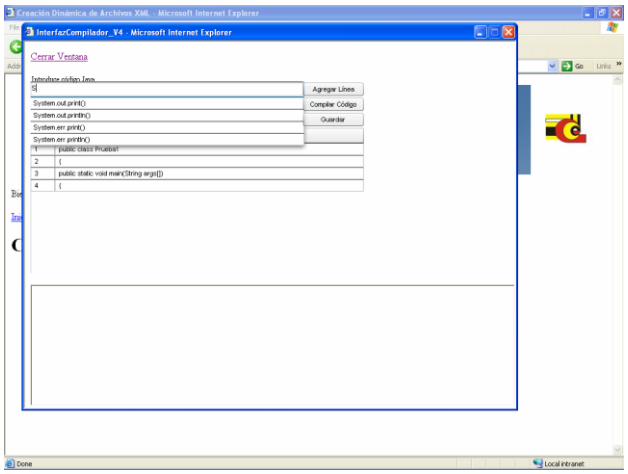

Figura 1. Componente grid de codificación del profesor para prácticas de laboratorio.

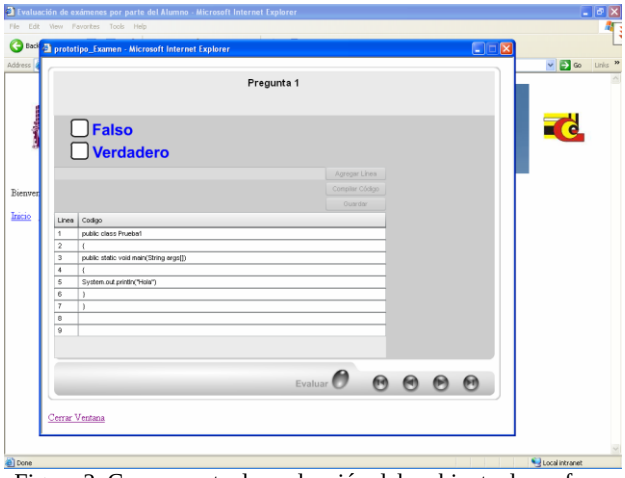

Figura 2. Componente de evaluación del ambiente de profesor constituido por los subcomponentes: falso/verdadero, codificación y navegación.

### 3.2 La arquitectura del LVPJ

El sistema del LVPJ se baso en la arquitectura LTSA de IEEE pero los procesos del sistema se basaron en agentes. Además se implemento una modificación a la arquitectura usada ya en otros de nuestros sistemas [17], consistiendo en un registro automático de preferencias de los estudiantes. Esto permite un registro de preferencias del usuario desde los componentes (I)RLCOO de forma automática, hacia los registros del usuario, sin pasar por el agente Coach, con el fin de registrar algunas métricas del estudiante de forma automática como son: la trayectoria del estudiante en los materiales de contenido, tareas completadas, tareas incompletas, uso de herramientas de colaboración, actividades completadas, actividades incompletas, estadísticas de materiales más visitados, tiempo en cada pregunta de la evaluación, tiempo total de la evaluación, trayectoria del estudiante en los materiales de evaluación, etc. Todas estas métricas del estudiante son almacenadas en los registros del usuario, para su posterior análisis por medio del sistema MAS, con la finalidad de generar una retroalimentación personalizada, identificación de posibles problemas de aprendizaje, y reconfiguración de trayectoria de aprendizaje en función de los resultados.

### 3.3 Patrón de composición

El patrón de composición provee una fuerte solución para construir sistemas complejos basados en componentes. Los sistemas complejos están constituidos de componentes más pequeños, denominándose dos tipos básicos: indivisibles y compuestos, contenidos dentro de un contenedor. Una de las ventajas más importantes de este patrón de diseño es que permite a los clientes tratar a los componentes indivisibles y compuestos de la misma manera, permitiendo simplificar la API para su manejo de forma significativa. Permitiéndonos definir operaciones independientemente del tipo para el manejo de los componentes (I)RLCOO: cargar_IRLCOO, descargar_IRLCOO, agregar_IRLCOO, remover_IRLCOO, etc.

### 3.4 Implementación del patrón de composición de los componentes (I)RLCOO

Los nodos que contienen otros componentes se denominan componentes compuestos (I)RLCOO. Los nodos IRLCOO son indivisibles y no pueden contener otros componentes. Estos componentes indivisibles constituyen una parte de la interfaz API – IRLCOO. Los componentes compuestos a su vez son utilizados por el contenedor de la aplicación *Web*, esta aplicación tiene una tres componentes compuestos que son: contenedor, (I)RLCOO, y navegación. Este esquema se utiliza en las diferentes vistas desplegadas hacia el usuario. La idea es hacer una separación del contenido de la navegación, en un esquema similar al esquema ADL. Logrando modificar los contenidos y la navegación en tiempo de ejecución de acuerdo a los resultados de los estudiantes.

### 3.5 Patrón Modelo-Vista-Controlador

Ya se ha empleado este patrón Modelo-Vista-Controlador (*Model-View-Controller*, MVC por sus siglas en inglés) en varios de nuestros proyectos [13-14], siendo fundamental en el desarrollo de arquitecturas reutilizables, consistentes y fácilmente mantenibles. Esto permite un ahorro de tiempo en el desarrollo de proyectos posteriores. Este patrón hace una separación de tres elementos: modelo, vista y controlador. Tiene varias ventajas como ya se han mencionado en un trabajo previo [14], pero solo mencionaremos la que consideramos la más importante que es la separación completa entre la lógica de negocios y la presentación.

### 3.6 Implementación del patrón MVC y subsistemas

Se empleo Struts como marco de trabajo de la fundación Apache, ya es un estándar de facto [22]. Struts ofrece muchas ventajas a los desarrolladores de aplicaciones *Web*, ya que implementa el patrón MVC, además contar con una gran cantidad de librerías para desarrollar aplicaciones *Web*.

De los subsistemas implementados usando Struts sobresalen tres que a continuación mencionaremos: Login, Upload y XML. El primero es la entrada al sistema, permitiendo validar si el usuario esta registrado en la base de datos del sistema, por medio de un Java *Bean* para el manejo del modelo. El segundo subsistema es para la transferencia de archivos multimedia de los materiales educativos y prácticas del Laboratorio Java, por medio de la librería *org.apache.struts.upload.FormFile*, para subirlos del lado del servidor. El tercer subsistema es el más complejo de los tres, básicamente se encarga de la creación y modificación de los diferentes archivos XML de las diferentes partes del LVPJ, mencionando a continuación solo los más importantes archivos XML: exámenes, preguntas, navegación, codificación, meta datos ADL. Se utilizaron dos *frameworks*, para la estructuración de los documentos XML, Xerces para implementar la API Modelo Objeto Documento (*Document Object Model*, DOM por sus sigla en inglés) [23-24], mientras que para la persistencia en archivos XML se utilizo Xalan, que es un *framework* que implementa Lenguaje de Transformación de Hojas de Estilo Extensible (*eXtensible Stylesheet Language Transformations*, XLST por sus siglas en inglés) [25].

El modelo MVC puede asociar diferentes vistas dentro de este patrón, el modelo necesita informar a todas las vistas asociadas que un cambio ha tomado lugar. Las vistas pueden ser complejas, con múltiples ventanas conteniendo cada una elementos de la interfaz. Las vistas utilizan el patrón de composición por medio de los (I)RLCOO, simplificando el proceso de actualización. Creando vistas complejas sin tener que enviar eventos de actualización para cada vista anidada. Los (I)RLCOO usan una Máquina Virtual (*Virtual Machine*, MV por sus siglas en inglés), posibilitando que las aplicaciones *Web*

puedan correr en múltiples plataformas necesitando solo la MV para su ejecución. La propuesta es que los componentes (I)RLCOO corran sobre una VM, posibilitando al igual que en Java se puedan ejecutar sobre múltiples plataformas de software/hardware, en nuestro caso la MV se denomina FP. La Figura 3 muestra un esquema de la implementación del patrón MVC con componentes (I)RLCOO incrustados corriendo dentro de un patrón MVC. Nuestra propuesta consiste en mezclar las vistas del usuario del patrón MVC en conjunción con el patrón de composición de los (I)RLCOO.

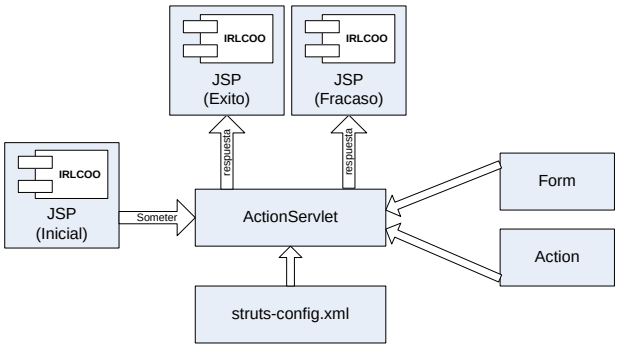


Figura 3. Implementación del patrón MVC con la VM y componentes IRLCOO incrustados.

### 3.7 Mapeado Objeto/Relacional

Hibernate es una solución al problema de persistencia de datos en Java. Mediante el manejo entre la aplicación y la base de datos relacional, permitiendo al desarrollador concentrarse en los problemas de la aplicación. Hibernate es una solución no intrusiva, lo que significa que no requiere seguir reglas específicas y patrones de diseño cuando se escribe la lógica de negocios y las clases de persistencia. Hibernate se integra de forma sencilla con aplicaciones anteriores y nuevas. En resumen es un *framework* Java para el Mapeado Objeto/Relacional (*Object/Relational Mapping*, ORM por sus siglas en inglés) [26].

La persistencia de la aplicación *Web* se manejo en forma conjunta con Hibernate y Struts, además de desarrollar varias clases que a continuación mencionaremos: *HandlingException, HibernateUtilities, StudenDAO*. Claro, en conjunto con las clases: *StudentActionForm* y *StudentAction*, y los archivos de configuración del proyecto *hibernate.cfg.xml* y *student.hbm.xml*. La clase *HandlingException* manipula las excepciones dentro del código Hibernate. La clase *HibernateUtil* inicializa Hibernate y obtiene acceso a la sesión de persistencia y operaciones de transacciones. La clase *StudentDAO* tiene todas las funcionalidades necesarias para que los datos del estudiante se puedan hacer persistentes en la base de datos mediante Hibernate. La clase *StudentActionForm* es un contenedor de los parámetros recibidos desde el estudiante. La clase *StudentAction* permite ejecutar las acciones del sistema en función de los resultados obtenidos. Se tienen dos archivos de configuración, el primero *hibernate.cfg.xml* realiza la configuración de Hibernate, el segundo *student.hbm.xml* permite personalizar el ORM. La Figura 4 muestra el diagrama de clases de la clase *StudentAction* de la capa de persistencia basada en Hibernate.

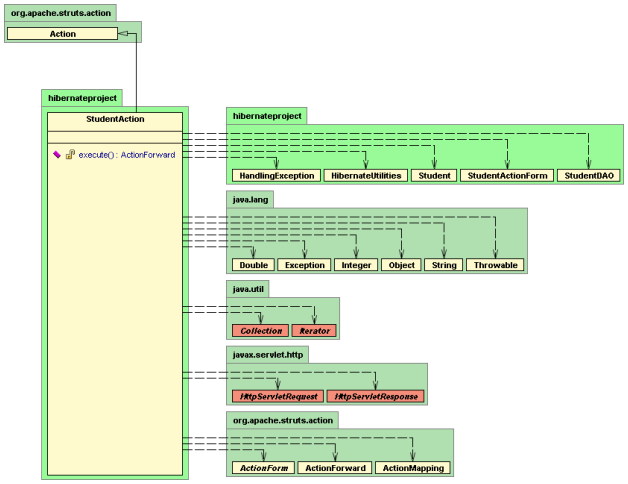


Figura 4. Diagrama de clases de *StudentAction*.

### 3.8 Resultados experimentales

La Figura 5 muestra un examen dentro del ambiente del estudiante, esta vista generada usando Struts se construye por el FP que constituye la MV del sistema, mostrándonos una vista de los componente (I)RLCOO dentro del contenedor de evaluación del LVPJ, esta evaluación está constituido de tres componentes que son: contenedor, (I)RLCOO de opción múltiple, e (I)RLCOO de navegación.

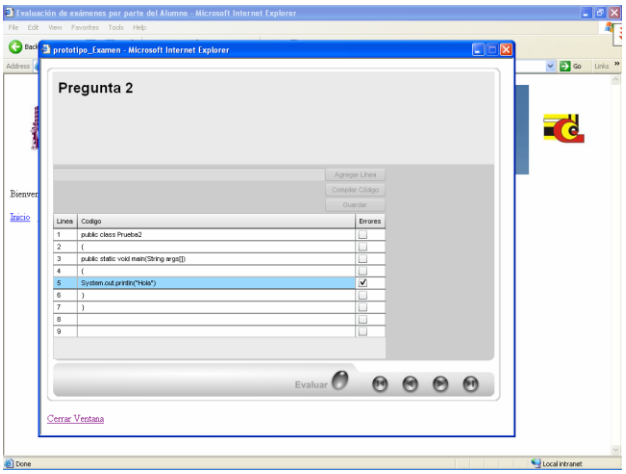


Figura 5. Evaluación del LVPJ de opción múltiple del ambiente del estudiante.

### 3.9 Sistema Multi-Agentes

Se implementó un MAS básico para el análisis de las acciones del estudiante para configurar su retroalimentación y reconfiguración dinámica. El sistema MAS se implemento utilizando el *framework* JADE versión 3.6 [8]. El MAS utiliza el paquete *jade.wrapper.gateway*, y más específicamente las clases: *JadeGateway* y *GatewayAgent*. El evento POST que lanza al MAS es disparado desde el navegador del estudiante, para posteriormente ser manejado por el *GateWayServlet* dependiendo del mensaje de acción (*sendmessage*) recibido. La acción crea un objeto *BlackBoardBean*, que será nuestro canal de comunicación entre el agente *ConexAgente* y el Servlet. El *AgenteDB* recibe el mensaje enviado por el *ConexAgente* y realiza la consulta a la base de datos, obtiene la respuesta y la envía al *ConexAgente*, que está en espera de dicha respuesta. El

*ConexAgente* escribe la respuesta sobre el *BlackBoardBean* y le envía una notificación al Servlet de que ya está la respuesta. Finalmente el Servlet envía la respuesta a la página JSP de respuesta del usuario. Los pasos anteriores se ven reflejados en la Figura 6 y la Figura 7 que muestran la arquitectura del MAS y el diagrama de clases del *ConexAgente* del sistema respectivamente.

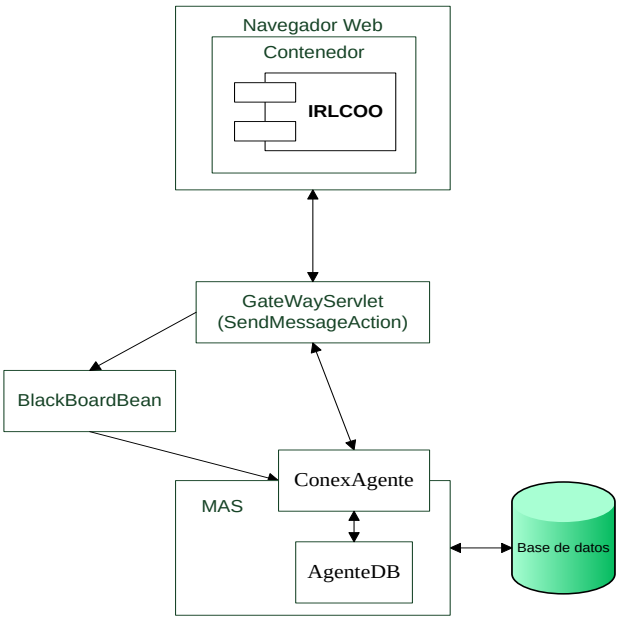


Figura 6. Diagrama a bloques del funcionamiento del MAS.

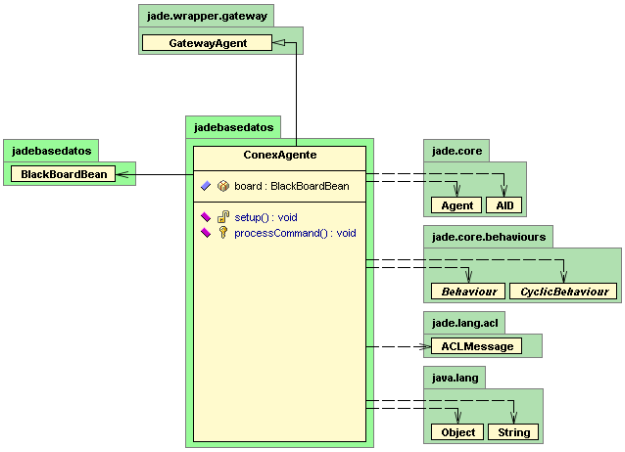


Figura 7. Diagrama de clases del agente ConexAgente.

## 4. CONCLUSIONES

El sistema permite automatizar la creación de materiales educativos didácticos, para prácticas de LVPJ, y permite disponer de un compilador en línea para los profesores/estudiantes con características avanzadas de un Entorno de Desarrollo Integrado (*Integrated Development Environment*, IDE por sus siglas en inglés). El sistema permite la captura de métricas del estudiante de forma automática, a través de una modificación a la arquitectura LTSA de la IEEE, la capa de persistencia se implementa por medio del *framework* Hibernate. La persistencia de ciertas métricas se hace de manera automática sin pasar por el proceso del *Coach*, mediante una arquitectura basada en el modelo MVC utilizando Struts y algunas utilerías del *middleware* (*Sevlets*/JSP/Java *Beans*), y componentes (I)RLCOO escritos con ActionScript 3.0.

El sistema basado en componentes tienen la habilidad de: responder al cambio rápidamente y una necesidad de reducir costos. Entre las ventajas que ofrece este sistema, sobresale fundamentalmente la del sistema de compilación en línea con un entorno IDE con características avanzadas para los profesores/estudiantes, resultando en una forma innovadora de usar la computadora, al proveer una forma automatizada para la creación de prácticas de un LVPJ y materiales educativos didácticos de una forma sencilla para el profesor, dándole a la WBE una nueva dimensión. El sistema mostrado está basado en el modelo de componentes de software, por lo que se encuentra listo para el cambio, permite reducir costos, incrementa la reutilización de materiales, fácil integración con otros componentes (I)IRLCOO, reducción de la complejidad y agrega una nueva dimensión en la interacción con la computadora para prácticas de un LVPJ. Se redujo el tamaño de la base de datos, esto se logro por medio del uso de los archivos XML como meta datos, además de permitir crecer al sistema de una forma sencilla, ya que el sistema carga por medio de una interfaz única los componentes, no teniendo que dar de alta ningún componente específico en la base de datos. Permitiendo implementar una especie de polimorfismo, con lo cual se pueden agregar nuevos componentes (I)RLCOO que se adapten a la interfaz, sin reconfigurar la arquitectura del sistema.

Reconocimientos. Los autores de este artículo agradecen al Instituto Politécnico Nacional (IPN) – Centro de Investigación en Computación (CIC) y a la Universidad Nacional Autónoma de México (UNAM) – Coordinación de Universidad Abierta y a Distancia (CUAED) por su apoyo parcial para este trabajo dentro del proyecto SIP del IPN: 20101278. Los autores desean reconocer a todos sus colegas y estudiantes que participaron en el diseño y desarrollo del software, y materiales de aprendizaje descritos en este artículo.

## 5. REFERENCIAS

[1] B. Collis, "Applications of Computer Communications in Education: An Overview", **IEEE Communications Magazine**, March 99, pp. 82-86.

[2] R. Kwan, R. Fox, F. T. Chan & P. Tsang, **Enhancing Learning Through Technology Research on Emerging Technologies and Pedagogies**, World Scientific Publishing Co. Pte. Ltd 2008.

[3] Open Knowledge Initiative, MIT, URL: http://web.mit.edu/oki/

[4] Advanced Distributed Learning Initiative, URL: http://www.adlnet.org

[5] Global IMS Learning Consortium, URL: http://www.imsproject.org/

[6] R. Peredo Valderrama, Leandro Balladares Ocaña, L. Sheremetov, "Development of Intelligent Reusable Learning Objects for Web-Based Education systems", **Expert Systems with Applications**, 28(2): 273-283, Pergamon Press, 2005.

[7] ActionScript 3.0 Language and Components Reference, Adobe, URL: http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/

[8] JADE Agent Platform, URL: http://jade.tilab.com/

[9] R. Brown, W. Baron, & W. D. Chadwick III, **Designing Solutions with COM+ Technologies**, Microsoft Press 2001.
[10] D. Chappell, **Understanding Active X and OLE**, Microsoft Press 1996.
[11] About Dynamic Data Exchange, URL: http://msdn.microsoft.com/en-us/library/ms648774.aspx
[12] J. Siegel, **Quick CORBA 3,** John Wiley & Sons 2001.
[13] R. Peredo Valderrama, I. Peredo Valderrama, L. Balladares Ocaña, "Sistema evaluador usando *Web* semántica para educación basada en *Web"*, **5ta. Conferencia Iberoamericana en Sistemas, Cibernética e Informática** (CISCI 2006), IIIS, 2006.
[14] R. Peredo Valderrama, I. Peredo Valderrama, L. Balladares Ocaña, "Sistema generador de contenidos multimedia interactivos didácticos usando componentes de software para educación basada en *Web"*, **6ta. Conferencia Iberoamericana en Sistemas, Cibernética e Informática** (CISCI 2007), IIIS, 2007.
[15] C.Szyperski, **Component Software -Beyond Object-Oriented Programming,** Addison-Wesley 1999.
[16] L. Sheremetov & R. Peredo Valderrama, "Development of Reusable Learning Materials for WBE using Intelligent Components and Agents", **International Journal of Computers & Applications**, 25(3): 170-178, ACTA Press, 2003.
[17] A. Canales, A. Peña, R. Peredo, H. Sossa & A. Gutiérrez, "Adaptive and intelligent web based education system: Towards an integral architecture and framework", **Expert Systems with Applications**, 33: 1076–1089, Pergamon Press, 2007.
[18] M. E. Davis, J. A. Phillips, **Flex 3: A Beginner's Guide**, McGraw-Hill Osborne Media 2008.
[19] IEEE Learning Technology Standards Committee. URL: http://ltsc.ieee.org/
[20] IEEE 1484.11.2 Standard for Learning Technology – ECMAScript Application Programming Interface for Content to Runtime Services Communication. November 10, 2003 URL: http://ltsc.ieee.org/
[21] IEEE 1484.12.1-2002 Learning Object Metadata Standard. URL: http://www.ieee.org/
[22] Struts, URL: http://struts.apache.org/
[23] Xerces Java Parser, URL: http://xerces.apache.org/xerces-j/
[24] DOM specification, URL: http://www.w3.org/DOM/
[25] XLST specification, URL: http://www.w3.org/TR/xslt
[26] C. Bauer & G. King, **Hibernate in Action**, Manning Publications 2004.