LEARNING TECHNOLOGY SYSTEM ARCHITECTURE BASED ON AGENTS AND SEMANTIC WEB

Alejandro Canales and Rubén Peredo

Computer Science Research Center of National Polytechnic Institute, Col. Nueva Industrial Vallejo
Del. Gustavo A, Madero, D.F. 07738, Mexico City, Mexico

Keywords: Web-Based Education, Agent-Oriented Programming, IRLCOO.

Abstract: The paper presents a new Learning Technology System Architecture that is implemented using agents and semantic Web. The architecture is divided into client and server parts to facilitate adaptivity in various configurations such as online, offline and mobile scenarios. The implementation of this approach to architecture is discusses in SiDeC (authoring tool) and Evaluation System, which are researching, evaluating and developing an integrated system for Web-Based Education, with powerful adaptivity for the management, authoring, delivery and monitoring of such material.

1 INTRODUCTION

The use of Web-Based Education (WBE) as a mode of study is due to the increase in the number of students and limited learning content resources available to meet a wide range of personal needs, backgrounds, expectations, skills, levels, etc. In this way, the purpose of the delivery process is very important, because it means to produce learning content and to present it to the learner in multimedia form. Nowadays, there are approaches over this process that focus on new paradigms to produce and deliver quality content for online learning experiences, such as a special type of labeled materials called Intelligent Reusable Learning Components Object Oriented (IRLCOO), developed by Peredo et al (2005).

IRLCOO are part of a new Learning Technology System Architecture (LTSA) based on IEEE 1484 specification (IEEE, 2001) and open standards such as XML (XML, 2006) as a bar coding system and to make sure that the learning content is interoperable, the Global IMS Learning Consortium (IMS, 2005), Advanced Distributed Learning (ADL), and SCORM (ADL, 2006). This paper is organized as follows: in Section 2, LTSA and IRLCOO are described; in Section 3 y 4, the authoring system called SiDeC and the Evaluation System are presented; finally, the conclusions are discussed.

2 LEARNING TECHNOLOGY SYSTEM ARCHITECTURE

Our LTSA is based on layer 3 of IEEE 1484 specification. This architecture is presented in figure 1, and consists in four processes manage by agents: Learner Entity, Evaluation, Coach, and Delivery process; two stores: Learner Records and Learning Resources; and fourteen information workflows.

First, the Coach process has been divided in two subprocesses: Coach and Virtual Coach. The reason is because we considered that this process has to adapt to the learners' individual needs in a quick way during the learning process. For this, some decisions over sequence, activities, examples, etc., can be made manually for the coach but in others cases these decisions can be made automatically for the virtual coach.

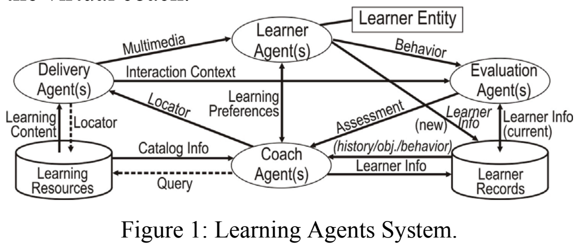


Figure 1: Learning Agents System.

Briefly, the overall operation has the following form: (1) the learning styles, strategies, methods, etc., are negotiated among the learner and other stakeholders and are communicated as learning preferences; (2, new proposal) the learner information (behavior inside the course, e.g., trajectory, times, nomadicity, etc.) is stored in the Learner Records; (3) the learner is observed and evaluated in the context of multimedia interactions; (4) the Evaluation produces assessments and/or learner information; (5) the learner information (keyboard clicks, mouse clicks, voice response, choices, written responses, etc., all over learner's evaluation) is stored in the learner history data-base; (6) the Coach reviews the learner's assessment and learner information, such as preferences, past performance history, and, possibly, future learning objectives; (7, new proposal) the Virtual Coach reviews the learner's behavior and learner information, and automatic and smartly he makes dynamic modifications on the course sequence (personalized to learner's needs) based on the learning process design; (8) the Coach/Virtual Coach searches the learning resources, via query and catalog info, for appropriate learning content; (9) the Coach/Virtual Coach extracts the locators (e.g., URLs) from the available catalog info and passes the locators to the delivery process, e.g., a lesson plan or pointers to content; and (10) the Delivery process extracts the learning content and the learner information from the Learning Resources and the Learner Records respectively, based on locators, and transforms the learning content to an interactive and adaptive multimedia presentation to the learner.

2.1 IRLCOO Platform

IRLCOO were developed with Flash. Flash is an integrator of media and have a powerful programming language denominated ActionScript 3.0 (Adobe, 2007). This language is completely Object Oriented and enables the design of learning components that allows multimedia content of side client. At Run-Time, the components load media objects and offer a programmable and adaptive environment to the student's necessities. Flash already has Smart Clips for the learning elements denominated Learning Interactions. The aim is to generate a multimedia library of IRLCOO for WBE systems with the purpose to separate the content from the control. Thus, the components use different levels of code inside the Flash Player. With this structure, it is possible to generate specialized components which are small, reusable, and suitable to integrate them inside a bigger component at Run-Time by Delivery process. The liberation of ActionScript version 3.0 inside Adobe Flash© allows the implementation of the Object Oriented paradigm. With these facilities IRLCOO are tailored to the learners' needs. In addition, this IRLCOO development platform owns certain communication functionalities inside the Application Programming Interface with LMS, Multi-Agent System (MAS), and different frameworks, as AJAX (Crane, 2006), Hibernate (Peak, 2006), Struts (Holmes, 2004), etc., and dynamic load of assets in Run-Time.

IRLCOO are meta-labeled with the purpose of complete a similar function as the product bar codes, which are used to identify the products and to determine certain characteristics specify of themselves. This contrast is made with the meta-labeled Resource Description Framework (RDF-XML) (RDF, 2004), which allows enabling certain grade inferences on the materials by means of the Semantic Web Platform.

2.2 Communication between IRLCOO and Web Services

The Web Service (WS) standards enable a set of basic interactions required in a Service Oriented Architecture (SOA). WS allow access to functionality via the Web using a set of open standards that make the interaction independent of implementation aspects such as the operating system platform and the programming language used.

ActionScript 3.0 adds the component *WebServiceConnector* to connect to WS from the IRLCOO. The *WebServiceConnector* component enables the access to remote methods offered by a LMS through SOAP protocol. This gives to a WS the ability to accept parameters and return a result to the script, in other words, it is possible to access and join data between public or own WS and the IRLCOO. It is possible to reduce the programming time, since a simple instance of the *WebServiceConnector* component is used to make multiple calls to the same functionality within the LMS. The components discover and invoke WS using SOAP and UDDI, via middleware and a JUDDI server (JUDDI, 2005). Placing a Run-Time layer between a WS client and server dramatically increases the options for writing smarter, more dynamic clients. Reducing the needs for hard-coded dependencies within WS clients (see figure 2). It is only necessary to use different instances for each one of the different functionalities. WS can be unloaded using the component and deployed within an IRLCOO.

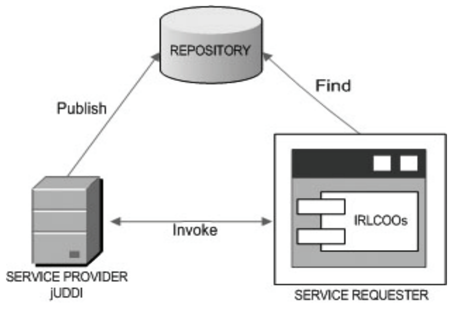


Figure 2: Web service architecture with IRLCOO.

2.3 Mapping Object/Relational

Hibernate is a powerful, high performance object/relational persistence and query service. Hibernate lets develop persistent classes following object-oriented idiom - including association, inheritance, polymorphism, composition, and collections (Hibernate, 2005).

According with figure 1, the "Learner Info (new)" flow is implemented with Hibernate. Creating the InfrastructureException class for Hibernate Errors and Exceptions, this class is used to manipulate exceptions in the Hibernate code. Later creating the HibernateUtil class is possible to initiate Hibernate and get access to a session of the persistence and transaction operations.

The StudentDAO class encapsulates all functionalities necessary to persist student's registers data in the database using Hibernate. Thus, the action class will be responsible by get information from student, pass to persist for the DAO class and return success or error to the user. Mixing Struts and Hibernate and creating the StudentAction class, this class is an adapter between the contents of an incoming HTTP request and the corresponding business logic that should be executed by this process request. The controller (RequestProcessor) will select an appropriate Action for each request, create an instance, and call the execute method. The database configuration in Hibernate is made trough of the hibernate.cfg.xml file. The mapping between Student Class and database is carried out in the Student.hbm.xml file.

3 SiDeC

In order to facilitate the development of learning content, it was built an authoring system called SiDeC (Sistema de Desarrollo de eCursos - eCourses Development System). SiDeC is a system based on LTSA to facilitate the authoring content to the tutors who are not willing for handling multimedia applications. In addition, the Structure and Package of content multimedia is achieved by the use of IRLCOO, as the lowest level of content granularity.

SiDeC is used to construct Web-based courseware from the stored IRLCOO (Learning Resources), besides enhancing the courseware with various authoring tools. Developers choose one of the SiDeC lesson templates and specify the desired components to be used in each item. At this moment, the SiDeC lesson templates are based on the cognitive theory of Conceptual Maps (CM), but in the future we will consider others theories such as: Based-Problems Learning (BPL), the cases method, etc.

SiDeC has a metadata tool that supports the generation of IRLCOO to provide online courses. This courseware estimates learners' metrics with the purpose to tailor their learning experiences. Furthermore, the IRLCOO offer a friendly interface and flexible functionality. These deliverables are compliance with the specifications of the IRLCOO and with learning items of SCORM 1.2 Models (Content Aggregation, Sequencing and Navigation, and Run Time Environment) (ADL, 2006). Metadata represent the specific description of the component and its contents, such as: title, description, keywords, learning objectives, item type, and rights of use. The metadata tool provides templates for entering metadata and storing each component in the SiDeC or another IMS/IEEE standard repository.

In figure 3, the SiDeC implements the CM as a navigation map or instructional and learning strategy allowing to the learner to interact with content objects along the learning experiences. These experiences follow an instructional-teaching strategy. These kinds of strategies carry out modifications of the learning content structure. Such modifications are done by the designer of the learning experience with the objective of provide significant learning and to teach the learners how to think (Díaz-Barriga, 2002). The learning content can be interpreted in a Learning Content Tree.

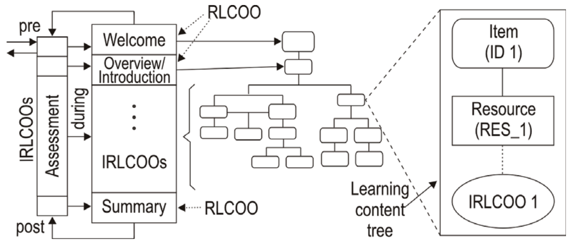
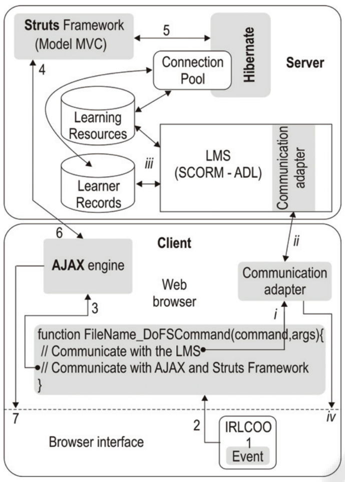


Figure 3: Learning content generated for the SiDeC.

3.1 Communication between IRLCOO and LMS

Our communication model uses an asynchronous mode in Run-Time Environment (RTE) and joins to LMS communication API of ADL (ADL, 2006), AJAX (Asynchronous JavaScript And XML) (Crane, 2006) and Struts Framework (Holmes, 2004) for its implementation. The LMS communication API of ADL consists of a collection of standard methods to let the Client to communicate with the LMS. The browser-based communication model is depicted in figure 4.



```
function FileName_DoFSCommand(command,args){
 // Communicate with the LMS
// Communicate with AJAX and Struts Framework
}
```

Figure 4: Communication model between IRLCOO, LMS and AJAX and Struts Framework.

According to figure 4, the communication model starts: (1) when an IRLCOO generates an event. (2) Form the browser interface is made a JavaScript call to the function FileName_DoFSCommand (command,args), which handles all the FSCommand messages from IRLCOO, LMS communication API, and AJAX and Struts methods.

The communication with the LMS starts when: (i) the standard methods call to the Communication Adapter (written in JavaScript). (ii) The communication adapter implements the bidirectional communication ADL´s API between the Client and the LMS. (iii) The LMS realizes the query-response handling and the business logic. The purpose of establishing a common data model is to make sure that a defined set of information about content can be tracked by different LMS environments. If, for example, it is determined that tracking a student's score is a general requirement, then it is necessary to establish a common way for content to report scores and for LMS environments to process such information (for interoperability and reuse of such content). If every chunk of content used its own unique scoring representation learning management systems would not know how to receive, store or process such information. These definitions are derived from the IEEE Standard for Computer-Managed Instruction document (P1484.11), which originated within the Aviation Industry CBT Committee (AICC) (IEEE, 2005).

The communication with AJAX and Struts Framework begins when AJAX-Struts method is called. (3) An instance of the XMLHttpRequest object is created. Using the open() method, the call is set up, the URL is set along with the desired HTTP method, typically GET or POST. The request is actually triggered via a call to the send() method.

(4) A request is made to the server, this might be a call to a servlet or any server side technique. (5) The Controller is a servlet which coordinates all applications activities, such as: reception of user data, data validations, and control flow. The Controller is configured for a XML file. The Controller calls to perform method of Action, it passes to this method the data values and the Action reviews the characteristic data that correspond to the Model. The business objects (JavaBeans) realize the business logic, (5) usually a database access by Hibernate. The Action sends the response to the Controller. The Controller reroutes and generates the interface for the results to the View (JSPs). The View makes the query to the Business objects based on the correspondent interface. (6) The request is returned to the browser. The ContentType is set to text/xml, the XMLHttpRequest object can process results only of the text/ html type. In more complex instances, the response might be quite involved and include JavaScript, DOM manipulation, or other technologies. The XMLHttpRequest object calls the function callback() when the processing returns. This function checks the readyState property on the XMLHttpRequest object and then looks at the status code returned from the server. (7) Provided everything is as expected, the callback() function sends HTML code and it does something interesting on the client, i.e. advanced dynamic sequence.

This communication model provides new wide perspectives for the WBE systems development, because it provides the capabilities of communication, interaction, interoperability, security, and reusability, between different technologies. For example, the LMS communication API allows us to make standard database queries of learners' information such as personal information, scores, assigned courses, trajectory, etc. While the communication with AJAX and Struts Framework provides the capability of modify the learner's trajectory according to variables from the learner records in RTE (advanced dynamic sequence), components management (IRLCOO) – remember that these components are built and programming with XML – then, this model provides the way to write, load, change and erase XML files in the Server side.

4 EVALUATION SYSTEM

The Evaluation System for WBE is designed under the same philosophy used for the SiDeC. The functionality of the Evaluation System lays on the analysis of the learner's profile, which is built during the teaching-learning experiences. The profile is based on metrics that elicited from the learner's behavior at Run-Time. These measures are stored into the learner records that compose the profile. The generation of new sequences of courses is in function of the results obtained, besides the account of the adaptation level.

The Evaluation System combines IRLCOOs, additional meta-labels, and a Java Agent platform. Also, some technologies of the Artificial Intelligence field are considered in order to recreate a Semantic Web environment. Semantic Web aims for assisting human users to achieve their online activities.

In resume, the components and operation of the SiDeC and Evaluation System are outlined in figure 5. Basically the Evaluation System is fulfilled through two phases. The first phase is supported by the LMS, and is devoted to present the course and its structure. All the actions are registered and the presentation of the contents is realized with IRLCOO content. The evaluations are done by evaluating IRLCOO and in some cases by simulators based on IRLCOO. These processes are deployed by the Framework of Servlets, JSPs and JavaBeans.

The second phase analyzes the learner's records carried out by the Server based on JADE MAS. This agent platform owns seven agents: Snooper, Buffer, Learner, Evaluation, Delivering, Coach, and Info. The fundamental idea is to automate the learner's analysis through the Coach/Virtual Coach, and to give partial results that can be useful for the learner's final instruction. These agents are implemented as Java-Beans programs, which are embedded in the applications running both at the client and server sides. The Snooper Agent works as a trigger by means of the INFORM performative, which activates the MAS server's part. This agent is deployed into a Java Server Page that uses a JavaBean. During the lesson or once evaluation is finished, the graphical user interface activates the Snooper Agent and sends it the behavior or evaluation metrics (using Agents Communications Language or FIPA ACL (FIPA, 2001) to be analyzed at the server-side of the MAS. The Snooper Agent activates the system, whereas the Buffer Agent manages the connection and all the messages from the client. Both tasks are buffered and send them to the Coach Agent. Then the Coach Agent requests to the learner records for the preferences learner, trajectory, previous learner monitoring information, etc. The Coach Agents analyzes this information to determine if the learner needs help. If this situation is true, the Coach Agent requests to the learning resources the needful learning content (URLs) and it sends the learning contents (URLs) to the Delivery Agent. The Delivery Agent sends the learning content to the Learner and Evaluation Agents for its presentation. These agents employ the dynamic sequencing to change the course or assessment sequence. The sequencing is defined for the instructional strategy based on CM and it employs the SCORM Sequencing/Navigation. Once the necessary information is received (sequence, kind of IRLCOO and localization, etc.), this is represented as a string, which is constructed dynamically by the rule-based inference engine known as JENA (JENA, 2006) and JOSEKI server (JOSEKI, 2006), to generate dynamic feedback.

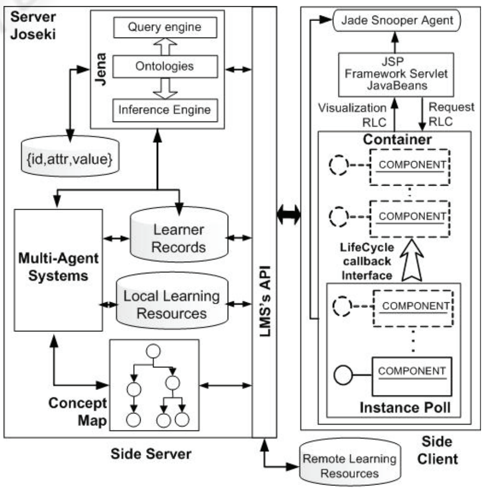


Figure 5: Semantic Web Platform for WBE.

4.1 Semantic Web Platform

The overall architecture of Semantic Web Platform, which includes three basic aspects (see figure 5):

1. The query engine receives queries and answers them by checking the content of the databases that were filled by info agent and inference engine.
2. The database manager is the backbone of the entire systems. It receives facts from the info agent, exchanges facts as input and output with the inferen-ce engine, and provides facts to the query engine.
3. The inference engine use facts and ontologies to derive additional factual knowledge that is only provided implicated. It frees knowledge providers from the burden of specifying each fact explicitly.

Again, ontologies are the overall structuring principle. The info agent uses them to extracts facts, the inference engine to infer facts, the database manager to structure the database, and query engine to provide help in formulating queries.

JENA was selected as the inference engine. It is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS, OWL, SPARQL and includes a rule-based inference engine (JENA, 2006).

While JOSEKI was selected as Web API and server. It is an HTTP and SOAP engine supports the SPARQL Protocol and the SPARQL RDF Query language. SPARQL is developed by the W3C RDF Data Access Working Group (JOSEKI, 2006).

5 CONCLUSIONS

LTSA, IRLCOO and Semantic Web Platform allow developing authoring and evaluation systems to create adaptive and intelligent WBE. Our approach focus on: reusability, accessibility, durability, and, interoperability of the learning contents.

The communication model composes for the LMS communication API, AJAX and Struts Framework, IRLCOO, WS, Semantic Web, and JUDDI. It provides new development capabilities for WBE systems, because their integrant technologies are complementary. SiDeC and the Evaluation System were developed under this model to help in the automation and reduce of the complexity of the learning content process.

The incorporation of Web Semantic Platforms helps us to create intelligent and adaptive systems (bidirectional communication), according to the users' needs.

ACKNOWLEDGEMENTS

Authors of this paper would like to thank the Instituto Politécnico Nacional (IPN), Centro de Investigación en Computación (CIC) for the partial support for this project within the project IPN 20071166.

REFERENCES

ADL (2006). Advanced Distributed Learning Consortium. Retrieved January 24, 2006 from http://www.adlnet.org

Adobe (2007). Adobe© Flash©. Retrieved February 26, 2007 from http://www.adobe.com

Crane. D., Pascarello, E., James, D., 2006. Ajax in Action, Manning Publications, 1st edition.

Díaz-Barriga, F., 2002. Educational strategies for a significant learning, Mc Graw Hill Publication, Second edition.

FIPA (2001). FIPA ACL message structure specification, XC00061. Retrieved August 20, 2005 from http://www.fipa.org/

Hibernate (2005). Hibernate. Retrieved October 11, 2006 from http://www.hibernate.org/

Holmes, J., 2004. Struts: The Complete Reference, Mc Graw Hill – Osborne Publications, 1st edition.

IEEE (2001). IEEE 1484.1/D9 LTSA: Draft Standard for Learning Technology - Learning Technology Systems Architecture. Retrieved December 8, 2003 from http://ieee.ltsc.org/wg1

IEEE (2005). IEEE Learning Technology Standards Committee. Retrieved December 10, 2005 from http://ltsc.ieee.org/wg11/index.html

IMS (2005). Global IMS Consortium. Retrieved March 22, 2005 from http://www.imsproject.org

JENA (2006). Jena – A Semantic Web Framework for Java. Retrieved June 9, 2006 from: http://jena.sourceforge.net/

JOSEKI (2006). Joseki - A SPARQL Server for Jena. Retrieved June 9, 2006 from: http://www.joseki.org/

JUDDI (2005). Apache Web Service Project. Retrieved November 4, 2006 from http://ws.apache.org/juddi/

Passin, T., 2004. Explorer's Guide to Semantic Web. Manning Publications Co., 1st edition.

Peak, P., Heudecker, N., 2006. Hibernate Quickly, Manning Publications, 1st edition.

Peredo, R., Balladares, L., Sheremetov, L., 2005. *Development of intelligent reusable learning objects for web-based education systems.* Expert Systems with Applications. 28(2). 273-283.

RDF (2004), Resource Description Framework. Retrieved January 15, 2005 from http://www.w3.org/RDF/

XML (2006). eXtensible Markup Language specifications. Retrieved January 7, 2007 from http://www.w3.org/XML.