Multi-agent system for the making of intelligence and interactive decisions wit hin the learner’s learning process in a web-based education environment

A. Canales-Cruz[1]*, V. G. Sánchez-Arias[1], F. Cervantes-Pérez[2], R. Peredo-Valderrama[3]

[1]Centro de Alta Tecnología de Educación a Distancia
(cated), CUAED, Universidad Nacional Autónoma de México
*alejandro_canales@cuaed.unam.mx
victor_sanchez@cuaed.unam.mx
[2] Coordinación de Universidad Abierta y Educación a Distancia (CUAED),
Universidad Nacional Autónoma de México.
francisco_cervantes@cuaed.unam.mx
[3] Centro de Investigación en Computación del Instituto Politécnico Nacional.

ABSTRACT
The main focus of this paper is to show the concepts, architectures, interaction techniques, and general approaches to the analysis and specification of a multi-agent system for the making of intelligence and interactive decisions inside the learner’s learning process. These contributions are applied in the development of an integrated system for Web-based education with powerful adaptivity for the management, authoring, delivery and monitoring of learning content.

Keywords: Web-Based Education, multi-agent system, Semantic Web.

RESUMEN
El enfoque principal de este artículo es mostrar los conceptos, arquitecturas, técnicas de interacción y enfoques generales para el análisis y especificación de un sistema multi-agente para toma de decisiones inteligentes e interactivas dentro del proceso de aprendizaje del estudiante. Estas contribuciones se aplican en el desarrollo de un sistema integrado para la educación basada en Web, con gran adaptabilidad para la administración, autoría, entrega y monitoreo del contenido de aprendizaje.

Palabras clave: Educación basda en Web, sistema multiagente, Web semántica.

1. Introduction

Nowadays, research about the development of Web-based education (WBE) systems revolves around the consideration of technological and pedagogic requirements and their functionalities necessary for a Web environment. Several approaches were used to develop decision-making assistance tools for WBE systems. But the intelligent software agents are a unique generation of information society tools that independently perform various tasks on behalf of the human user(s) or other software agents. The WBE requires the development of new, more intelligent methods, tools, and theories for the modelling and engineering of agent-based systems and technologies [1]. This directly involves a need for consideration, understanding, and analysis of human factors, e.g., the learner’s knowledge and skill, learning, performance capabilities, and compatibilities. This is because software developers utilize a student model based on the conceptualization and maintenance of a mental model that the system achieves in relation to the learner along the teaching–learning experiences [2]. Therefore, the study of interrelated factors, such as people’s and agents’ capabilities, constitutes an important and critical subject in the intelligent social agents field. This should eventually lead to more robust, intelligent, interactive, learning, and adaptive agents.

To respond to these needs, an intelligent and interactive multi-agent aystem (MAS) is presented. This MAS is applied for the development of WBE systems and it considers the diversity of requirements and provides the needed functionalities based on the facilities of the Web.

The purpose of this paper is to show a new architecture for the development of WBE systems based on the IEEE LTSA (Learning Technology System Architecture) specification [3]. In order to achieve this goal, this paper is organized as follows: In section 2, the Agent Architecture based on IEEE LTSA is presented; whereas in section 3, the methodology for the development of learning objects is shown. These objects are a special type of Sharable Content Object (SCO) according to the Sharable Content Object Reference Model (SCORM). SCORM is used to create reusable and interoperable learning content [5]. The learning materials are labeled with the Resource Description Framework (RDF) [6] and the eXtensible Markup Language (XML) [7] to be used for the rule-based inference engine known as JENA [8], and the JOSEKI server to implement a semantic platform [9].

Afterwards, in Section 4, an intelligent and interactive MAS is shown. MAS contains all the logic to make the decisions in order to deliver intelligent and collaborative teaching–learning experiences suitable for the particular requirements of each individual. This MAS is a set of standard specifications supporting inter-agent communication and key middleware services. All the communications are done according to the entire FIPA2000 standards [10] and they are represented through an agent-specific extension of Unified Modeling Language (UML) known as AUML or agent.

In section 5, the authoring tool called CCObÁ is depicted [4]. Besides, this system is based on the learning objects methodology (reviewed in Section 3), Semantic Web and MAS. Other used technologies for the implementation are AJAX (Asynchronous JavaScript And XML) [11], which is used for communication between components and LMS's API (Application Programming Interface), JADE (Java Agent DEvelopment Framework) [12], Struts implements the design pattern MVC (Model-View-Controller) [13], Servlets, JSPs and JavaBeans implements other functionalities of the system under the model MVC, DOM (Document Object Model) and XSLT (Extensible Stylesheet Language Transformations) for XML persistence [7], Hibernate for object/relational persistence and query service [14].

Finally, in Section 6, we conclude with some comments and evaluation on this work.

2. Agent architecture

WBE systems are too large, complex, dynamic, open, and decision making is managed centrally or via predefined techniques. We propose that the only feasible alternative is for computational intelligence to be embedded at many and sundry places in such environments to provide distributed control [15]. Each locus of embedded intelligence is best thought of as an autonomous agent that finds, conveys, or manages information. Because of the nature of the environments, the agents must be long-lived (they should be able to execute unattended for long periods of time), adaptive (they should be able to explore and learn about their environment, including each other), and social (they should interact and coordinate to achieve their own goals, and the goals of their society; they should rely on other agents to know things so they do not have to know everything).

Social agency involves abstractions from sociology and organizational theory to model societies of agents. Since agents are often best studied as members of MAS, this view of agency is important and gaining recognition. Sociability is essential to cooperation, which itself is essential for moving beyond the somewhat rigid client-server paradigm of today to a true peer-to-peer distributed and flexible paradigm that modern applications call for, and where agent technology finds its greatest payoffs.

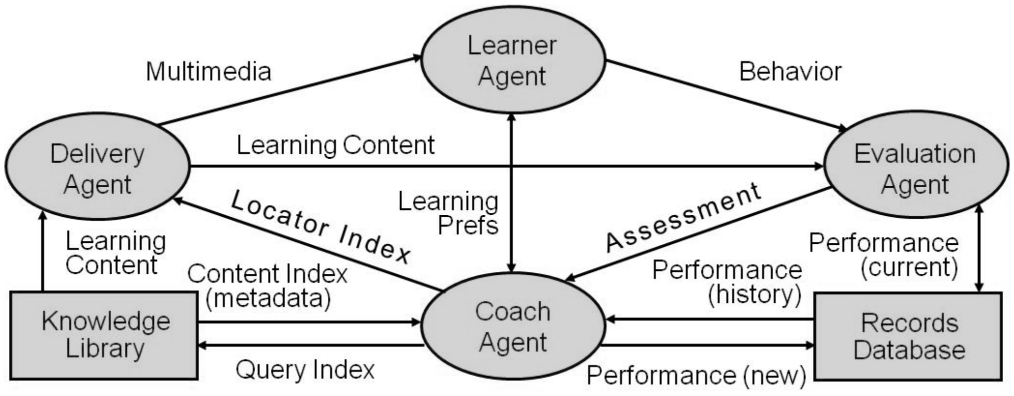


Figure 1. Agent Architecture.

Our agent architecture is based on layer 3 of the IEEE 1484 LTSA specification. This architecture provides a framework to understand and apply the reusability, interoperability and portability of the Learning Content Management System (LCMS) [3]. Lyer 3 is depicted in Figure 1 and consists of four processes managed by agents: learner entity, evaluation, coach, and delivery; two stores: learner records (Records Database) and learning resources (Knowledge Library); and twelve information work-flows.

Layer 3 of the IEEE 1484 LTSA specification was modified to support processes based on agents to adapt to the learners' needs in an intelligent form. For example, the coaching process has been divided in two sub-processes: 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 by the coach but in other cases these decisions can be made automatically by the virtual coach agent.

2.1 Adaptation in wbe systems

The term "adaptation" is an important aspect in WBE systems. The application of adaptation can provide better learning environments in such systems. There are two forms of WBE systems development for supporting the learners' individual needs [15]:

• Those that allow the learner to change certain system parameters and adapt their behavior accordingly are called *adaptable*.

• Those that adapt to the learners' needs in an intelligent form and are automatically based on the system's conjecture are called *adaptive*.

This work is centered in the development of intelligent, adaptive and interactive WBE systems. The constant change of WBE systems requires taking into account the following aspects:

• The adaptation with respect to current domain competence level of the learner.

• The suitability with respect to domain content.

• The adaptation with respect to the context in which the information is being presented.

To cover these aspects, we propose to develop a new architecture which captures the interactions of learners with the system to extract information about their competence level for various domain concepts and tasks represented in the system.

According to the potential and dynamism in the WBE systems, we are developing a suitable way to capture interactions over the Internet and to provide a continuous interaction pattern for a given learner (learner agent, see Figure 1). In WBE systems, the interactions between client and server normally take place using the Hypertext Transfer Protocol (HTTP). HTTP is a stateless protocol, which makes it difficult to track the learner's progress and hence to analyze the learner's mental processes. However, by using the technologies of LMS API, Web Services, Ajax [11], Hibernate [14], Struts Framework [13] and Semantic Web Platform (depicted in section 5.1) together, we can more accurately record the learner's behavior in the browser. Part of the new architecture resides on the server and part on the user's machine.

2.2. Learner entity

The learner entity allows adaptive behavior of the system by providing information about the learner agent. The learner agent processes granular information about the learner's competence level for various tasks represented in the system (depicted in Figure 1). The learner agent is used by the system to

• support adaptive navigation guidance - based on prioritized successors and the learner's needs,
• support context based on previous learning components,
• support dynamic messaging and feedback, e.g. navigation, learning content, current context and progression.

Figure 2 shows the summary of the learner agent which provides adaptation to the learners in an intelligent form and is automatically based on the virtual coach agent.

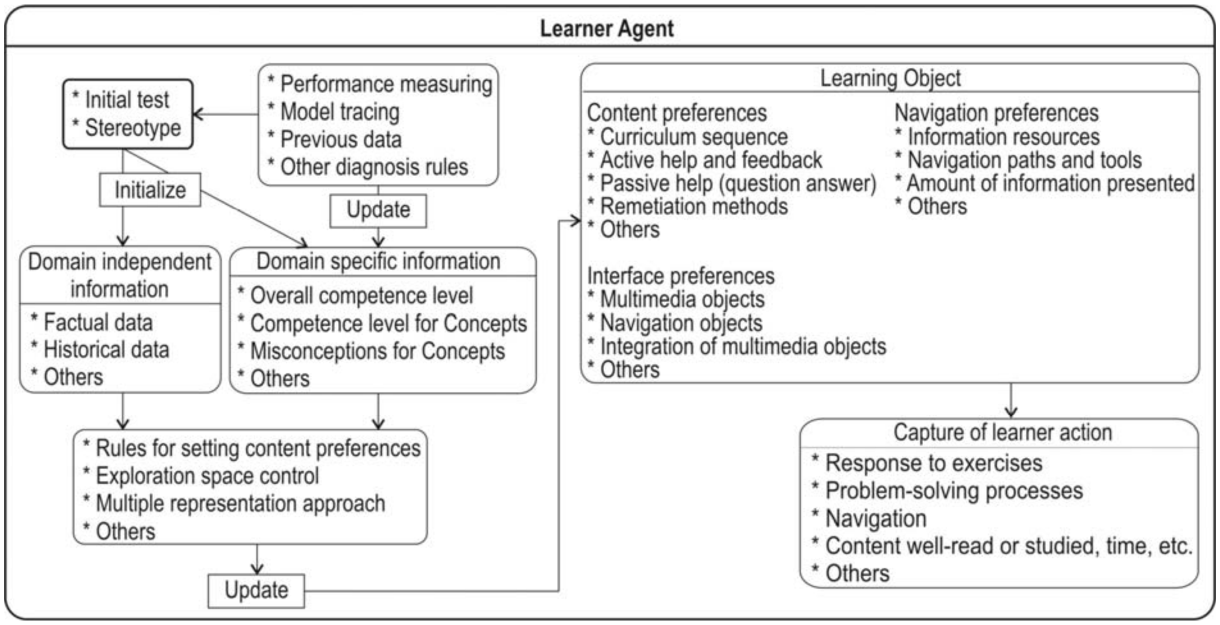

Learner Agent
* Initial test
* Stereotype
* Performance measuring
* Model tracing
* Previous data
* Other diagnosis rules
Initialize
Update
Domain independent information
* Factual data
* Historical data
* Others
Domain specific information
* Overall competence level
* Competence level for Concepts
* Misconceptions for Concepts
* Others
* Rules for setting content preferences
* Exploration space control
* Multiple representation approach
* Others
Update
Learning Object
Content preferences
* Curriculum sequence
* Active help and feedback
* Passive help (question answer)
* Remetiation methods
* Others
Navigation preferences
* Information resources
* Navigation paths and tools
* Amount of information presented
* Others
Interface preferences
* Multimedia objects
* Navigation objects
* Integration of multimedia objects
* Others
Capture of learner action
* Response to exercises
* Problem-solving processes
* Navigation
* Content well-read or studied, time, etc.
* Others

Figure 2. Learner Agent – top view.

3. Methodology for the development of learning objects

The methodology is centered in the learner's learning and combines a multidisciplinary group for constructing the learning objects (LOs). The leader of this group is the tutor (called content expert) and she/he interacts with the instructional designer, graphics designer, and Web programmer. These last ones are responsible for (1) analyzing, designing, and evaluating the didactic units; and (2) designing, developing and evaluating the LOs. Figure 3 depicts this methodology.

The methodology starts when the content expert provides information such as the topic, educative purposes, general contents, etc. LOs are developed in two phases: pedagogical and technological. This work is made by the instructional designer,graphics designer, and Web programmer. The new LOs are stored in the database (called Knowledge Library according Figure 1). Finally, LOs are managed by the LCMS and shown to the learners.

LOs were developed with Java and are used to develop the authoring tool called CCOBÁ described in Section 5.

At run time, the components load media objects and offer a programmable and adaptive environment to the learner's needs through the composite pattern. The composite pattern provides a robust solution to build complex systems that are made up of several smaller components. The system is made by components that may be individual media objects or containers that represent collections of media objects. The composite pattern streamlines the building and manipulation of complex media objects that are composed of several related pieces. The complex media objects are built as hierarchical trees, the structure components can be individual components (primitives or indivisible objects) or composite components that hold a collection of other components, and they allow the clients to treat both individual components and composite components the same way, simplifying the interface [16]. This pattern has particular utility in Java, allowing easily building and manipulating complex media objects.

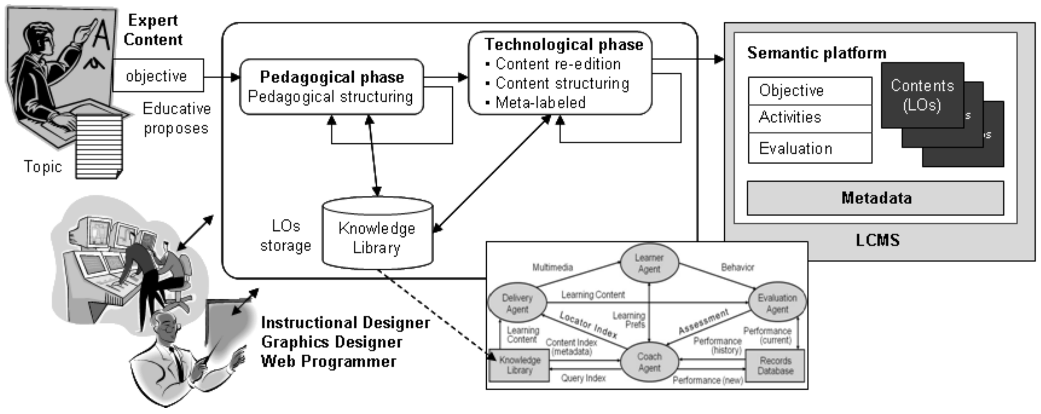


Figure 3. Methodology for the development of learning objects.

The purpose is to generate a multimedia library of LOs for WBE systems with the purpose to separate the content from the control. 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 the delivery process. In addition, the LOs platform owns certain communication functionalities inside the Application Programming Interface (API) developed for our MAS, dynamic load of assets, and assets composition. The middleware for the components uses different frameworks as Ajax [11], Web Services [17], Hibernate [14], Struts [13], etc.

LOs are meta-labeled with RDF-XML [6, 7], which allows enabling certain grade inferences on the materials by means of the Semantic Web Platform.

4. Intelligent and interactive multi-agent system

The construction of our intelligent and interactive MAS is carried out with Jadex. Jadex is a software framework for the creation of goal-oriented agents following the belief-desire-intention (BDI) reasoning engine that allows to program intelligent software agents in XML and Java [18]. The reasoning engine is very flexible and can be used on top of different middleware infrastructures such as JADE. Jadex aims to build up a rational agent layer that sits on top of a middleware agent infrastructure and allows for intelligent agent construction using software engineering foundations. Moreover, Jadex allows developing MAS addressed by the Foundation for Intelligent Physical Agents (FIPA) specifications.

FIPA has defined a standard agent communication language, namely FIPA-ACL [10]. This language has the advantage that it relies not only on a syntactic definition, but also on a semantic form. In other words, the FIPA-ACL standard formally specifies a precise meaning for each primitive communication in the language.
According with Figure 1, our main MAS is conformed by learner, evaluation, coach/virtual coach, and delivery agents. In this case, it is focused on the learner agent. In Figure 4, an overview of the abstract Jadex architecture is presented. Viewed from the outside, the learner agent is a black box capable of sending and receiving messages. The main activity of the learner agent consists of interpreting the received FIPA-ACL messages. This interpretation activity can be refined into two main functions: the first one produces some sense about the input message, while the second one consumes this sense and updates the agent's respective activities and beliefs.

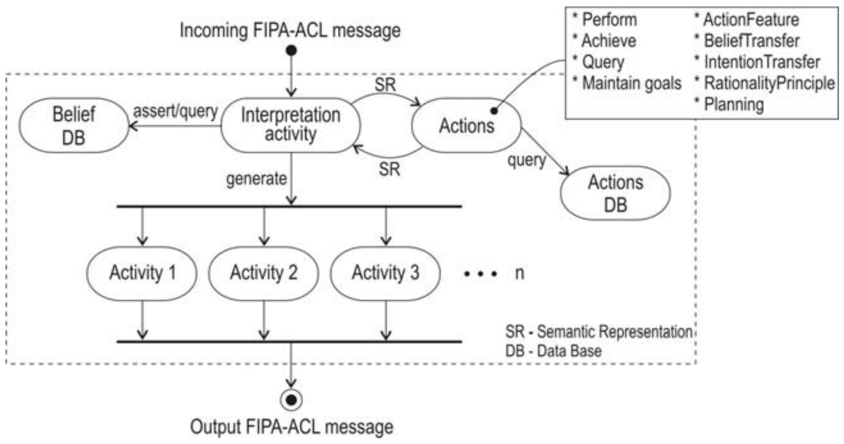


Figure 4. Jadex abstract architecture for the learner agent.

4.1. Beliefs

Beliefs represent the state of the agent's knowledge about the world, itself and other agents [18]. For example, performance measuring, model tracing, previous data, etc. (see Figure 2). The belief representation in Jadex allows arbitrary Java objects to be stored instead of relying on a logic-based representation. This facilitates integration with our Web-based Applications (CCOBÁ), e.g. classes generated by ontology modelling tools and our database mapping layers can be directly re-used. Objects are stored as named facts (called beliefs) or named sets of facts (called belief sets). Using the belief names, the 'belief data base' can be manipulated by setting, adding or removing facts. In addition, a more declarative way of accessing beliefs and belief sets is provided by the Object Query Language (OQL) - like queries.

4.2. Goals

Goals are the motivational force driving an agent's activities. They come in different flavours allowing various attitudes of an agent to be expressed. Activities consist of performing some arbitrarily complex courses of action [18]. These actions include the communicative actions defined by FIPA-ACL, such as inform or request, as well domain-specific actions.

One of the core concepts used by the interpretation activity is the Semantic Representation (SR), which represents a produced or consumed one. Jadex currently supports four application-relevant goal types: perform, achieve, query and maintain. Next, an example of activities based on Figure 4 (learner agent) is described.

First, the "ActionFeature" produces several semantic representations (SRs) from a received message that represent the semantic features of the corresponding communicative action. The FIPA-ACL standard defines two semantic features for each communicative action: the feasibility precondition and the rational effect. The former mainly gives rise to a SR starting the precondition was necessarily true before the communicative action was performed. The latter gives rise to an SR (usually called "intentional effect") stating the sending agent intends the rational effect to become true. In our case, the model tracing represents the intention of a definition action, and the domain specific information denotes the corresponding rational effect. These SRs feed the other activities, leading to the agent's reaction.

The "BeliefTransfer" manages the adoption of beliefs suggested by other agents. It applies to any SR stating an external agent intends the interpreting agent to believe a fact, and produces a new SR stating the interpreting agent actually believes this fact. For our example, the Model tracing results from applying the "BeliefTransfer" to the SR Navigation priority.

Similarly, the "IntentionTransfer" manages the adoption of other agents' intentions. It applies to any SR stating an external agent has an intention, and produces a new SR stating the interpreting agent actually has the same intention. This application may be customized to specify the expected cooperative attitude for the interpreting agent in terms of the intentions to adopt and the external agents to cooperate with. In our case, the SR of domain specific information results from applying the "IntentionTransfer" to model tracing, initial test and stereotype.

Finally, the last three activities manage the planning capabilities of the interpreting agent by creating proper activities in order to satisfy her/his intentions.

4.3. Plans

Means–end reasoning is performed with the objective of determining suitable plans for pursuing goals or handling other kinds of events such as messages or belief changes.

The “Perform” tries to directly perform an intended action (for example, content preferences). The “RationalityPrinciple” searches the agent’s base of actions for an action whose rational effect matches a given intention (for example, the delivery agent is necessary to thecontent preferences). Lastly, the “Planning” uses an external planner to find an (arbitrarily complex) action plan whose performance brings about the input intention.

4.4. Capabilities

A capability results from the packaging of specific functionality into a module with precisely defined interfaces [18]. An agent can be composed of an arbitrary number of capabilities that themselves may include any number of subcapabilities. Jadex contains several generic plans and predefined capabilities in the package jadex.planlib. Basic platform features can be accessed by using the Agent Management System (AMS) and Directory Facilitator (DF) capabilities. The AMS capability offers goals for agent management such as creating new agents or destroying existing ones, whereas the DF capability can be used for accessing yellow pages services such as registering agents or searching specific services via goals.

4.5. Implementing mas

Implementing our MAS with Jadex is simple. It actually consists of implementing the cooperative and domain-specific agent features rather than analyzing in detail and coding all possible messages and interaction protocols. MAS programming is carried out following three main tasks.

The first task consists of implementing the domain-specific actions - these are part of the MAS. In Figure 1, the only four domain specifications that have been implemented are “put on” and “take off”.

The second task consists in coding the agent’s belief base management to handle domain-specific facts.

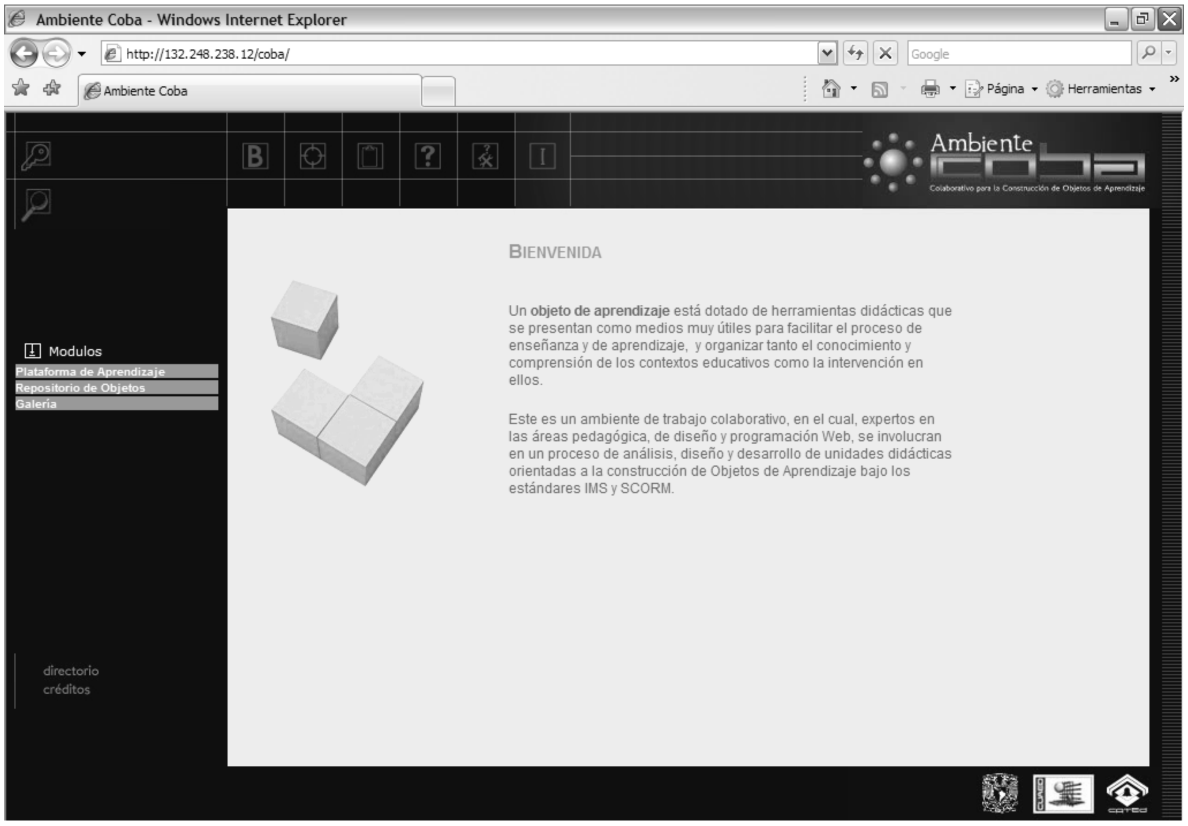
Finally, the last task consists of customizing the cooperation principles of the agent, including the “BeliefTransfer” and “IntentionTransfer” activities. In most cases, a significant part of the code developed to implement an agent is domain-specific. Hence, it can be reused by other agents in this domain without any additional development.

5. Ccobá

To facilitate the development of LOs, an authoring system called CCOBÁ was built (Construcción de Objetos de Aprendizaje – Construction of Learning Objects) [4], see Figure 5. CCOBÁ is a system based on our Agent Architecture to facilitate the authoring content to the tutors who are not willing to handle multimedia applications. In addition, the structure and package of content multimedia is achieved by the use of SCORM, as the lowest level of content granularity.

CCOBÁ is used to construct Web-based courseware from the stored LOs (Knowledge Library, see Figure 1), besides enhancing the courseware with various authoring tools. Developers choose one of the CCOBÁ lesson templates and specify the desired components to be used in each item. At this moment, the CCOBÁ lesson templates are based on the learning strategies of Based-Problems Learning, the cases method, and Based-Project Learning.

The inclusion of these learning strategies obeys to the instructional design pattern for the development of the courses. Thus, the courses do not only have theoretical or practical questions, but rather they include a mental model about individual thought process. These learning strategies have the purpose of helping the learners to maximize the knowledge acquisition.


Ambiente Coba - Windows Internet Explorer
http://132.248.238.12/coba/
Google
Ambiente Coba
Página
Herramientas
Ambiente coba
Colaborativo para la Construcción de Objetos de Aprendizaje
Modulos
Plataforma de Aprendizaje
Repositorio de Objetos
Galeria
directorio
créditos

BIENVENIDA

Un **objeto de aprendizaje** está dotado de herramientas didácticas que se presentan como medios muy útiles para facilitar el proceso de enseñanza y de aprendizaje, y organizar tanto el conocimiento y comprensión de los contextos educativos como la intervención en ellos.

Este es un ambiente de trabajo colaborativo, en el cual, expertos en las áreas pedagógica, de diseño y programación Web, se involucran en un proceso de análisis, diseño y desarrollo de unidades didácticas orientadas a la construcción de Objetos de Aprendizaje bajo los estándares IMS y SCORM.

Figure 5. CCOBÁ homepage.

CCOBÁ has a metadata tool for supporting the generation of XML files for LOs to provide on-line courses. This courseware captures the learners' metrics with the purpose to tailor their learning experiences (through the learner agent). Furthermore, the LOs offer a friendly interface and flexible functionality. These deliverables are in compliance with the specifications of SCORM 1.2 Models (Content Aggregation, Sequencing and Navigation, and Run Time Environment) [5]. 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 CCOBÁ or another IMS/IEEE standard repository.

5.1. Communication model

Our communication model between LOs, MAS and LCMS is shown in Figure 6. This model uses an asynchronous mode in Run-Time Environment (RTE) and LCMS communication API of ADL [5], Ajax [11], Struts Framework [13] for its implementation.

This communication model provides new wide perspectives for the WBE systems development, because it provides the capabilities (i.e.: communication, interaction, communication API allows us to make standard database queries of learners' interoperability, security and reusability) of the different technologies. For example, the LMS information such as personal information,

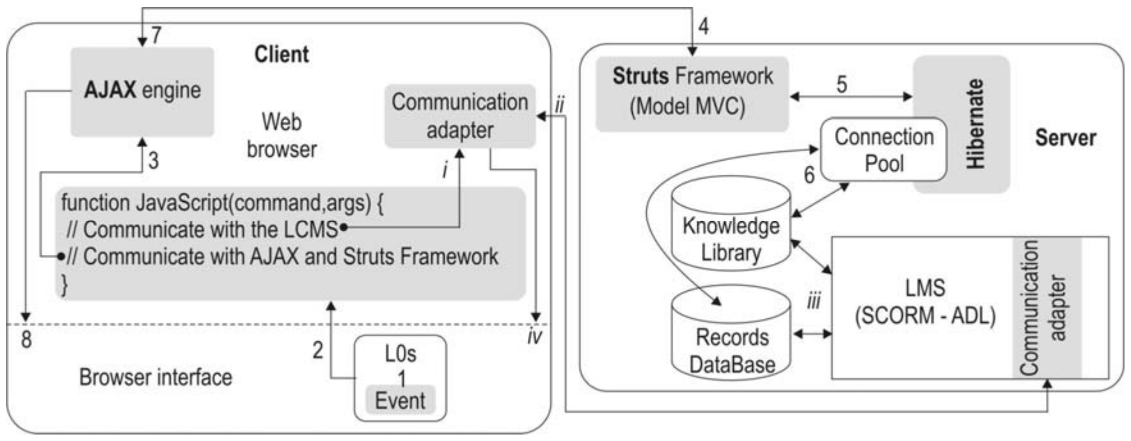


Figure 6. Communication model.

scores, assigned courses, trajectory, etc. While the communication with Ajax and Struts Framework provides the capability of modifying the learner's trajectory according to variables from the learner's records and the learner agent in RTE (advanced dynamic sequence), components management (LOs) – remember that these components consume and use XML files– then, this model provides the way to write, load, change and erase XML files in the server side.

AJAX is a way of developing Web applications to create interactive applications that is executed in the client side, in other words, the Web browser maintains the asynchronous communication with the server backstage. This way, it is possible to carry out changes in the same page without the need to reload it. On the other hand, the Struts Framework is a tool for Web application development under the Java MVC (Model-View-Controller) architecture. This Framework provides the advantage of maintainability, performance (tags pooling, caching, etc.), and reusability (it contains tools for the field validation that are executed in the client or server side).

6. Conclusions

This work has introduced an instance of an intelligent and interactive WBE system. Our approach focuses on reusability, accessibility, durability, and interoperability of the learning contents, which are built as LOs, as the main component for delivering teaching content.

LOs offer a common interface and functionality that makes the authoring of learning content that is delivered by dynamic sequencing easy. LOs accept feedback via assessment based upon the MAS platform. The information provided is considered as rough data because it is based on parameters elicited from the behavior of the learner.

MAS is used as an open middleware implementation. The technical goal of MAS is clear: To deploy distributed information technologies in such a way that the availability of services may be more efficient and flexible. Among the advantages stemmed from the use of MAS are the following: Systems integration can be performed in a high degree, the functionality of the LOs is increased substantially as well as the possibility of implementing different techniques, learning styles, instructional strategies, and interaction techniques.

References

[1] Plekhanova, V., Intelligent Agent Software Engineering, 1st edition, Idea Group Publishing, 2003.

[2] Autoreference1 – Magazine "Expert With Applications", Publishing by ELSEVIER, 2007.

[3] IEEE 1484.1/D9 LTSA (2001). Draft standard for learning technology – Learning Technology Systems Architecture. Retrieved January 11, 2006 from http://ieee.ltsc.org/wg1

[4] Sánchez-Arias V, G., Contreras J., Hernández N., "CCObÁ: Un ambiente colaborativo para el diseño, desarrollo y seguimiento de unidades didáctica basadas en la tecnología de objetos de aprendizaje" Memorias Congreso Internacional Virtual Educa Brasil 07. June 18-22, 2007.

[5] ADL - Advanced Distributed Learning Consortium. Retrieved January 24, 2006 from http://www.adlnet.org

[6] RDF - Resource Description Framework. Retrieved January 15, 2006 from http://www.w3.org/RDF/

[7] XML - Extensible Markup Language specification. Retrieved March 15, 2006 from http://www.w3.org/XML/

[8] JENA – A Semantic Web Framework for Java. Retrieved June 9, 2006 from: http://jena.sourceforge.net/

[9] JOSEKI Joseki - A SPARQL Server for Jena. Retrieved June 9, 2006 from: http://www.joseki.org/ [10] Panjabi, M.M., White, A.A. Basic biomechanics of the spine. Neurosurgery, Vol. 7, pp. 76–93 (1980).

[10] FIPA ACL message structure specification, XC00061. Retrieved August 20, 2006 from http://www.fipa.org/

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

[12] JADE version 3.6. Retrieved May 5, 2008 from: http://jade.tilab.com/dl.php?file=JADE-all-3.6.zip

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

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

[15] Autoreference2 – Chapter of the Book "Intelligent Agents in the Evolution of Web and Applications" for Springer, 2008.

[16] Sanders, W., Cumaranatunge C., ActionScript 3.0 Design Patterns, 1st edition, O'Reilly Media, Inc., Publication, 2007.

[17] Newcomer, E., Lomow, G., Understanding SOA with Web Services, 1st edition, Addison Wesley Professional Publication, 2004.

[18] Pokahr, A., Braubach, l., Lamersdorf, W., Jadex: A BDI Reasoning Engine. Programming Multi-Agent Systems, 1st edition, Kluwer Academic Publishers, 2005.

Authors´ Biography

Alejandro CANALES-CRUZ

He graduated in communication and electronic engineering and obtained his MSc in microelectronic science from the Instituto Politécnico Nacional, Mexico. He received his PhD in computer science from the Centro de Investigación en Computación of the Instituto Politécnico Nacional, Mexico. His main research lines are Web-based education, mobile collaborative learning, artificial intelligence based on multi-agent systems and design of secure software. He has been author, or co-author, of multiple papers published in international journals as well as chapters of scientific books and in peer reviewed conference proceedings.

Víctor Germán SÁNCHEZ-ARIAS

Dr. Sanchez graduated from the Faculty of Engineering of Universidad Nacional Autónoma de México, UNAM. He has a MSc in computer science research from the Instituto de Investigaciones en Matemáticas Aplicadas (IIMAS) of UNAM. He was granted the Advanced Studies Diploma (DEA) from Institut National Polytechnique de Grenoble (INPG), France, and a Ph.D. in computer engineering from INPG, France. He has been researcher at the IIMAS-UNAM and LANIA. He has published several papers in national and international journals. He has also been involved in the organization of various academic events and collaborated with other institutions on nationally and internationally research projects. Since March 2006, he has been director of the Center for Advanced Technology in Distance Education (CATED) of the UNAM, where he conducts research and innovation in the area of education supported by information technology and telecommunications.

Francisco CERVANTES-PÉREZ

Dr. Cervantes-Pérez received his B.E. degree in mechanical electrical engineering (Control, Communications and Electronics) from Universidad Nacional Autónoma de México, UNAM; then, he pursued his masters´s studies in electrical engineering (Digital Electronics and Microprocessors) at the same university, and he got his Ph.D. in computer and information sciences from the University of Massachusetts at Amherst, Mass in the USA. Currently Dr. Cervantes-Pérez is a researcher at the Center for Applied Sciences and Technological Development at UNAM, and the director of the Coordination of Open University and Distance Education of the same university. He has been author, or co-author, of multiple papers published in international journals as well as chapters of scientific books, and in peer reviewed conference proceedings. His main research areas of interest are intelligent computing, computational neuroscience, information systems and the use of technology in education.

Rubén PEREDO-VALDERRAMA

He received the B. Sc. degree from the Escuela Superior de Ingeniería Mecánica y Eléctrica, ESIME, the M. Sc. degree from the Centro de Investigación en Computación, (CIC), Instituto Politécnico Nacional, IPN. He is currently candidate to a Ph.D. degree at CIC – IPN. His main research lines are Web-based education, Semantic Web, multi-agent systems and multimedia. In 1999, he joined CIC - IPN, where now he is working in the area of artificial intelligent (AI). Currently, he is a level-1 member of the SNI and has several publications in indexed international, national and institutional journals. He is author of a book chapter Springer and has several other publications and conference proceedings internationally and nationally.