International Journal of Information Technology & Decision Making
Vol. 12, No. 2 (2013) 277–307

DOI: 10.1142/S0219622013500120

PERSONALIZED KNOWLEDGE MANAGEMENT IN ENVIRONMENTS OF WEB-BASED EDUCATION

RUBÉN PEREDO VALDERRAMA

Superior School of Computer Science of National Polytechnic Institute
Unidad Profesional 'Adolfo López Mateos'
Col. Lindavista, Del. Gustavo A, Madero
D.F., Mexico City 07738, Mexico
rperedo@ipn.mx

ALEJANDRO CANALES CRUZ

Distance Education and Open University Coordination of Autonomous
National University of Mexico, Circuito Exterior
Cd. Universitaria, Del. Coyoacan
D.F., Mexico City 04510, Mexico
alejandro_canales@cuaed.unam.mx

ADRIANA N. RAMÍREZ SALAZAR

Interdisciplinary Professional Unit in Engineering and
Advanced Technologies of National Polytechnic Institute, Av. IPN No. 2580
Colonia Barrio la Laguna Ticomán Del. Gustavo A. Madero
Mexico City 07340, Mexico
adna76@hotmail.com

JUAN CARLOS CARAVANTES RAMÍREZ

Computer Science Research Center of National Polytechnic Institute
Unidad Profesional 'Adolfo López Mateos'
Col. Nueva Industrial Vallejo, Del. Gustavo A, Madero
D.F., Mexico City 07738, Mexico
caravanjc@yahoo.com

This paper is focused on the use of Knowledge Management techniques to develop environments of Web-Based Education to leverage the individual learning process for learners. Evolving use of Knowledge Management plays an important role to enhance problem solving skills. Recently, innovative approaches for integrating Knowledge Management into practical teaching activities have been ignored. The knowledge is defined as combination and organization of data and information in given context, and Knowledge Management provides acquire, storage, sharing, application and creation of knowledge objects. A new Architecture for the development of adaptive and intelligent Web-Based Education systems is presented. This architecture is based on a Learning Model for Personalized Knowledge Management.

Keywords: Multi-agent system; web semantic; web-based education; knowledge management.

1. Introduction

Web-Based Education (WBE) is currently an important research and development area and it has opened new ways of learning for many people. No one disputes whether WBE should be applied or not. The questions are how and when it should be applied. Benefits of WBE are: time and place are no longer barriers. Although this sounds rather promising, in fact there is little knowledge in the field of how to integrate all these different approaches and technologies in order to create the best possible WBE solution for the many different needs and situations.

According to Mason,[1] some of these needs are lifelong learning with its emphasis on just-in-time learning or personalized learning, and a learner-centered approach has helped to develop the interest in online courses using a resource-based model. The underlying aim of such courses is learning how to learn; that is, facilitating Knowledge Management (KM) skills such as searching, selecting and synthesizing information, discovering how and where to find answers and solutions, and understanding, transforming and presenting ideas. The Web provides both the cause — a wealth of information — and the means for doing this.

But some basic problems faced by the WBE and your learning community is how to improve the interactivity in the teaching–learning process, the changing role of the tutor from sage to guide, the need for KM skills and for team working abilities, and the move towards resource-based rather than packaged learning.[2] This indicates that competition will be driven by knowledge revolution in the future. Training learners to apply knowledge continues to be important as society evolves. Integrating KM into practical teaching activities is one of the best methods for learners to enhance their abilities in KM and problem solving.

Traditionally, KM focus on business,[3] but their use for education has not been ignored. The reason is that the application field is recent from the last decade. See Fig. 1 for a snapshot of the history of KM.

Other researchers have demonstrated how combining KM and WBE through the use of intelligent systems increases the learning performance. Del Peso and De Arriaga discuss intelligent e-learning systems through automatic construction of ontologies. This process allows automatic updating of the knowledge bases used in intelligent e-learning systems to increase interoperability and communication among knowledge bases.[10] Lau and Tsui discuss how the integration of KM within an WBE environment can provide a learning grid that enables the learner to identify the correct learning objects associated with the learner's context, needs, and preferences.[14] Shaw shows that tools such as knowledge maps can improve one's e-learning performance. Knowledge maps are similar to concept maps in showing visualized concepts, knowledge, and relationships.[12] In terms of improving e-learning performance, Ho and Kuo display through their research that organizations can improve adult workers' e-learning outcomes by facilitating positive computer attitudes.[15]

In this case, the paper presents a conceptual framework for personalized KM that consisting of a repertoire of methods, techniques, and tools with five activities

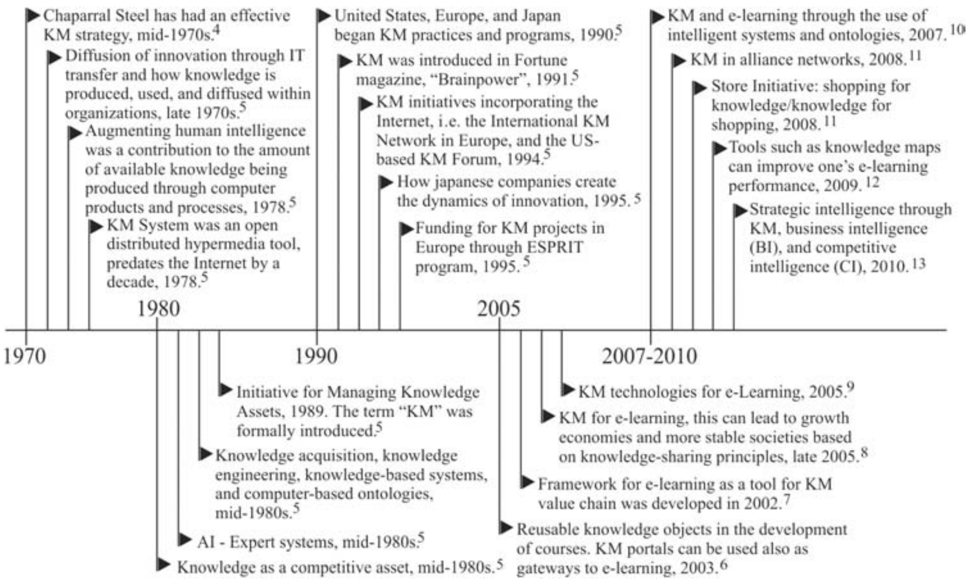

Fig. 1. KM time line.

(acquire, storage, sharing, application and creation) performed sequentially in environments of WBE. These activities are supported by the field of Artificial Intelligence (AI) and that they are attempts to understand and initiate human knowledge in computer systems. Besides, four main components of knowledge model are distinguished: a knowledge base, an inference engine, a knowledge engineering tool, and a specific user interface.[16]

The incorporation of knowledge base is one of the main improvement of this paper with respect to previous works[17,18] done by us. The knowledge base allows

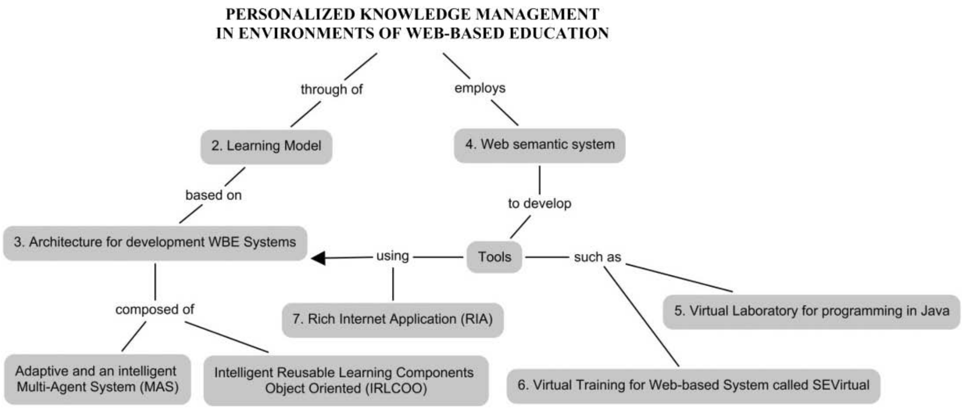

Fig. 2. Article structure outlined in a concept map.

integrating many information sources in the application. Decoupling the knowledge model from the application is critical to enabling reuse and data sharing between applications and users.

Figure 2 outlines the organization of the rest of the article.

2. Learning Model for Personalized Knowledge Management

In the literature, many published studies examine the operations and nature of KM. Beckman believed that KM consists of eight main steps: identify, acquire, select, store, share, apply, create, and sell.[19] Allee articulated that KM involves four levels: capture, sharing, application and creation.[20] Maryam and Dorothy's approach showed similarities to Allee's.[21] While considering the above proposals, this study organizes the KM process into five parts: acquire, storage, sharing, application and creation. The learning model for personalized KM is schematized in Fig. 3.

(1) During knowledge acquisition, learners actively access course related information.
(2) Learners start organizing knowledge during the storing step.
(3) The sharing step involves collaborating with other learners on collected knowledge and experience.
(4) During the stage of knowledge application, understanding how to classify and store knowledge is obtained.
(5) The last stage of knowledge creation, learners create knowledge through sharing and application.

The learning model of Fig. 3 has four main architectural components of knowledge model that are the knowledge base, the inference engine, the knowledge engineering tools, and the user interface. The knowledge base contains the domain specific knowledge that is used for learning problem solving in the domain. Knowledge can be represented and stored in the knowledge base with Resource Description Framework (RDF) for computer manipulation. The inference engine is based on an inference rule and a search strategy and contains algorithms. Algorithms are programmed by multi-agent system (MAS), and it is used to manipulate the knowledge stored in the

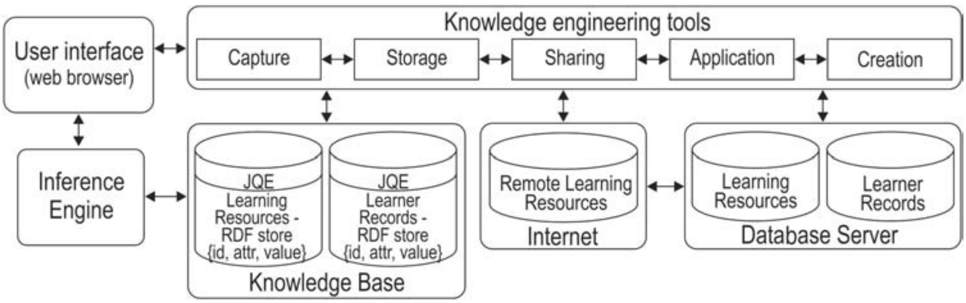

Fig. 3. Learning model proposed for personalized knowledge management.

knowledge base in order to solve learning problems. An inference rule is a way to deduce new knowledge from the existing knowledge base of rules and facts. The knowledge engineering tools enable users to store their knowledge in the knowledge base to deduce new knowledge from existing knowledge through a machine learning process. The interface is used for the system to interact with the user, the WBE environment and other systems such as databases.

The aim of this work is to show a new architecture for building applications with the ability to manage knowledge in a WBE environment but including the artificial intelligence. The result is an adaptive system with features such as:

- Supporting the learners' individual needs — there are two forms: allowing the learner to change certain system parameters and adapt their behavior accordingly; and the system is adapted to the learners' needs in intelligent form and automatically based on the system's conjecture.
- Each learner has a personal account for storing and sharing information. It also records each learner's schedule.
- The knowledge creation has been divided into application of prior knowledge and exploration of new knowledge. The best method guides learner in sharing, self-improvements, and knowledge creation.

3. Multi-Agent System

The agent-based systems are one of the most important areas of research and development in information technology in recent years, underpinning many aspects of broader information society technologies.[22]

The fundamental concept of agent-based systems is an agent. Jennings *et al.* suggest in Ref. 23 a definition of the term agent: "An agent is a computer system, situated in some environment, which is capable of autonomous actions in this environment in order to meet its design objectives".

Software agents are means to build distributed applications that enables us to create artificial social systems. Agents are characterized by, among other things, autonomy, proactivity and an ability to communicate.

3.1. *Main MAS scenario*

Our main MAS scenario has four agents: Learner, Coach, Evaluation and Delivery. The Learner agent collects information about the Learner, and serializes in RDF triples via IRLCOO, depositing this information in knowledgebases: the information describes the Learner's behavior in the Learning Management System (LMS) environment, enabling us to infer the Learner's behavior. The Coach agent provides a dynamic and personalized feedback at Run Time, based on the information collected on knowledgebases and databases (Learner Records and Learning Resources), the learning process design and physical coach. The Evaluation and Delivery agents work together to reconfigure dynamically the sequence and level of learning materials.

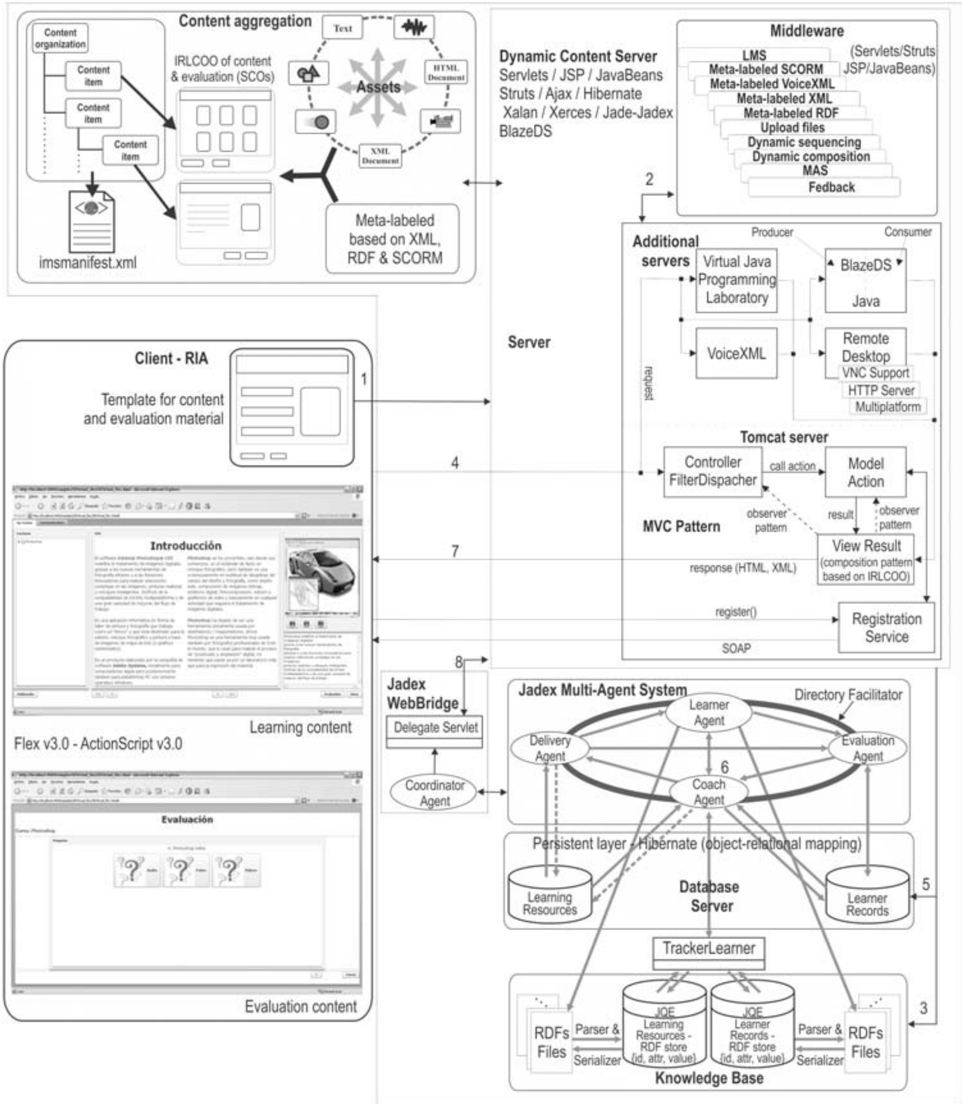

Fig. 4. Generation of learning and evaluation contents at runtime within an environment of WBE.

Figure 4 depicts the integration of the Web application and multi-agent platform based on JADEX and WebBridge.

IRLCOO was developed by Peredo *et al.*[24] The IRLCOO represent a kind of learning content characterized by rich multimedia, high interactivity and intense feedback that is supported by means of a standard interface and functionality.

The system for the generation of learning and evaluation contents at runtime within an environment of WBE is depicted in Fig. 4. The system is the result of an investigation and has materialized in a prototype described in this paper. The prototype will be tested with real users in a short time. Next, a brief description of this system is presented.

In Fig. 4, it starts in client side during the learner evaluation process. First, the Coach develops the content materials and assessments using the authoring tool of the SEVirtual described in Sec. 6. (1) The learner enters in the system, creates knowledge through interacting with your partners and the local and remote Learning Resources; the learner share and apply the information that they have. (2) The learner interacts with the content and evaluation materials and collaborative tools based on IRLCOO. (3) IRLCOO takes learner's metrics e.g., tracking learning, chat, discussion forums, completed activities, time, etc., depositing them in the knowledge base. (4) Once learner has presented the evaluation, (5) the results are deposited in the database. (6) The system responds executing among others the following main plans of Coach Agent: ReconfigurationPlan, DynamicSequencingPlan, and FeedbacksPlan; in order to provide to the learner a dynamic reconfiguration of contents, i.e, the system offers him other available materials. Finally, the learner receives a dynamic feedback depending on the learner's metrics and goals defined in databases and knowledge bases. (7) The dynamic feedback is processed and deployed in the Web Browser. (8) The communication problems between agents and the Web application with Model-View-Controller (MVC) pattern are solved with WebBridge, Sec. 3.5 described this action.

MAS was constructed with some technologies that are described in the next sections.

3.2. *JADE*

The Agent technology has been the subject of extensive discussions and investigation within the scientific community for many years, only recently it has seen any significant degree of exploitation in business applications. MAS are being used in a wide variety of applications, but adopting the agent oriented paradigm is not easy, because there are a number of domain independent issues that must be solved, being the communication an example. The fundamental in Java Agent Development (JADE) is to develop a middleware to provide the domain independent of the infrastructure, thus the developers are focused on the problem. JADE is possibly the most widespread middleware using the agent oriented paradigm.[25]

In agreement with Ref. 26, the reasons for using the JADE framework to build Web applications are:

- It is a middleware system with a flexible infrastructure allowing easy extension with add-on modules.
- The framework simplifies the development of agent oriented applications at a Run-Time environment, implementing features required by agents.
- It is written in Java taking the benefits of language and third party libraries, reducing complexity to construct MAS with minimal expertise in agent theory.

3.3. *JADEX*

JADE Extension (JADEX) is an agent oriented reasoning engine for writing rational agents with eXtended Markup Language (XML) and the Java programming language. JADEX simplifies the implementation based on BDI model, introducing beliefs, goals and plans as classes/objects that can be created and manipulated inside the agent. The beliefs in agents based on JADEX are any kind of Java object that are stored in a beliefbase. The Goals represent the concrete motivations that influence on agent's behavior. To carry out the goals the agent executes plans, which are procedural recipes coded in Java. The new application uses an Agent Definition File (ADF) based on XML to develop the MAS with JADEX, and uses Java classes for the plan implementation. The ADF is the specification of construction of an agent based on JADEX.[27]

3.4. *Belief-Desire-Intention*

The Belief-Desire-Intention (BDI) architecture introduced by Bratman is the most interesting and widespread agent architecture, based on the theoretical foundation and the number of implemented and successfully applied systems, as a model for describing rational agents. See Refs. 28 and 29 for more details.

The model consists of three concepts, where beliefs capture the informational attitudes, desires capture the motivational attitudes, and finally the intentions capture the deliberative attitudes of agents. The model has been adopted and transformed into a formal theory with an execution model for agents, based on beliefs, goals, and plans.[27]

3.4.1. *Plans*

The Procedural Reasoning System (PRS) is a framework for constructing Real-Time reasoning systems that can perform complex tasks in dynamic environments; JADEX supports two styles of plans. The called service plan is a plan that is usually running and waits for service requests. A service plan manages private event waitqueue and receives events for later processing. The called passive plan is the standard version of a plan, inside PRS systems. Usually such plan is only running, when it has a task to achieve, the triggering events and goals are specified in the Agent Definition File (ADF).

Plans play a central role in our system based on JADEX, because they encapsulate the procedure for achieving mental attitudes of MAS. Plans consist of two parts in Jadex. The plan class standard is a Java class that extends and redefines Plan class from JADEX framework and must implements the abstract body() method which is invoked after plan instantiation. The plan body is associated to a plan head in the ADF. The following code shows the Plans for dynamic reconfiguration of the course at Run-Time inside the ADF.

```
<!--Defining the Plan3 -->
...
```

```
<plan name="Plan3">
   <parameter name="idCourse" class="String">
      <goalmapping ref="reconfiguration.idCourse"/>
   </parameter>
   <parameter name="result" class="String">
      <goalmapping ref="reconfiguration.result"/>
   </parameter>
   <body class="ReconfigurationPlan"/>
      <trigger>
         <goal ref="reconfiguration"/>
      </trigger>
</plan>
...
```

The Coach can set initial goals to achieve in a group or individual, modifying the corresponding ADF. We have different service plans and passive plans, but three plans stand out among the others, ReconfigurationPlan, DynamicSequencingPlan, and FeedbacksPlan. When the learner has used the virtual environment, and presents the evaluation. In this moment the Evaluation Agent is executed, and retrieves the learner's metrics and the results of the evaluation. Finally, this agent compares the results with the expected goals. In the case of not satisfying the expected goals, the plans — ReconfigurationPlan and DynamicSequencingPlan — are executed. The first plan searches the materials on the topic recommended by the Virtual Coach, as well as materials from other authors on the topics where the learner has had problems. The second plan acts based on the results in the evaluation, the resulting action is to tailor the sequence and navigation of content in a course at run time pursuant to learner's needs. DynamicSequencingPlan modifies the imsmanifest_navigation.xml file and consequently the imsmanifest.xml file. The last plan enables the feedback, comparing learner's metrics and configuring at run time a dynamic personalized feedback depending on its results.

3.4.2. *Beliefs*

The agent's beliefbase represents its knowledge about the world. The beliefbase in JADEX is similar to a simple data-storage, which allows communication between different plans by means of shared beliefs. Opposite to most PRS BDI systems, JADEX allows to store arbitrary Java objects as beliefs in its beliefbase. There are two kinds of beliefs in JADEX. There are beliefs that allow the user to store exactly one fact and other beliefs that allow storing a set of facts. The beliefs and belief sets as primary storage capacities for the plans is used wide, being provided also with the Object-Query-Language (OQL) that allows simplifying the queries in environment agent-oriented. Enabling us to monitor single beliefs with respect to their state and cause an event when a condition is satisfied. The following snippet shows the code for beliefs extracted of the databases of Fig. 4 at Run-Time inside the XML capability.

```
<!--Defining the capability -->
<capability ...>
...
<!--Defining the beliefset db -->
<beliefs>
...
   <beliefset name="db" class="Tuple">
   </beliefset>
...
</beliefs>
...
<!--Defining the expression findlearnername -->
<expressions>
...
   <expression name="findlearnername" exported="true">
      select one $wordpair.get(1)
      from Tuple $wordpair in
      $beliefbase.getBeliefSet("db").getFacts()
      where $wordpair.get(0).equals($surname)
      </expression>
...
   </expressions>
...
</capability>

<!-- Evaluation of the expression from the plan -->
...
```

3.4.3. *Goals*

The Goal-oriented programming is one of the key concepts in the agent-oriented paradigm. It denotes the fact that an agent commits itself to a certain objective and may tries all the possibilities to achieve its goal. There are four types: achieve goal, query goal, maintain goal and perform goal.[30] The next code shows our achieve goal for to take metrics of course at Run-Time inside the ADF. The goal receives the metrics of the course with the following parameters: topic, write, idCourse, email, QuestionName, QuestionIndex, and the results are compared with the goal established by the Coach. If there is a difference, ReconfigurationPlan, DynamicSequencingPlan, and FeedbacksPlan are executed, in order to provide to the learners a reconfiguration, sequenciation and feedback personalized of the course content.

```
<!--Defining achieve goal metrics -->
...
```

```
<achievegoal name="metrics">
   <parameter name="topic" class="String"/>
   <parameter name="write" class="String"/>
   <parameter name="idCourse" class="String"/>
   <parameter name="email" class="String"/>
   <parameter name="QuestionName" class="String"/>
   <parameter name="QuestionIndex" class="String"/>
   <parameter name="result" class="String"/>
</achievegoal>
...
```

3.5. *Capabilities*

In our MAS, different agents often needed to use the same/similar functionalities that incorporate more than just plan behavior. Private or shared beliefs, goals and plans are a common functionality of agents. These units of functionality are comparable to the module concept in the Object Oriented Programming (OOP), differing mainly in the use of mentalistic notions. The capability concept was originally introduced by Busetta *et al.* in Ref. 31 and enhanced in Ref. 32, enabling packaging of a subset of beliefs, plans, and goals into an agent module and enabling better reuse of these modules in the system. The capability structure of an agent has a hierarchical structure; all elements of a capability have private visibility by default and need to be explicitly made available for usage in a connected capability. WebBridge is a capability used in the system that makes use of capabilities, enabling us to make use of agent technology and Web presentation in a transparent manner, the following snippet shows the code for using the WebBridge as a capacity in the system inside the ADF.

```
<!--Defining the capability of the WebBridge -->
...
   <capabilities>
      <capability name="webcap" file="WebInteraction"/>
   </capabilities>
...
```

In accordance with Fig. 4, the WebBridge helps to avoid problems between agents and the Web application with MVC pattern. When a request from the Web browser requires the execution of application logic, the agent layer is applied. First, the request is transferred from the delegate controller to a generic coordinator agent, which acts as a mediator between the MAS and the Web layer. The coordinator is responsible for finding an application agent to process the request. Once the appropriate application agent has been identified, the coordinator sends a message to this agent with the parameters about the request. The request is transformed to an agent message by the coordinator, the application agent does not need to know, if the request comes from the Web layer or another source. Later the application agent sends the result to the coordinator, which forwards it to the controller. At last a view is deployed based on a JSP page, which reads the results created by the application agent and displays it to the user. This operation allows solving the problems between the Web presentation and the layer of agents in the application developed by us.

3.5.1. *Interaction protocols capability*

The interaction protocols are patterns of allowed message sequences for different interaction purposes. Various interaction objective patterns are reusable in different domains. FIPA has standardized numerous domain independent protocols.[33] JADEX has built in support for the majority of these FIPA protocols, facilitating the development of standardized interactions. The specifications do not consider how these interactions should be interrelated in domain activities. The protocol domain interactions are extended specification in goal oriented interaction protocols. The domain and protocol layer encapsulate the complete functionality of a role.

The goal signatures contain information about the in and out parameters that are used. The protocols capability contains the functionality: initiator and participant sides of the interaction. Protocols capability use within MAS needed to include the capability in sections of ADFs. In plans, the protocols create a goal instance, setting the needed parameter value and dispatching it, when the protocol has finished the results can be read in parameters of the goal. The Request Interaction Protocol manages the interaction of an initiator and an agent, to perform some action.

3.6. *Events*

JADEX has two communication levels, the intra-agent and internal events. The first is used when two or more plans want to exchange information based on asynchronous message event passing, being one of the most common methods based on beliefs. Beliefs in JADEX are containers for Java objects, using the agent concept in conceptual modeling; an advantage is the use of conditions to trigger events depending on belief states, and it is possible to formulate queries and retrieve entities that match up to the query expression. The internal communication uses explicit internal events, dispatched to all interested plans, without supporting any BDI-mechanism.

3.7. *Directory facilitator*

The initial states of agents are determined among other things by our: beliefs, goals, and the library of known plans. JADEX uses a declarative and a procedural approach to structure the components of an agent. In addition to the BDI components, other information stored in the ADF are the service descriptions for registering the agent at the Directory Facilitator (DF).

The DF identifies which agents are active in the MAS, their location in the environment, services provided and the resources needed. It also notifies to the others agents when a new agent joins the MAS, the resources to use it and the services provided.

The fundamental difficulties for integrating agents and Web services are the mismatches in description and communication used. The proposal is based on a proxy-based integration approach that allows both technologies to work in parallel without restrictions, providing a gateway between agents and Web services. It enables run time deployment and control of Web services and agents.

The DF allows the agents to register their services and search services offered by other agents. The registered services must be refreshed by the agent before the time expires. We have two options to register an agent using lease times, by means of achieve goal "df_register" and the achieve goal "df_modify" to manually refresh the registration. The other option is using the maintain goal "df_keep_registered" with automatic refresh.

The DF shows service descriptions, specified in the FIPA Agent Management specification.[33] The services are described in DF service descriptions. Figure 4 shows the DF integrated into our Web application.

3.8. *Jadex WebBridge*

The Jadex WebBridge framework provides an instructive application that combines agent technology with the Web content presentation. It allows an easy integration of Web applications based on agents interacting with a user through a Web interface.[34] The foundation of the WebBridge architecture is the MVC pattern for Web development.[35] Figure 4 depicts the adaptation of the WebBridge environment in the Web application, combined the agent technology with Web content presentation. In Server side, the Delegate servlet is responsible for receiving the delegate requests to the tier agent. The delegated request is received by the Coordinator agent running on the Jadex agent platform. The Coordinator agent forwards the request wrapped in a message to the agent tier. The MAS dispatches the request depending on the configuration file of the Web application, the Coordinator agent creates the necessary agents of the appropriate type. Finally, the MAS will process the request, creating a view model, which is returned to Coordinator agent and dispatched to the Delegate servlet.

4. Web Semantic System

The Semantic Web is basically a Web of metadata described and linked in different ways. As its basis, we identify metadata, or information about information, that unambiguously specify machine-understandable facts about Web resources. The applications can add semantics through programming instructions; nevertheless, there is no formal standard for the semantics programming. Even though it is possible to create metadata documents semi-automatically, a more precise approach requires human intervention. Much of the information that can be usefully specified for a resource simply cannot be extracted without some kind of human interpretation. Also, the metadata that is to be recorded about a resource is often derived from a vocabulary of interesting categories that are relevant for subsequent processes. These vocabularies, called ontologies, can be required to adhere to standards that can only be applied by humans. Once this necessarily intelligent work has been performed, one of the biggest claims that the Semantic Web makes is that by formalizing the expression of the semantics of the information, every automatic application will be able to manage, understand, and reason on it.

The information is represented in the system as a set of assertions called statements denominated triples, with three parts: subject, predicate, and object; RDF language is used to represent these triples.[36] RDF is an optimized data model for sharing and interchange. The RDF serialization is done through the framework JENA via Learner's IRLCOO (see Refs. 17, 18 and 24 for more details). The information exchange is done when the abstract model is converted in a concrete format; it is based on Friend of a Friend (FOAF) ontology and files. FOAF is a linked information system, built using Semantic Web technology. The FOAF project was initiated by Brickley and Miller in early 2000,[37] and its goal was to create a Web of machine-readable pages describing people, the links between them and the things they create and do. A FOAF document is written in XML syntax, and adopts the conventions of RDF. The most important component of a FOAF document is the FOAF vocabulary, which is identified by the namespace URI "http://xmlns.com/foaf/0.1/". We have created a simple form where the learner can create his/her FOAF profile. The application generates a RDF file which can be downloaded by the user and published in his/her homepage in order to be catched by MAS. FOAF is also a machine readable ontology describing persons, their activities and their relations to other people and objects, and is expressed using RDF and the Web Ontology Language (OWL). On the one hand, the expressiveness and flexibility of RDF are not sufficient because it lacks an explicit support for specifying the meaning behind the descriptions. On the other hand, the RDF Schema provides us with the capability to add semantics to the Web application based on RDF data. Thus, the semantic Web enables dynamic machine processing for discovery, invocation, and composition of Web Services. For example, some Learners' goals are achieved when the system provides a rich semantic model for expressing the domain knowledge, and allows the exchange of information between different domains. The knowledge models are based on RDF/OWL to utilize the expressiveness of its semantics, using the JENA framework with integrated components to provide the RDF storage and retrieval, as well as the interpretation of OWL semantics. The knowledgebases are components that represent our collection of information that is accessed, described and processed ontologically in the Semantic Web application. The knowledgebases are a collection of facts (statements) collected by IRLCOO components.

We employ FOAF ontology using the class Learner and many of its objects to describe learners, their relations, and activities, institutions, and organizations related to education. The FOAF profiles, once learners have created them; they will be automatically added as instances of our ontology. The main classes and objects of FOAF vocabulary are expressed in RDF/OWL and use OWL rules and inferences. The ontology is created using Protégé Ontology Editor 4.0 Alpha, and making use of the Ontology Reasoner Pellet 1.5.2 and RDF Query Language (SPARQL) to generate knowledge and different statistics from the user profile data collected.

The FOAF ontology uses a predefined, reserved vocabulary of terms to define concepts and relationships of social networks, it is possible to relate learners through a knowledge model that contains typical social attributes such as: name, email address, likes, interests, work, friends, etc. Using FOAF learner's profiles within our Web application, it enables semantic machine readability for process data and enabling exchanging data. These profiles are connected with other learners' profiles through the property foaf:knows to find matches.

The FOAF ontology is employed within the Semantic Web environment that was implemented with: Java 1.6 Software Development Kit (SDK), Jena Semantic Web Framework, and other technologies mentioned in the construction of the ontology, see Refs. 38−41 for more information. The Jena Semantic Web Framework was selected because it provides a programmatic environment based on OWL, RDF, RDFS and SPARQL, counting with a rule-based inference engine.[41] The tools use the Java programming language.

Thus, the FOAF subsystem is used for building a model to describe learners. This model includes personal information, the relationships inside of their social life from different communities on the Web. The model helps to build a solid knowledge base based on typical social attributes. The FOAF subsystem generates a Semantic Web instance data of each learner and their relationships by the framework Jena,[41] moreover, it builds and publishes the learner' FOAF model. The FOAF ontology comprises social information that FOAF subsystem shares and combines for searches. It comprises learner's information collected via Learner Agent and IRLCOO components, providing to the Web application with social information. The Semantic Web statements coded within the FOAF subsystem describe the learner and their relationships with friends, the statements deliver information about the learner in the response/contact list using query searches. The FOAF subsystem loads the ontology and instance data, it comprised the retrieval of defined sets of social information from our Web application using SPARQL Protocol and SPARQL.[42] When translating instance data, a query language such as SPARQL seems to be best appropriate. SPARQL has the advantage to be widely used for querying RDF data on the Web. This makes SPARQL-based data translation more usable for semantic Web users in comparison to rule languages or XML-based extraction techniques. In others words, each FOAF ontology consists of three intertwined aspects: aspects related to the ontology structure, aspects related to the logical axioms such as disjointness, and aspects related to entities names. For detection we can use the SPARQL language.

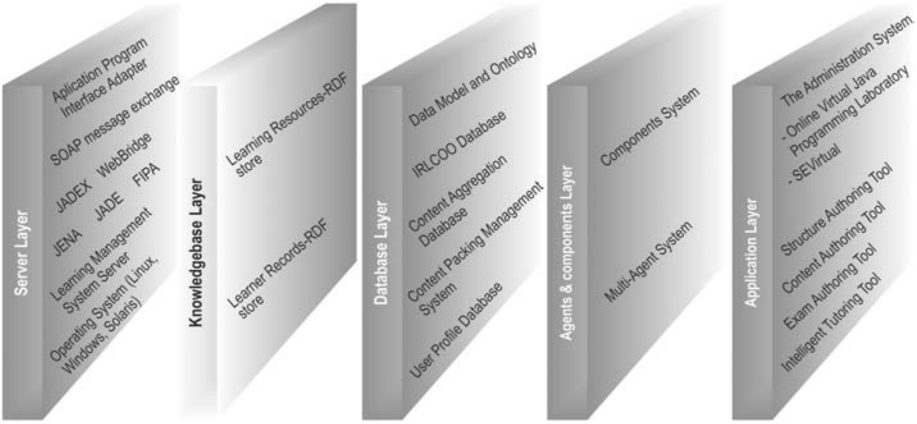

Fig. 5. An integral architecture for WBE systems.

Besides, for solving problems raised by heterogeneous ontologies can be achieved by matching the ontologies and processing the resulting alignments. This is typical of data mediation in which the data must be translated from one knowledge source to another. We propose to solve the data translation problem, i.e., the processing part, using the SPARQL query language. Finally, the Web application uses two reasoners: Jena and Pellet,[39] to infer social information at models based on the ontology. The application performs inference using the reasoning engine based on JENA that infers new information based on the contents of knowledgebases. Further the application collects learner's feedback scores from several modules and creates a reputation concept for learners. The concept can choose to include information or not depending on the coach's interest and values for a given situation. Conflicting information is incorporated into the concept, influencing the learner's reputation.

Figure 5 illustrates an overview of how Agents and Components Oriented Architecture is integrated. In general, the software architecture is divided into four layers: application, agents & components, database, knowledgebase, and server layers. The application layer includes an administration system, which is the ADL platform, to allow system administrators, instructors, and learners to manage learner records and curriculum. On the left side below the administration system, our asynchronized systems are incorporated. Thus, structure administration systems are separated from the learning content.

Figure 5 includes features of the architecture for WBE systems developed by Canales and Peredo,[18] and Peredo *et al.*[43] This architecture adds a new layer called Knowledge base.

5. Online Virtual Java Programming Laboratory

The objective of this work consists of explaining briefly the operation of an online virtual Java programming laboratory. Virtual laboratory is a place where there are not learners or tutors physically, but the learners can practice remotely. Virtual laboratory was developed depending on WBE paradigm and added with a MAS. The benefits offered by this kind of applications are: the learners can use this virtual laboratory wherever and whenever they want, as long as they have a connection to the Web; they can make progress to their own step; and it can be customizable in accordance with each learner's needs. Also, language configuration issues are reduced by having centralized the compiler. The purpose of this software tool is offering advanced practices in an Integrated Development Environment (IDE), with the objective of providing enhancing experiences.

Virtual laboratory has the function of acquire, storage, sharing, application and creation of knowledge objects. These knowledge objects are educational materials by using the Java compiler, in this case applied to an evaluation system, but not limited to this, being capable to be expanded to other scopes, such as contents and other methodologies.

Other advantages of the virtual Java programming laboratory versus on-class Java programming laboratory are:

- Reduce the cost of installation and maintenance.
- It is a self-learning tool.
- Education materials with a compiler online integrated.
- The workout time available is greater for learner's needs.
- The learner does not need to have the simulation software.

However, some disadvantages are:

- You cannot completely replace the practical experience of classroom instructor.
- There is a risk that learner will behave as a spectator.

5.1. *System general architecture*

This system implemented the MVC design pattern through the Apache Struts framework.[44] As shown in Fig. 6, the core of this system is the online Java compiler that uses the "javac" command in order to compile the classes, and Reflection[45] to invoke methods, both at Run-Time. In a few words, the system works as follows: on client side within the Web browser where the IRLCOO are loaded[35] and used as a Graphical User Interface (GUI) to enter the Java code to be compiled. Once entered the code, a servlet (CompiladorJava) is invoked in order to compile the code passing it by POST method. Internally, in order to compile, a .java file is created and compiled at Run-Time by using a suitable method inside this servlet. If the file cannot be compiled, then the corresponding error messages are returned. Otherwise, if the file can be compiled successfully, then a .class file with the same name is generated, and after that its main() method is tried to be executed by using Reflection. In other words, this class is loaded at Run-Time and its main()method is invoked. It is important to emphasize that the messages generated when the

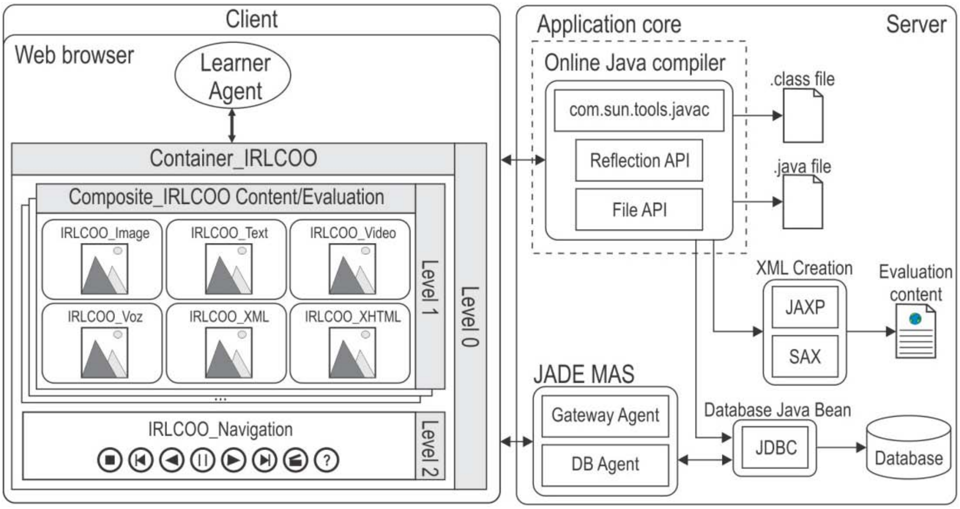

Fig. 6. System general architecture.

class is compiled and its main() method invoked, both are returned back to the client.

The virtual laboratory was implemented as an evaluation system. The information of the exams is stored in server side as XML files in a learner directory. Also, file paths are stored in a database through a JDBC Driver — using a Java Bean created with this purpose. Additionally, as can be observed in Fig. 6, the system includes another MAS for Online Virtual Java Programming Laboratory based on JADE.[25] MAS provides a dynamic feedback to the learner and this operation will be explained later. It is important to highlight the classes JadeGateway and GatewayAgent (from the package jade.wrapper.gateway) used to create a bridge between a servlet/jsp and a software agent with the purpose of interaction with the MAS via Web.

5.2. *Functions of the virtual laboratory*

The online virtual Java programming laboratory not only works as a remote compiling service, offers other functionalities or services such as:

- Remote compiling. The learner can compile Java programs at run time and sees the mistakes. When a mistake occurs the MAS can provide personalized feedback.
- Generation of assessments. The tutor can generate different types of questions, including questions about the source code and their syntax, see Fig. 7.
- Dynamic feedback at Run Time. The learner receives a personalized feedback during their learning process. This is possible because the MAS collects information of Internet (Remote Learning Resources), knowledgebases and databases (Learner Records and Learning Resources), and takes decisions on the

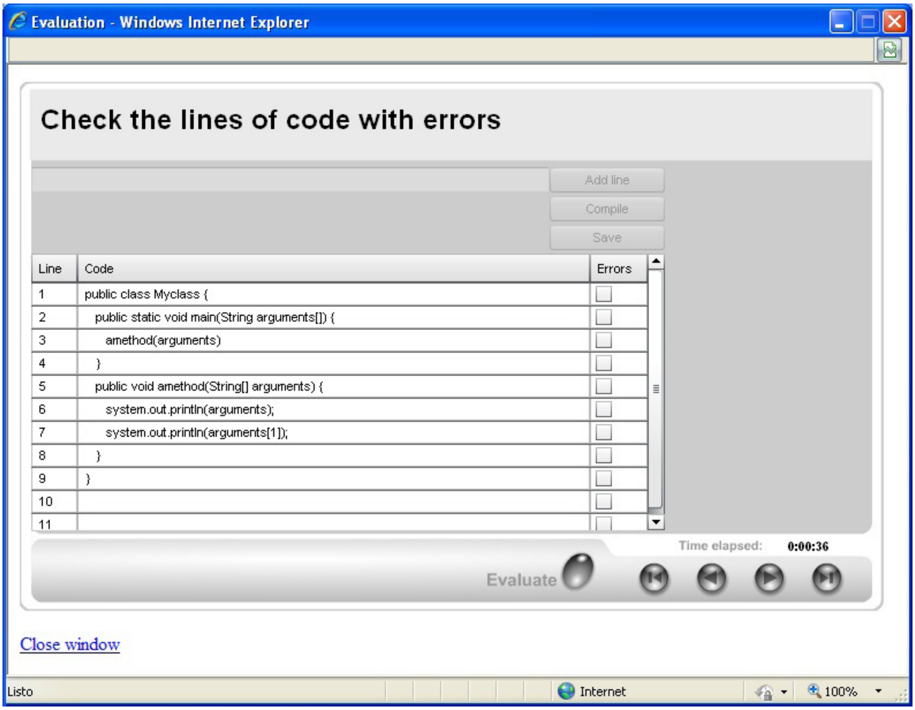

Fig. 7. Example of a question about the source.

recommendations of feedback for the Learner. The recommendations are based on the learning model for personalized KM depicted in Fig. 3. The Coach, Evaluation and Delivery Agents realize the functions of search, select and synthesis information, discovering how and where to find answers and solutions, and understanding, transforming and presenting ideas. Figure 8 shows the feedback interface where the red lines represent the mistakes, while the right side, the links to the feedback materials are placed.

- Store source code. The learner can make Java programs and save them in the system.
- Autocompletion based on AJAX. When learner enters the code in the compiler component, a list of matching words appeared pursuant to these letters, thus facilitating the capture of code. To develop the compiler component (IRLCOO), Action Script v3.0.46 is used. Figure 9 shows the online autocomplete.

5.3. *MAS for online virtual java programming laboratory*

The virtual laboratory was implemented with a basic MAS for the analysis of the learner's actions. In this case, to analyze and compare the time spent by each learner to solve a specific exam, and it provides a dynamic feedback. For implementing this MAS, JADE version 3.6 was used and the classes JadeGateway (singleton pattern)

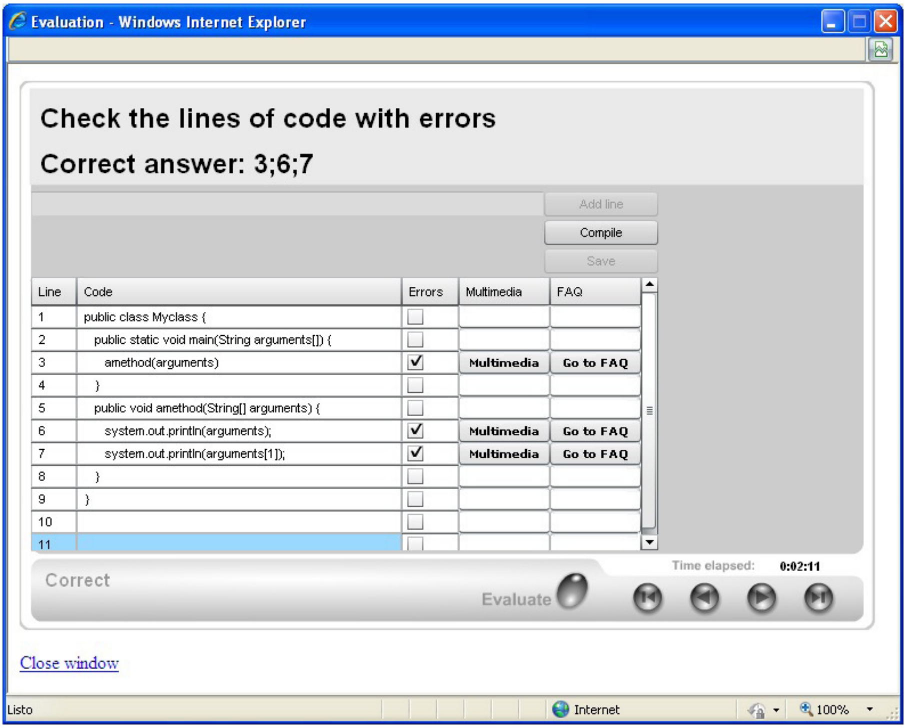

Fig. 8. Dynamic feedback at run time.

and GatewayAgent from the package jade.wrapper.gateway specifically. As shown in Fig. 10, MAS works as follows: first an HTTP request to the server is done by the Web browser and a POST message is generated, after that a servlet manages this message and invokes the SendMessageAction, this action creates a BlackBoardBean

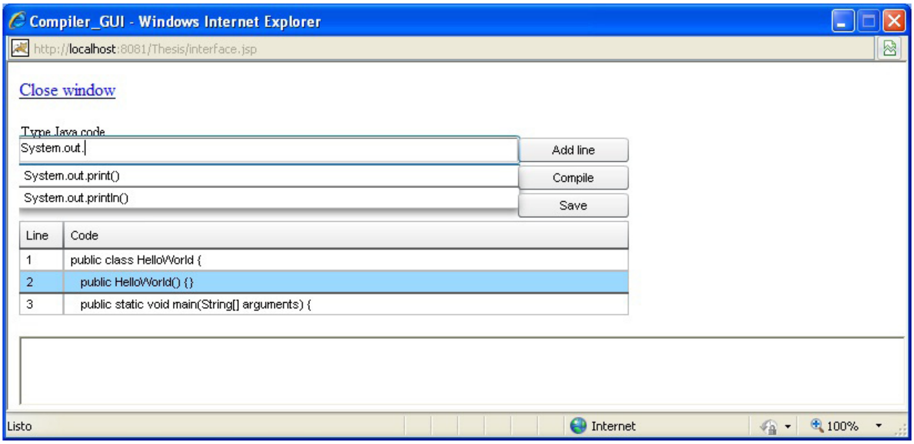

Fig. 9. Auto complete help in the InterfazCompilador component.

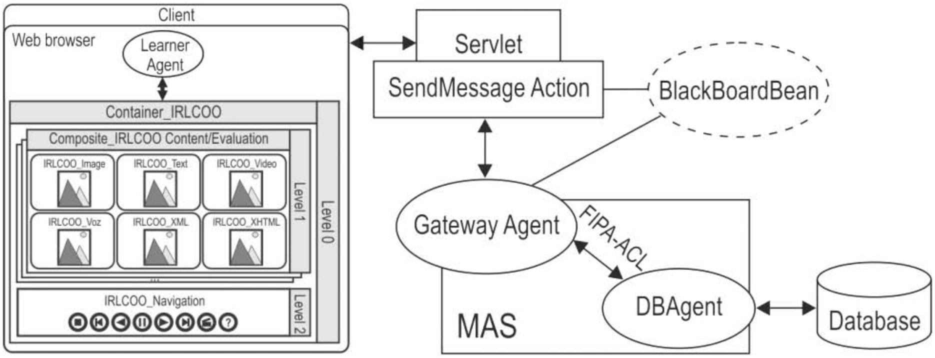

Fig. 10. Block diagram showing the MAS operation.

object which is a Data Transfer Object (DTO) that will be used as a communication channel between the servlet and the GatewayAgent through setter and getter methods. Then, the GatewayAgent obtains the BlackBoardBean object created previously in the servlet and extracts from it the recipient and the message. With this information, the GatewayAgent sends the FIPA-ACL message to the DBAgent that is the recipient, this one executes a query to the database, and with the obtained result the GatewayAgent responds to the DBAgent with a FIPA-ACL message, too. The GatewayAgent processes the response and sends it back to the servlet by using the same BlackBoardBean object mentioned previously, and this process ends when the servlet returns the response back to the Web browser.

6. SEVirtual

Virtual Training Web-based System called SEVirtual is composed of:

- Modules such as Login, Administrator (additions, deletions and modifications), Tutor, Learner, Generation of interactive materials, Courses, Assessment.
- Tools such as Chat, Interactive whiteboard, Remote desktop.

SEVirtual offers several functionalities for example:

- In order to facilitate the development of learning content, it was built as an authoring tool, which facilities the authoring content by 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. SEVirtual is used to construct Web-based courseware from the stored IRLCOO (Learning Resources). Developers choose one of the SEVirtual lesson templates and specify the desired components to be used in each item.
- The tutor can generate different types of questions, e.g., true and false, multiple choice, fill in blanks, selection of area, columns relate, etc.
- Personalized feedback to the learner during their learning process.

- Remote desktop enables to connect remote computers over the Web.
- In the electronic blackboard, the users may handwrite, edit and draw on the monitor that is like a chalk board.
- Online chat is a way of communicating by sending text messages to learners in the same chat-room in real time.

SEVirtual was developed with the integration of learning collaborative tools, an architecture based on agents, and using components of software.

SEVirtual integrates innovative components that enable collaboration and interaction among course participants. The components are independent units of production, acquisition and deployment, and also they are reusable. The components reuse is a major advantage by letting us to instantiate each component when it is necessary, and remove it when it is no longer needed, achieving great reusability, good design, implementation and system performance.

The major three goals of using software components are: conquering complexity because it is based on the principle of divide and conquer, managing change being that components are easy to adapt to new and changing requirements and reuse i.e., software reuse allows to design and implement something once and to use it over and over again in different contexts.[47]

The implemented components in the WBE system are: content generation component, audio/video synchronized component, assessment component, chat component, blackboard component and sharing desktop and resources component, the latter four components enable collaboration among learners.

SEVirtual converts the knowledge management process into practical teaching methods. Without being limited by a classroom, learners can browse materials and share the experience with others. They can also personalize knowledge organization and save it through the Internet. Learners can carry out self-learning which enhances problem solving skills. Learning activities are described as follows. When designing methods for knowledge acquiring, tutor should organize materials and analyze them because the learner has their learning style. This principle should guide learners when browsing materials and teach them how to acquire relevant knowledge and store information. The Personalized KM based on MAS offers each learner a personal account storing and sharing information. It also records each learner's schedule. Knowledge sharing methods should show how many articles users have posted to encourage the exchange and creation of knowledge. Group learning is recommended so that learners can participate in an organized way.

Figure 11 depicts the composition model diagram for the content generation component. Each time a subject of any type of component: Text, TitleDobleText, etc., is loaded and unloaded in the display of the course, the functions add_IRLCOO (IRLCOO:Component) and remove_IRLCOO(IRLCOO:Component) are invoked, the own function of each component is in the override method iterator(), in the same way when a component wants to access their children, it invokes the method getChild_IRLCOO(), the same way for the multimedia component which lets us to

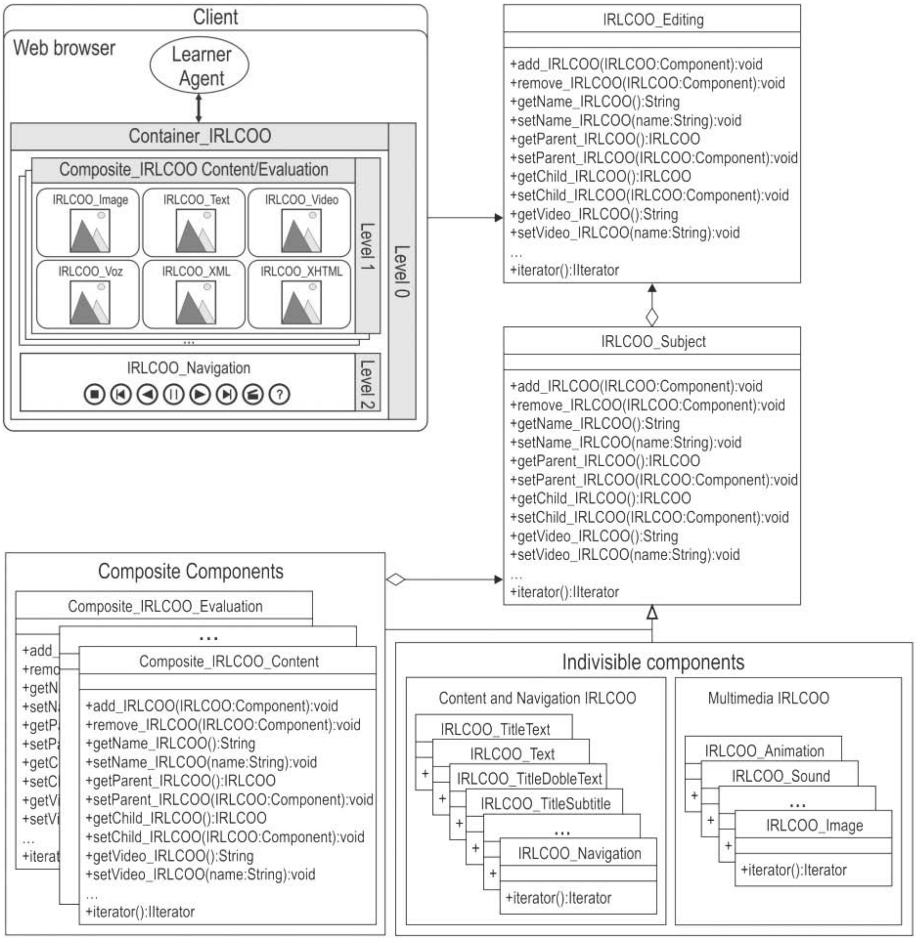

Fig. 11. Composition diagram of the component: Generation of content.

save some multimedia files as: audio, images and video to each subject. The composition models for the other components are similar. The IRLCOO are based on the composite pattern.

Figure 12 depicts SEVirtual with a Unified Modeling Language (UML) training course. On the left side of the screen displays the course content; on the center, the topics contents are linked with activities, practices of laboratory and homework; on the right side, the explanatory video with its captioning text at the bottom are shown. The navigation bar has a multimedia button to display the multimedia files of each topic and the evaluation button allows the learner to answer their evaluation of the course nevertheless, the learner can also navigate through the contents. The components allow the separation of content/navigation, as well as the composition/sequencing at Run-Time. Thus, the course can be adapted to the learner's needs.

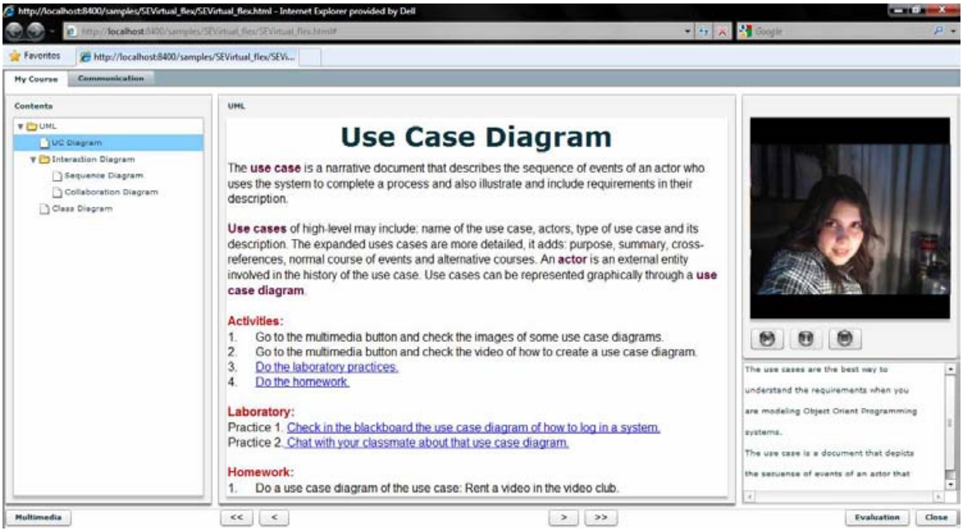

Fig. 12. UML training course.

The activities are linked with laboratory and homework. Figure 13 depicts the communication screen when the learner clicks at the link of one of the practices, on the left side of the screen displays the current participants of the course; on the center, the blackboard shows an example of use case diagram and the chat with a sample conversation of the activity; on the right side, the desktop sharing component and the download resources component allow the downloading of audio, video and image files.

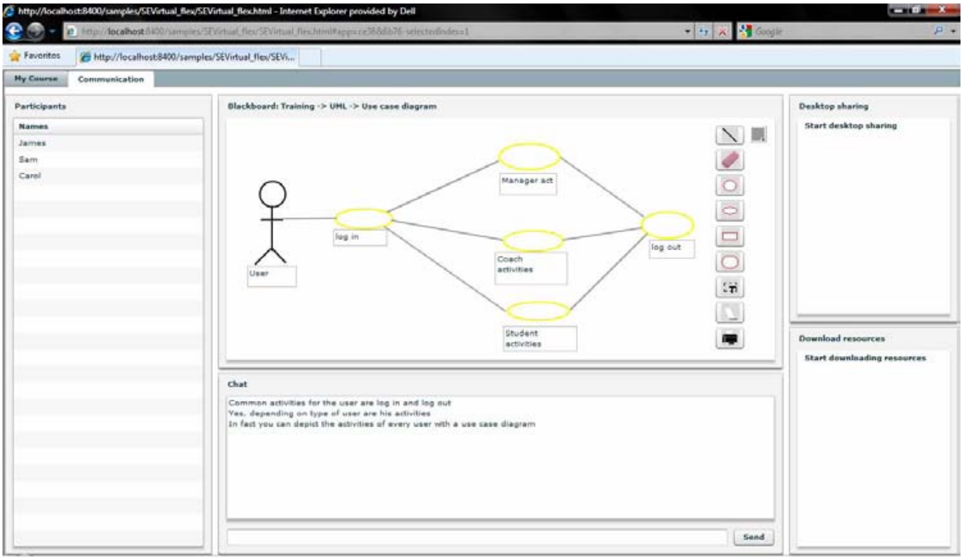

Fig. 13. Communication tab of UML training course.

SEVirtual automates the online courses production, being a useful tool for the tutor, and also would be available to all participants via Web, any time they have a connection to the Web. SEVirtual uses also the MVC design pattern through the Struts 2 framework,[44] for the middleware implementation on server side, providing a series of actions that can respond to the services on client side such as: course registration, save course, eliminate course, login, user registration, etc.

The learner receives personalized feedback from two types of approaches: (1) objective and reactive, through an assessment; (2) subjective and proactive based by MAS.

7. Rich Internet Application

The previous Web applications (SEVirtual and Online Virtual Java Programming Laboratory) proposed are Rich Internet Application (RIA). Using the free open source framework Flex 3,[48] which let us to create a RIA,[49] which is an interactive application that integrate multimedia, improve performance, and also allow two-way communication with the server without reloading the Web page. There are some important goals of using RIAs:

- Bidirectional communication with the server without reloading page.
- Multimedia integration.
- Media players integration.
- Eliminates roundtrip to the server.
- Interactive applications.
- Improved performance.

Flex 3 also allow us to work with new created software components and to use metadata as the data provider of some of the Flex's prefabricated components. Besides something important is that Flex 3 enables the communication with Java in our case Struts 2 and the visualization of the multimedia files is transparent due to the media players integration. The programming language used by Flex 3 is ActionScript3 (AS3).[46]

7.1. *Communication with the LMS*

The components are based on SCORM,[50] e.g., the chat component communicates with the LMS to indicate if the learner has successfully carried out the activity based on the use of chat as a target of the training course. The following code checks if the learner uses the chat, recording the activities of the learner in RDF triples via Learner's IRLCOO for the Semantic Web System, therefore it is possible to know the time that the learner has spent chatting, to later setting the target as complete/incomplete.

```
...
if(!ExternalInterface.available)
Alert.show("ExternalInterface is not available");
```

```
else
ExternalInterface.addCallback("asFunction", callActionScript);
setInitTime();
setInitObjective();
consumer.subscribe();
...
```

The following code block communicates with JavaScript (JS), being responsible of the communication with the LMS. The setInitTime() function calls a JS function named reportSessionTime(), this function initializes the time in the session.

```
function reportSessionTime() {
var dtm = new Date();
var n = dtm.getTime() - g_dtmInitialized.getTime();
return
SCOSetValue("cmi.session_time",MillisecondsToCMIDuration(n));}
```

The following code block communicates with a JS function named setObjective_js() and gives as argument the string “chat”, this function sets a new target within the LMS, which establishes the objective of the activity based on chat.

```
function setObjective_js(value){
var err = "true";
err = SCOSetValue("cmi.objectives.0.id", value);
return err;}
```

Once the learner has finished using the chat, the function close() is executed, this function gives the time spent during the session in the chat activity and calls the callActionScript() function. The following snippet shows the code that checks if the time spent in the session is greater than or equal to a specified activity time, the activity status is set depending on the results as complete/incomplete.

```
private function callActionScript(msg:String):void{
var t:Date = Date(msg);
if(t.getTime() >= time.getTime())
setStatus("completed");
else
setStatus("incomplete");}
```

The setStatus() function calls a JS function named setStatus_js(). The following snippet shows the code that sets the activity status to completed or incomplete depending on the value of the parameter: val.

```
function setStatus_js(val){
var err = "true";
err = SCOSetValue("cmi.success_status", val);
return err;}
```

For example SEVirtual is based on MAS (see Fig. 4), specially of the Coach Agent to evaluate the Learner, given a smarter feedback, reconfigured the course in accordance with the subjects that are difficult to the learner and given recommendations of books, exercises and Web pages about these topic, through the Delivery agent and using the components layer, adapting components in function of the Learner's metrics taken via IRLCOO. In this new architecture, the Learner's Agent serializes metrics of the Learner's behavior throughout course, allowing to the MAS based on these metrics, reconfigures the course, tailors the course and gives feedback and recommendations. The metrics are based on: tracking, assessments, activities, skills, successes and mistakes, etc.

Finally, SEVirtual and Online Virtual Java Programming Laboratory use a persistence layer based on Hibernate.[51] Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions. Hibernate generates SQL and relieves us from manually handling Java Database Connectivity (JDBC) result sets and object conversions. Metadata is defined in simple XML files.

8. Conclusion

The experience from this paper illustrates that KM process is not static; it evolves. Any KM for a WBE system must be able to support the acquisition, storage, sharing, application and creation of knowledge as a continual cyclical process.

The theoretical model for personalization of KM presented has an acceptable fit with the data collected but will greatly benefit from further validations with larger data sets and with more diversity in term of users represented.

The preliminary theoretical and practical findings of this research show that knowledge model plays an important role as well as the usage level of personalization technologies. The inclusion of ontologies in the development allows us to represent and share knowledge based on a common vocabulary and a specific format for the exchange of knowledge, providing a specific communication protocol, enabling the reuse of knowledge. The Semantic Web model was integrated in the Web application to separate domain specific business logic from the program itself.

The development proposed a new architecture that improves IEEE 1484 – LTSA architecture, adding: a metadata layer, a knowledge bases layer, and a MAS based on architecture BDI and another based on JADE. The layer metadata is implemented using JENA framework; the knowledgebases are filled via the content/assessment — IRLCOO components in conjunction with Learner's Agent, capturing the Learner's metrics within the LMS environment. The main MAS have four agents: Learner Agent, Evaluation Agent, Coach Agent and Delivery Agent. The main objectives of the MAS are to provide: dynamic reconfiguration, sequencing and feedback to the Learner; Learner's Agent via the content/assessment — IRLCOO components capture metrics within the LMS environment in knowledgebases, Evaluation Agent in conjunction with the evaluation — IRLCOO components capture the Learner's results within the LMS environment, Coach Agent/Coach takes decisions based on Learner's metrics in knowledgebases and relational database (Learner Records and Learning Resources), dynamically reconfigured and sequencing the course depending on the goals reached in the BDI model of the Coach Agent, carried out by means of Delivery Agent. The Coach Agent uses the TrackerLearner object to integrate a more comprehensive view about the Learner's status according to their results and metrics.

The Web application and Semantic Web allow acquisition of information, automating the persistence and retrieval of the information, allowing the integration of different perspectives about the Learner, providing a better vision of the state about the Learner to the Coach Agent/Coach, further is possible to figure things out through inference.

The Semantic Web is a new programming perspective that transforms distributed, confusing, and massive information into real solutions, giving to the metric taken a new dimension.

The software components enable: conquering complexity based on the principle of divide and conquers, managing change being that components are easy to adapt to new and changing requirements and maximize the reuse based on XML in different contexts. The components allow the separation of content/navigation, as well as the composition/sequencing at Run-Time. The course can adapt to the learner's needs at Run-Time. Furthermore, all courses developed with these tools are of RIA type.

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

Nowadays, we are also working on smart agents that use XML as the communication language and work with the second generation of Web Services. In addition, we are working out on learning styles that are suitable to apply to the WBE system.

Acknowledgments

Authors of this paper would like to thank the Universidad Nacional Autónoma de México (UNAM) — Coordinación de Universidad Abierta y a Distancia (CUAED), and Instituto Politécnico Nacional (IPN) – Escuela Superior de Cómputo (ESCOM) to the project SIP-IPN: 20121741 and 20131560 for the economic support for the development of this work.

References

1. R. Mason, Models of Online Courses (2001), http://www.usdla.org/html/journal/JUL01_Issue/article02.html.

2. L. Esnault, *Web-Based Education and Pedagogical Technologies: Solutions for Learning Applications* (IGI Global, USA, 2007).
3. E. Coakes and S. Clarke, *Encyclopedia of Communities of Practice in Information and Knowledge Management* (Idea Group, UK, 2005).
4. D. Leonard-Barton and G. Preuss, Chaparral steel: Rapid product and process development, Case 692018 (Harvard Business School, 1991).
5. K. Ghuman and K. Aswathappa, *Management: Concepts, Practice & Cases* (Tata McGraw-Hill Education, New Delhi, India, 2010), pp. 595–596.
6. J. Lamont, Special section: E-learning world: Bridging the worlds of e-learning and KM *KMWorld* **12**(2) (2003) 6–9.
7. R. Wild, K. Grigg and T. Downing, A framework for e-learning as a tool for knowledge management, *Industrial Management and Data Systems Journal* **102**(7) (2002) 371–380.
8. P. Barker, Knowledge management for e-learning, *Innovations in Education and Teaching International* **42**(2) (2005) 111–121.
9. M. Lytras, A. Naeve and A. Pouloudi, A knowledge management roadmap for e-learning: the way ahead, *International Journal of Distance Education Technologies* **3**(2) (2005) 1–11.
10. J. Del Peso and F. de Arriaga, Intelligent e-learning systems: Automatic construction of ontologies, *Current Themes in Engineering Technol* **1007** (2008) 211–222; doi: http://dx.doi.org/10.1063/1.2937608.
11. A. De Man, *Knowledge Management and Innovation in Networks* (Edward Elgar, Northampton, MA, USA, 2008).
12. R. Shaw, A study of learning performance of e-learning materials design with knowledge maps, *Computers and Education Journal* **54**(1) (2009) 253–264, doi: 10.1016/j.compedu.2009.08.007.
13. J. Liebowitz, Strategic intelligence, social networking, and knowledge retention, *IEEE Computer* **43**(2) (2010) 87–89, doi: 10.1109/MC.2010.52.
14. A. Lau and E. Tsui, Knowledge management perspective on e-learning effectiveness, *Knowledge-Based Systems Journal* **22**(4) (2009) 324–325.
15. L. Ho and T. Kuo, How can one amplify the effect of e-learning? An examination of high-tech employees' computer attitude and flow experience, *Computers in Human Behavior Journal* **26**(1) (2009) 23–31, doi: 10.1016/j.chb.2009.07.007.
16. S. Dhaliwal and I. Benbasat, The use and effects of knowledgebased system explanations: Theoretical foundations and a framework for empirical evaluation, *Information Systems Research* **7**(3) (1996) 342–362.
17. A. Canales, A. Peña, R. Peredo, H. Sossa and A. Gutiérrez, Adaptive and intelligent Web based education system: Towards an integral architecture and framework, *Expert Systems with Applications* **33** (2007) 1076–1089.
18. A. Canales and R. Peredo, Adaptive and intelligent agents applied in the taking of decisions inside of a web-based education system, in *Intelligent Agents in The Evolution of Web and Applications. Studies in Computational Intelligence* (Springer Berlin/Heidelberg, 2009), pp. 87–112.
19. T. Beckman, A methodology for knowledge management, in *Proc. IASTED Int. Conf. AI and Soft Computing* (Banff, Canada, 1997), pp. 105–120.
20. V. Allee, Twelve principles of knowledge management, *Training & Development* **51**(11) (1997) 71–75.
21. D. Ritchie and B. Hoffman, Incorporating instructional design principles with the World Wide Web in *Web-Based Instruction* (Educational Technology Publications, Englewood Cliffs, NJ, 1997), pp. 135–138.

22. I. Trencansky and R. Cervenka, *The Agent Modeling Language — AML: A Comprehensive Approach to Modeling Multi-Agent Systems* (Birkhäuser Basel, Slovakia, 2007).
23. N. R. Jennings, K. Sycara and M. Wooldridge, A roadmap of agent research and development, *Journal of Autonomous Agents and Multi-Agent Systems* **1**(1) (1998) 7–38.
24. R. Peredo, L. Balladares and L. Sheremetov, Development of intelligent reusable learning objects for Web-based education systems, *Expert Systems with Applications* **28**(2) (2005) 273–283.
25. Telecom Italia, Java Agent Development Framework (2010), http://jade.tilab.com/.
26. F. L. Bellifemine, G. Caire and D. Greenwood, *Developing Multi-Agent Systems with JADE* (Wiley Series in Agent Technology, Wiley, UK, 2007).
27. A. Rao and M. Georgeff, BDI Agents: From theory to practice in *Proc. First Int. Conf. Multi-Agent Systems (ICMAS'95)* (The MIT Press, Cambridge, MA, USA, 1995), pp. 312–319.
28. M. Bratman, *Intention, Plans, and Practical Reason* (Harvard University Press, Cambridge, MA, USA, 1987).
29. A. Pokahr and L. Braubach, JADEX User Guide (2007), http://www.agilemethod.csie.ncu.edu.tw/download/agent/userguide.pdf.
30. A. Pokahr and L. Braubach, JADEX Tutorial (2007), http://ftp.heanet.ie/mirrors/sourceforge/j/project/ja/jadex/jadex/0.80/tutorial-0.80.pdf.
31. P. Busetta, N. Howden, R. Rönnquist and A. Hodgson, Structuring BDI agents in functional clusters, in *Intelligent Agents VI, Proc. 6th Int. Workshop, Agent Theories, Architectures, and Languages (ATAL)'99* (Springer, Berlin, New York, 2000), pp. 277–289.
32. L. Braubach, A. Pokahr and W. Lamersdorf, Extending the capability concept for flexible BDI agent modularization, in *Proc. Third Int. Workshop on Programming Multi-Agent Systems (ProMAS'05)* (Netherlands, 2005), pp. 99–114.
33. FIPA XC00061, FIPA ACL message structure specification (2001), http://www.fipa.org.
34. A. Pokahr and L. Braubach, The webbridge framework for building web-based agent applications, in *Languages, Methodologies and Development Tools for Multi-Agent Systems* (Springer Berlin/Heidelberg, 2007), pp. 173–190.
35. W. Sanders and C. Cumaranatunge, *ActionScript 3.0 Design Patterns: Object Oriented Programming Techniques* (Adobe Developer Library, USA, 2007).
36. W3 Consortium, Resource description framework specification (2003), http:/www.w3.org/RDF/default.htm.
37. W3 Consortium, The Friend of a Friend (FOAF) project (2010), http://www.foaf-project.org/.
38. Sun Microsystems, Java Standard Edition (2010), http://java.sun.com/javase/downloads/index.jsp.
39. Clark & Parsia LLC, Ontology Reasoner Pellet 1.5.2 (2008), http://clarkparsia.com/pellet.
40. Protégé, Protege 4.0.2 (build 115) (2008), http://protege.stanford.edu/download/protege/4.0/installanywhere/.
41. Jena Semantic Web Framework, Jena — A Semantic Web Framework for Java (2009), http://jena.sourceforge.net/index.html.
42. SPARQL Query Language for RDF (2008), http://www.w3.org/TR/rdf-sparql-query/.
43. R. Peredo, A. Canales, A. Menchaca and I. Peredo, Intelligent Web-based education system for adaptive learning, *Expert Systems with Applications* **38** (2011) 14690–14702.
44. Apache Struts, Struts 2 (2009), http://struts.apache.org/.
45. Oracle Corp., The Reflection API (2009), http://download.oracle.com/javase/tutorial/reflect/.

46. Adobe Systems Incorporated, Action Script Technology Center (2010), http://www.adobe.com/devnet/actionscript/.
47. A. J. A. Wang and K. Qian, *Component-Oriented Programming* (Wiley-Interscience, USA, 2005).
48. Adobe Systems Incorporated, Adobe Flex 3 (2010), http://www.adobe.com/es/products/flex/.
49. Adobe Systems Incorporated, Adobe AIR (2010), http://www.adobe.com/mena/products/air/.
50. Advanced Distributed Learning, SCORM 2004 4th Edition Version 1.1 Documentation. (2004), http://www.adlnet.gov/Technologies/scorm/SCORMSDocuments/SCORM%202004%204th%20Ed%20V1.1/Documentation%20Suite/SCORM_2004_4ED_v1_1_Doc_Suite.zip.
51. JBoss Community Team, Hibernate (2010), https://www.hibernate.org/.