An Adaptive Web Platform based on a Multiagent System and Ontologies

Eliana Scheihing, José Carrasco, Julio Guerra, Denis Parra
Universidad Austral de Chile, Chile
{escheihi, selincarrasco, julioguerra}@uach.cl, denisparra@gmail.com

SUMMARY

This article presents an overall description of the SPORAS system, which has been designed and implemented to adapt sequences of hypermedia material from a web based course, while taking into account the student's individual learning style and knowledge level. The SPORAS system incorporates mechanisms for monitoring the user in background and for controlling his or her progress, as well as for resource sequencing, all of them supported by a multiagent system. The Felder-Silverman learning style model and the Bloom Taxonomy of Educational Objectives have been used as reference sources to model the student's learning style and for tracking the knowledge level in each Learning Objective. The using of ontologies has been thought to facilitate the knowledge sharing between SPORAS and other analog systems. Two characteristics distinguish this system from others that are similar: First, its ability to deduce the learning style of each student through unobtrusive observation and analysis of their actions; and second, it only suggests the order in which content should be presented in, and does not prevent the student from following a different sequence.

SPORAS was implemented by integrating servlets, javascript for user monitoring, a multiagent system based on JADE framework, a MySQL database and ontologies build in Protégé used as knowledge models.

Keywords: adaptivity, learning styles, e-learning, multiagent systems, ontologies

INTRODUCTION

The goal of adaptivity systems is to adapt themselves to the needs of the individual through the use of models that represent his/her objectives, preferences and knowledge [1]. Therefore, given that each student has distinct interests, knowledge and learning styles for mastering different course contents, it is natural, that one of the areas adaptivity systems can be applied to is education, [1][3]. Systems such as ISIS-Tutor [1][5], ELM-ART [1] and AHA [1][6] are examples of adaptive systems applied to education. In this context, the primary objective of SPORAS is to adapt the contents of a Web-based course by putting into sequential order hypermedia educational resources, based on the individual learning style of each student and his/her level of progress in each topic. A model for the student's learning style is constructed by analyzing the degree of interest he or she demonstrates for the materials that are displayed, as determined by the system. Each material must be characterized correctly for this purpose. The student's level of progress is determined based on these same resources and further confirmed by conducting evaluations. The system design seeks to stay within two restrictions that would differentiate SPORAS from other systems:

a. The learning style should be determined without administering an initial test.
b. The system only suggests, and does not impose, the resources that it believes to be most appropriate.

SPORAS ARCHITECTURE

SPORAS provides course content through a Web interface. A multiagent system is responsible for recording student behavior as he/she interacts with resources. The system constructs and updates the student's model and progress levels, as well as establishing the sequence in which content resources will be displayed. Course contents and its associated resources are specified in a Content Ontology. The student model, along with records of resources viewed and actions taken are stored in a database (see Figure 1).

The multiagent system was developed using JADE, a tool that complies with the development standard for FIPA agents. JADE provides a set of library functions, a debugging interface, and a performance environment that facilitates the development of multiagent systems.

Jakarta-Tomcat was selected as the Web server because it enables communication between servlets and JADE, given that both are implemented in JAVA.

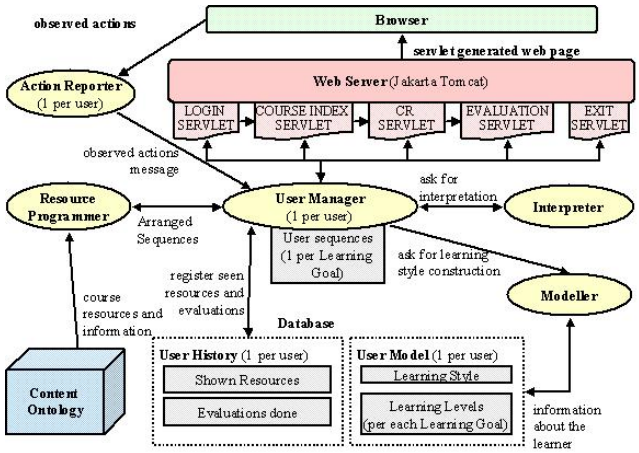

Figure 1: The figure shows the interaction between the Web server, servlets, agents, content ontology and the database

Ontologies

Protégé has been used for the development of both ontologies that have been implemented:

- The Content Ontology, used to describe and maintain course content, accessed through the API provided by Protégé, and
- The Multiagent System Ontology, which permits JADE agents to share a common language for message content. However, as this ontology is modeled on Protégé, it must be converted to the ontological schematic defined by JADE [2].

The database was constructed using MySQL and contains a) the structures necessary for maintaining data on each user account and student learning style, b) levels of progress towards objectives, and c) records of resources viewed and evaluations taken.

Multiagent System Ontology

The Multiagent System Ontology establishes the necessary concepts, actions and predicates in order to give meaning to the message content that constitute conversations between agents [2].

This ontology allows for the sharing of Content Ontology and database information, using a common format, in conversations carried out between the agents. At the implementation level, the multiagent system ontology is, finally, a set of Java classes that can be directly and conventionally accessed by the agents. Nonetheless, this ontology cannot be utilized as a permanent repository of information.

Contents Ontology: the course and study material

The course material and its organization are defined in the Contents Ontology. This ontology was constructed using Protégé-2000 and serves as a repository for course information.

This ontology defines two primary concepts (or classes): Learning Objective and Content Resource. Course contents are defined by a list of learning objectives, each one of which has a set of associated content resources and questions.

Learning Objective: The ontology breaks down a course, defining all "learning objectives" the teacher expects students to reach. It should be noted that an exhaustive definition of course objectives is essential, no matter how minimal or simple they may be. Each objective has among its attributes the following information:

- Resources: a set of Content Resources that corresponds to all resources directed to reaching the learning objective.
- Level: The level of objectives takes as its value one of 6 levels defined in Bloom's Taxonomy of Educational Objectives [4]: Knowledge, Understanding, Application, Analysis, Synthesis and Evaluation.
- Required Objectives: a list of additional objectives whose fulfillment is necessary for accomplishing the current objective.
- Questions: a set of Questions useful for evaluating each level the student completes until he or she reaches the level of the objective to be accomplished.

Content Resource: The Content Resource class defines a framework for characterizing the hypermedia course material. As part of its attributes each resource has:

- File: the archive name that is the resource itself. All files have been limited to the html format in order to simplify implementation.
- Level: can be any one of the levels until reaching the target level to which it belongs.
- Learning Style: Three parameters that indicate presentation, perception and processing characteristics, based on the Felderman-Silverman learning styles model. [3].
- Expected Time of Reception: the average time, measured in minutes, a student is expected to take to read, see, and process a resource.

Servlets

System Web pages are generated dynamically in the server using servlets. Each servlet carries out a specific task and constructs a different kind of page:

Servlet Login: is responsible for the user validation process, consulting the database when the user attempts to enter the system. It generates a welcome page or, when validation fails, an error page.
Servlet CourseIndex: request the list of personalized learning objectives according to the identified student's learning levels, using this information to present a "Course Index" page.
Servlet ContentResource: responsible for requesting the resource sequence for a learning objective. Depending upon the characteristics of the first resource in the sequence and the student's learning level, it decides whether to show a page with the resource content or to prompt the Evaluation servlet to generate an evaluation page.
Servlet Evaluation: When the ContentResource servlet decides an evaluation test must be given, the Evaluation servlet generates a page with a 4-question test, conducts the assessment, and then notifies the result to the agent platform (see 4. Tracking and Progress Control)

Platform Agents

The SPORAS multiagent system is composed of 4 types of agents:
UserManager: this agent responds to one specific user; thus there will be as many UserManagers as there are students connected to the system. The UserManager both receives and creates reports of monitored student actions (see 4. Tracking and Progress Control), as well as receiving, managing and responding to content requests made by the servlets (see 6. Adaptive Sequencing of Resources).

Interpreter: a type of agent responsible for the process of "interpreting" observed student actions when interacting with a resource. Depending on the system load level, as many types of this agent can be created as is necessary, When the UserManager agent asks for "interpretation" service, it interacts with the Interpreter agent, which is unoccupied at that moment.
Modeller: a type of agent that is responsible for constructing the learning style of a student by analyzing the record of viewed resources and the levels of progress made (see 5. Construction of the Learning Style). As with the Interpreter agent, more than one Modeller agent can be generated to respond to the actual load level placed on the system.
ResourceProgrammer: this type of agent can construct a resource sequence for a student as well as for a particular learning level (see 6. Adaptive Sequencing of Resources). As with previous agents described, more than one ResourceProgrammer agent can be generated to respond to the actual load level placed on the system.

OVERALL SYSTEM FUNCTION

To achieve adaptivity of course content, the system must solve three problems: a) to maintain and control a record of the student's progress according to the resources viewed and the student's success on assessments; b) to gradually determine the learning style of the student, using the Felder-Silverman model [3], and c) to construct ordered sequences of content resources, in which the first resource is the most appropriate for display, while taking into account student progress and his/her learning style. The solution to these problems is described in the three following subsections.

Tracking and Progress Control

Progress Measures

Before describing how progress control is achieved, it is necessary to delineate the specific measures used to maintain the student progress level. Every learning objective has three progress measures associated with it: work level, evaluated level and projected level. These levels use values defined in Bloom's Taxonomy of Educational Objectives [4].

1. **Work Level:** represents the level at which the student presumed to be. This level starts at the value 1 when the student first accesses the resources linked to a learning objective and the Work Level moves up upon successful evaluation.
2. **Evaluated Level:** this level starts with the initial value of 0 (zero) and increases when the work level is evaluated. It always has a value one unit below the work level.
3. P**rojected Level:** When the resource has been chosen and not suggested by the system, it takes the value of the resource level that has been viewed and accepted with the level above the Work Level

User Monitoring

In order to determine the degree of student interest for the information displayed, the system observes the actions the student takes in relation to each one of the resources shown on the screen. This process, which has been labeled "user monitoring," records certain events that can be captured with a browser: the amount of time the page is on-screen, scrolling movements, the selection of text or images, and the movement of the mouse over links.

Monitoring is achieved through JavaScript functions in the browser and communicated by an ACL message generated and sent by an applet to the agent platform, which is located in the server. This applet, known as ActionReporter, is considered a design-level agent even though it is not in fact a JADE agent.

The message sent by the ActionReporter is received by the student's UserManager agent, which then records the observed actions in the data base.

Interpretation of Behavior

The interpretation process occurs immediately after the User Monitoring step. After recording the actions in the data base, the student's UserManager instructs an Interpreter agent to decide if the behavior of the student in relation to the resource corresponds to an Acceptance or Rejection. The algorithm used by the Interpreter agent to interpret behavior has been simplified down to a comparison between the characteristics of the resource and the monitored actions. Nevertheless, the process has been implemented in a modular fashion to permit the incorporation of more complex mechanisms in the future. Finally, the result of the interpretation process (acceptance, rejection, undecided) is communicated to the UserManager agent, which completes the recording of the monitored actions using this information. Furthermore, the projected level is updated at the resource level when it is greater than the work level and the interpretation is an Acceptance.

Progress Control

Progress control is achieved through evaluations. Each time a resource is set to be displayed, the servlet ContentResource verifies that the resource level does not exceed the student work level for the learning objective (see resource level in the Content Ontology). If the resource level is greater, it instructs the Evaluation servlet to construct and show a page with a 4-question test to evaluate the level that comes immediately before the requested resource level. This level usually coincides with the work level, but not always: suppose that the student's work level is 2 and the next resource to be displayed is a level 5 resource (it is possible that no resources exist for levels 3 and 4). The system should evaluate level 4 so that the work level, if the evaluation is passed, increases to level 5 and the next resource can be displayed.

Once the evaluation is completed, the Evaluation servlet communicates the result to the corresponding UserManager, which, in the interim, has been waiting for this message and has maintained, without implementing it, the sequence which has generated the need for evaluation (see 6. Adaptive Sequencing of Resources). The UserManager verifies the

outcome of the evaluation, and depending on its success or failure, carries out two distinct processes:

- When the evaluation is successful, it updates the evaluated level and work level, and then answers the Evaluation servlet, which in turn prompts the ContentResource servlet to now show the resource.
- When the evaluation result indicates failure, the UserManager asks for the reconstruction of the sequence of resources, emphasizing the resource levels that did not pass the evaluation process and then it notifies the Evaluation servlet. This updates the ContentResource servlet which will then construct a content page using a specially ordered sequence.

Constructing the Learning Style

According to the Felder-Silverman Model [3], a student's learning style is made up of 4 dimensions, each with values between two poles, as can be seen in the following table:

Dimension	Pole (+)	Pole (-)
Perception	Sensing	Intuitive
Input	Visual	Verbal
Processing	Active	Reflective
Understanding	Sequential	Global

Table 1. the four dimensions of the Felder-Silverman Learning Styles model

In SPORAS, each dimension is represented as a whole number in the data base, with positive or negative values depending on the pole indicating tendency (positive and negative poles are defined in Table 1).

The values for the Perception, Input and Processing dimensions are calculated through an analysis of the characteristics of perception, presentation and the processing of resources viewed by the student. The Understanding Dimension in a learning style is calculated based upon an analysis of the differences between the student's progress towards the course Learning Objectives and the record of resources viewed, according to whether they were chosen or suggested by the system.

Each time a viewed resource has been interpreted, the student's UserManager requests the construction of the student learning style, which is achieved with a Modeller class agent.

Adaptive Sequencing of Resources

The student's UserManager agent manages a resource sequence for each learning objective. In addition to the list of ordered resources associated with the objective, each sequence is a structure that contains advance information about the student and an ordered list of questions.

Each time a content resource needs to be shown, the ContentResource servlet requests the learning objective sequence from the UserManager. The ContentResource servlet then generates a page displaying the first resource in the sequence and an ordered menu with links to the rest of the resources in that sequence. In this way the student can freely choose to access a specific resource instead of following the sequence suggested by the system.

ResourceProgrammer agents carry out the tasks of sequence construction and updating. Sequence construction involves obtaining information about resources and questions from the Content Ontology, as well as advance information from the data base. Updating a sequence includes recording the previous advance information and the resource order, as well as questions based upon the record of resources viewed, progress levels and the individual learning style of the student.

CONCLUSIONS

In order to allow for the future incorporation of more complex, complete or efficient mechanisms, the three system tasks requiring inference (constructing the learning style, interpretation, and constructing sequences) have been incorporated in a modular fashion into the agents that carry them out.

The main reason for this is that the system has been designed and constructed without conducting practical tests to apply the learning styles model and incorporated mechanisms have been designed using assumptions.

The use of a multiagent system is an important advantage for SPORAS. It allows for extension of agent platforms through several machines, without compromising design or implementation, and thus distributing the work load between agents in different locations.

The development of this system based upon knowledge models such as Ontologies, have permitted a glimpse into its utility as a model for reusing knowledge. Future efforts expect to implement the ontology using a standard such as IMS o SCORM. Even though a prototype of the system described in this model has been developed, evaluations assessing its function and effectiveness with students have yet to be conducted. Nevertheless, during the writing of this paper, practical tests on for its use have been undertaken with students majoring in Information Technology Engineering at the Universidad Austral de Chile. The results obtained will be used for improving the accuracy of the inference model and for identifying new information sources that could enhance the reliability of the knowledge included in the whole system processes.

GLOSSARY

JADE: Java Agent Development Framework. http://jade.tilab.com
FIPA: Foundation for Intelligent Physical Agents. http://www.fipa.org
API: Application Programming Interface.
Protégé: software for editing ontologies and knowledge bases. http://protege.stanford.edu
MySQL: database manager. http://www.mysql.com
ACL: Agent Communication Language.

REFERENCES

1. Brusilovsky P. (2003) Developing adaptive educational hypermedia systems: From design models to authoring tools. In: T. Murray, S. Blessing and S. Ainsworth (eds.): Authoring Tools for Advanced Technology Learning Environment. Dordrecht: Kluwer Academic Publishers.
2. Caire G. (2002). Application-Defined Content Languages and Ontologies, http://jade.tilab.com/doc/CLOntoSupport.pdf.
3. Felder R. M. & Silverman L. K. (1988). Learning and Teaching Styles in Engineering Education, Engr. Education, 78(7), 674 - 681
4. Bloom B. S. & Krathwohl D. R. (1956). Taxonomy of Educational Objectives: The Classification of Educational Goals, by a committee of college and university examiners. Handbook I: Cognitive Domain. New York, Longmans, Green.
5. Brusilovsky P. & Pesin L. (1994) ISIS-Tutor: An adaptive hypertext learning environment, Proc. JCKBSE'94, Japanese-CIS Symposium on knowledge-based software engineering.
6. De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D., Stash, N. (2003) AHA! The Adaptive Hypermedia Architecture. Proceedings of the ACM Hypertext Conference, Nottingham,UK