A Metadata Based Web Distance Learning Platform

Francesca Cirillo, Massimo de Santo, Andrea Cozzolino, Marco Marsella, Saverio Salerno

A Metadata Based Distance Learning Platform

Francesca Cirillo
DIIMA
University of Salerno, Italy

Massimo De Santo
DIIIE
University of Salerno, Italy

Andrea Cozzolino, Marco Marsella, Saverio Salerno
CRMPA
Salerno, Italy

***Abstract* – Distance Learning is acquiring a role that becomes more and more important with the huge diffusion of the Internet and related technologies. Consequently, the investigation for adequate architectures and platforms supporting flexible Distance Learning engines and solution is nowadays of great interests in the Scientific Community. The present paper introduces a platform based on the adoption of Metadata concepts and on the use of a "processor-like" behavior of the Web Course Delivery engine that appears to be a good solution when different and heterogeneous modules have to be integrated in the platform. The proposed system is currently under development and test in the framework of a European Esprit Project.**

***Index terms* – Metadata, Distance learning, Intelligent Tutoring Systems, Case Base Reasoning.**

I. INTRODUCTION

The United States Distance Learning Association (USDLA) defines distance learning as "the acquisition of knowledge and skills through mediated information and instruction. Distance learning encompasses all technologies and supports the pursuit of life long learning for all". [1]

During our first studies on web based distance learning, we discovered that often it is synonymous for "reading". As a result, we wondered on differences between distance learning systems and books. Our response on this topic is "interactivity" and "adaptiveness".

In typical distance learning, two different kinds of actors use the system: learners and teachers. Unfortunately they cannot interact each other without mediation due to the physical distance. For this reason, one of the main issues in such systems is the need to provide support for different kinds of Learner/Teacher interactions, for example:

1. Monitoring of Learners by the teachers through the system;
2. Automatic monitoring of Learners by the system itself;
3. Traditional teacher/learner interaction such as messages and forum areas.

Another point whose importance is typically misjudged is that learners are different one another in many ways. Usually, each learner exhibits a different level of sensibility to different didactic approaches, and, furthermore, she can react in different modes to the use of specific presentation media. Starting from her own cultural level, a given learner can be inclined to understand in a easier way some of the presented arguments or, what is sadly more often the case, she can find very difficult to understand a given subject. A truly complete system should be able to understand this kind of differences and react to them accordingly.

Finally, it can be noted that we are already in presence of a huge amount of didactical materials available through the Internet and that such an amount is increasing in an exponential way. Consequently, we decided to focus our work on the technologies needed for attending "web courses". Presently, to use a web browser, as unique tool to access the system, is the simpler, the more portable and easier solution a learner can find.

II. STATE OF THE ART

In the last decade much research has been done on distance learning and many solutions were implemented. We analyzed some platforms for distance learning and we evidenced the usual functionality that they have.

>From one side, many of the presently available systems are not flexible in the organization and presentation of didactical materials. In fact, they give to the tutor functionality for creating a course (the same course for all the learners) while they do not allow the tutor to tailor the course for a given single learner. Furthermore, once a course is established, the learner can only undergo the sequence of lessons prepared by the tutor in a passive way.

On the other side, a new learning generation of training systems is born. It is characterized by the presence of virtual training assistants that represent the system intelligent component. This objective has been pursued for more than three decades by researchers in education, psychology, and artificial intelligence. These new kinds of systems are known as Intelligent Tutoring Systems (ITS). Today, prototype and operational ITS provide practice-based guidance to support training on line.

In some cases, the intelligent components of an ITS enable computer interfaces to become more human or more "friendly" when interacting with human users. Namely, they customize lessons presentation on the basis of the learner preferences. In other cases, intelligent components are able to provide on-the-fly “ad-hoc” appropriate presentations of learning materials whose content is suitably tailored for different individual users or user groups.

The question is opened about the way in which the architecture of ITS has to be organized. The purpose of this paper is to present a system which has been designed to be a flexible platform for web courses delivery that can be used as a “core” system for the new generation of ITS systems.

III. Architectural Issues: Course Material Organization and Storage

One of the main aims of our system is to enable Tutors to organize in a unconventional manner the didactical materials. The flexible organization provided allow the tutor to create and assign courses on the basis of starting knowledge and preferences of any single student.

In the following, we describe the basic assumptions underlying the architecture of the system.

In order to obtain a flexible and open architecture, a very careful choice of used data structures is needed. We based all our data structures on IEEE LTSC Learning Object Metadata (LOM) standard [4]. Using LOM standard for Metadata, tutors can index the didactical material. This makes it possible for all modules to access and use in a flexible way information gathered and stored by the system.

Metadata is information about an object, be it physical or digital and its main goal is to locate in efficient and effective way resources over a system or a computer network. Metadata is “data about data”: a concept that can be easily understood using a “supermarket metaphor”. In a supermarket, things inside boxes, like butter or shampoo, are the data; the information on the label is the metadata.

Metadata is organised into categories, or fields. Each field represents a characteristic of the learning resource, for example, the resource's title or a summary abstract. Each field has a value and some fields may have multiple values. For example, the abstract field may have only one value, a brief summary of the material, or the field may have multiple values, if the same abstract is provided in several different languages.

Let now us introduce the following definitions about Learning Object, Curriculum, Learning Goal and let us describe how Curricula are related to Learning Goals.

A *Learning Object* can be defined as any kind of web-deliverable document, for example:

1. A single lesson represented by an hypertext (an HTML page or a little sequence of them);
2. A simulation (an HTML page containing a Java interactive applet);
3. A virtual world (a VRML file);
4. A test (an HTML page with an evaluating Java applet).

A *Learning Object* Metadata Standard [4][7][8] defines the minimal set of properties needed to allow these objects to be managed, located, and evaluated. The standard will accommodate the ability for locally extending the basic properties. Relevant properties of learning objects include type of object, author, owner, terms of distribution, and format. Where applicable, *Learning Object Metadata* may also include pedagogical properties, such as: teaching or interaction style, grade level, mastery level and prerequisites.

A *Curriculum* is an ordered list of Learning Objects, which can be used for providing a specific student with all the knowledge related to a specific Course.

A *Learning Goal* is a set of key concepts to be learned for successfully completing a specific Course. The concepts and their relations are represented using a Conceptual Graph [5].

On the base of this definition, it is clear that Learning Goals indicate which concepts a student has to learn and the Curriculum specifies how these concepts have to be learnt.

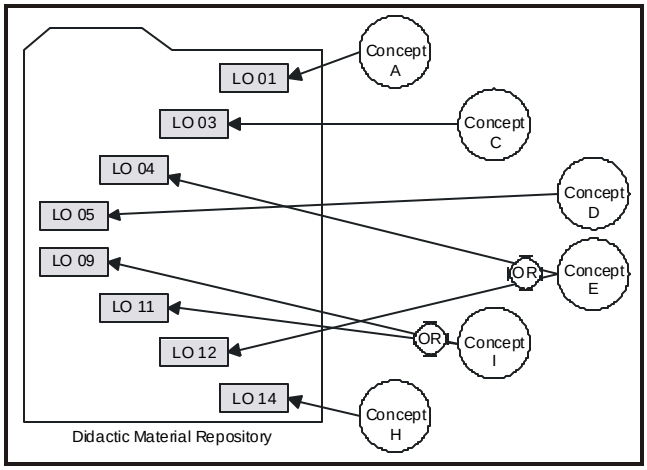

Figure 1 - Relationship between Concepts and Learning Objects (LO).

It is important to note that a given Curriculum is related to only one specific student. Students cannot share their Curricula because different students require different ways to learn about the same Learning Goals depending on their knowledge state and Preferences.

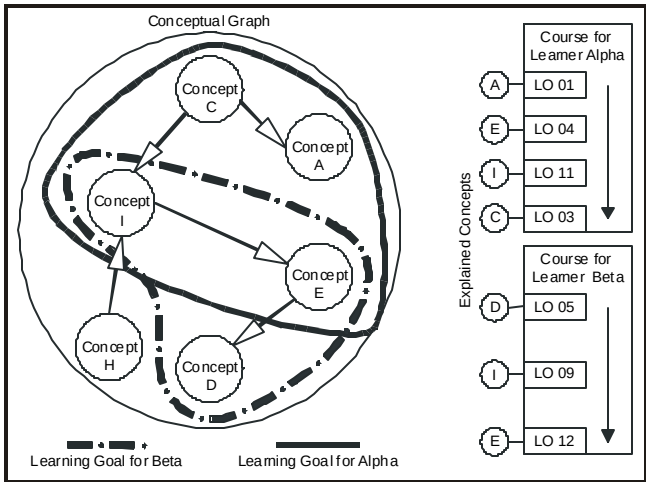

Figure 2 – Relationship among Learning Objects, Learning Goals and Curricula.

The system stores all information related to a single student, for example: the curriculum, the knowledge state, the time he spent on a given lesson, the amount of knowledge he learned from the lesson (for example, using test based evaluations) and his preferences (i.e., to which kind of resources the student has shown to be more receptive). All this information can be used by an intelligent tutoring system, which generates the best curriculum for a specific student.

The web course delivery engine is based on an extendible command interpreter and on a predefined set of commands that can be started by the student through the GUI interface (HTML pages). The engine works on learning domain objects in the same way a CPU works on a program flow instructions. Each command is constituted by a well-arranged sequence of queries to the database, whose final result is a new HTML page. A mechanism named "Extender" has been inserted into the engine in order to manage particular commands outside typical sequence of queries.

One of the key concepts of our system is that all the essential elements of a web learning system (for example, didactical materials and student activity) can be managed through the defined entities in a uniform way.

This allows tutors:

1. To perform the personalization of the course to classroom level and/or to learner level;
2. To build a learner oriented web course;
3. To control the learners progresses and understand their difficulties;
4. To arrange sequences of Learning Objects in a specific curriculum on the base of the didactical methodology adopted by the tutor himself.

Furthermore, the use of defined entities through a standard implementation based on Metadata allows easy interfacing with different Intelligent Tutoring Systems.

IV. Architectural Issues: The Core Interpreter

Our web course delivery platform is based on extendible command interpreter. A predefined (and context sensitive) set of commands is offered to the student through HTML pages. When a command is submitted, our interpreter (which can be thought of as an interface engine) retrieves instructions and data from a repository and generates a response page.

The core of our architecture is a simple cycle into a Java Servlet™ that works on HTML template pages and data retrieved from a database. Unfortunately, in some cases this cycle is unable to allow the realization of all the operations needed to perform the requested task. For this reason we introduced a mechanism to "extend" the cycle. This approach allowed us to construct a flexible and extendible interface in a very short time.

To better understand how engine works, the reader can think to a CPU working on program flow instructions. It fetches from a "repository" the instructions (micro-code) needed to execute the requested machine language command. Then it applies those instructions on the input data. Our system applies this cycle in the same manner on learning domain objects.

The main cycle can be simply described as follows.

Each time a users requests an operation:

1. The system gathers the data inserted by the user into the current page through HTML/HTTP form interface;
2. A "command" identifier (always present) is extracted;
3. Instructions related to all needed operation are retrieved from database themselves. The system collects them using command identifiers as index to access the command database;
4. If needed, the system puts the data collected into the database;
5. Data needed to construct the response page are selected from database;
6. Response data are embedded into the template page by a textual substitution of predisposed tags using collected data.

Note that instructions collected into step 3 are both the SQL queries to use into steps 4 and 5 and the template page to use in step 6.

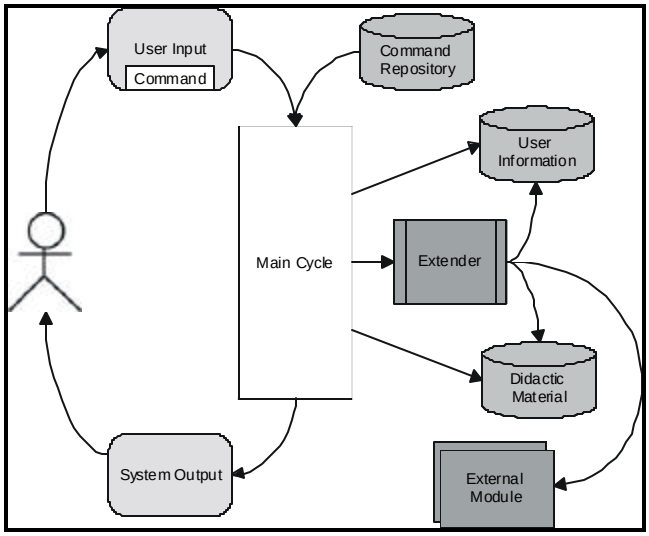

Figure 3 – The main "cycle" of the system and the "extender" mechanism.

An *extender* is a Java™ interface that we use to "extend" the main cycle. Between each step of the cycle and the following one, a function of the *extender* can be invoked. *Extender*s are usually developed to implement "commands" that interacts with external data sources, e.g. an external intelligent tutoring system. Also the information on how and when to call *extenders* are stored into the *command database*.

As an example of the use of Extenders mechanism, let us consider the case of the *milestone* mechanism. A *milestone* is a marker that a tutor or an external module can insert into a learner curriculum. The goal is to set *flags* into course fruition where an external module must be invoked.
When a *milestone* is reached, the *extender* mechanism is used for allowing the Interpreter to call and external module which is in charge for changing the students "state".

V. CONCLUSIONS

We implemented an innovative system for distance learning over the web. Using this platform we was able to reach three important objectives: the possibility to organize easily didactical material tailored to specific learning goals (knowledge that must be transferred to students) and arranging it into several curricula; the possibility to attend a course using a simple web browser; and, finally, the possibility to control in a fine grained manner the student knowledge progress. The system is also designed to be architecturally simple, easy to extend, to integrate with other modules and to customize. This allows fully exploiting system capabilities.

This system was designed and developed as base platform for InTraSys (Intelligent Training System in Technical Assistance), in the framework of the European ESPRIT project N. 29.082.

On the top of the described core structure, four different modules have been integrated: an intelligent tutoring system [2], two different simulation engines [3] and a case based reasoning system. Among these modules, InTraSys interface platform and Intelligent tutoring system are particularly integrated but, according to our experience, it will be easy to integrate quite each kind of additional module into the system.

In particular, our system offers a privileged interface to the Intelligent tutoring system. Given that the ITS described in [2] is based on a network of intelligent agents, we adopted the following strategy to integrate the two components. We simply added a dumb agent, the *spooler* agent acting like a bridge and distributing message between the platform and the tutoring system.

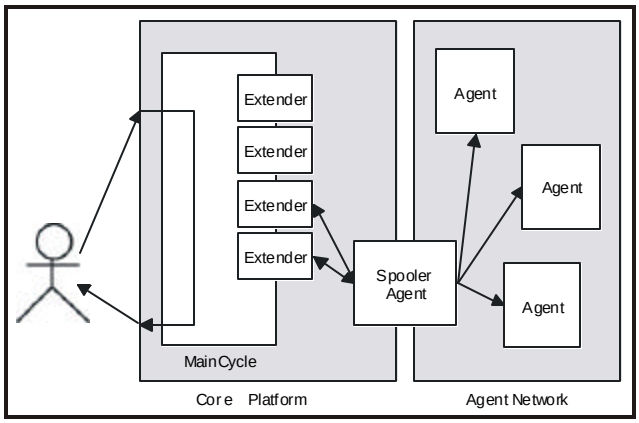

Figure 4 – Integration of the Core Engine with the ITS

Using *extenders* and *spooler* we realised also the logging mechanism. In order to support intelligent tutoring system we need a detailed log of learner activity, to better evaluate their response to the system. *Extenders* mark to the *spooler* each meaningful event and then the *spooler* logs it and, eventually, spawns it to other agents.

The system is now available in a prototype version and it is undergoing an evaluation phase. During this phase, we will teach the same arguments to two groups of students with homogeneous knowledge background. One of these groups will use our system and the other one will use class based traditional teaching methods, in order to evaluate the effective impact of the our proposed distance learning platforms and strategies on the student learning curve.

REFERENCES

[1] United States Distance Learning Association. (http://www.usdla.org), March 2000

[2] N. Capuano, M.Marsella, S. Salerno, "*ABITS: An Agent Based Intelligent Tutoring System for Distance Learning*", Proceeding of ITS 2000, Springer-Verlag, 2000

[3] A. Cavallone, M. Gaeta, M. Marsella, F. Orciuoli, "*Distance Learning Models and Web Simulation*", Proceeding of International Conference on Web-based Modeling & Simulation, San Diego-California 2000

[4] W. Hodgins et al., "*Learning Object Metadata, Working Draft Document 2.5*", IEEE Learning Technology Standards Committee (LTSC), 1999. (http://ltsc.ieee.org/).

[5] J. Sowa, *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, 1984.

[6] A. Cavallone, C. D'Apice, M. Marsella, F. Orciuoli, S. Salerno, "*A web-based simulation environment for technical staff tutoring in computer network*" proceeding of SCS Euromedia, Anversa-Belgio, 2000.

[7] Hamalainen M., Whinston A.B. and Vishik S., "*Electronic Markets for Learning: Edication Brokerages on the Internet*", Communications of the ACM, Vol.39, N.6, 1996.

[8] AA.VV., "*IMS Metadata Specifications*", EDUCOM IMS project, 1999.