

# The EUME Project: Modelling and Design of an Intelligent Learning Management System

E. Sánchez, M. Lama, R. Amorim, A. Riera, J. A. Vila and S. Barro

*Intelligent Systems Group  
Departamento de Electrónica y Computación. Facultad de Físicas  
Universidad de Santiago de Compostela. Spain*

**www.eume.net**

**Abstract.** The EUME project is intended to develop an Intelligent Learning Management System (ILMS) with the aim to improve the quality of traditional teaching strategies as well as to facilitate the implementation of new learning methodologies. At this project stage, we have developed a knowledge model of the educational domain, designed a component-based software architecture, and implemented the second cycle of the construction plan. In this paper an overview of the main results achieved so far is presented.

## Introduction

In the field of computer-based learning, many types of systems have been proposed in order to support different educational methodologies. The most popular paradigms [5] are: Computer Aided Instruction (CAI), Instructional Tutoring Systems (ITSs) and Computer Supported Collaborative Learning (CSCL). CAI is the older one and is intended to support traditional behaviourism learning. Intelligent Tutoring Systems (ITSs) are computer-based systems focus on providing personalized learning tools; they encapsulate instructional models that specify what to teach as well as teaching strategies that specify how to teach [9]. A natural extension of ITS, is the Authoring Intelligent Tutoring Systems (AITs). Their goal is the design and development of ITSs by using popular features from commercial authoring systems, originally intended to help on writing and developing hypertext and multimedia applications [7]. Computer Supported Collaborative Learning (CSCL) systems, finally, are oriented to support collaborative learning experience in which two or more agents engage the goal of constructing knowledge based on group discussion and decision-making processes.

When focusing on in-classroom learning scenarios, there exist problems that are common to any learning process and are not solved by any of the aforementioned paradigms. For instance: the resource management problem. In a classroom context, the term *resource* provides an abstraction to denote other learning entities like hardware devices, software applications, educational materials and course information. The resource management problem can then be stated as how to handle every available resource in a unified, simple and appropriate manner. Learning Management Systems (LMS) have been traditionally used to cope with this problem.

In this work we present the EUME system, which aims to solve the resource management problem in the context of collaborative learning. The first step has consisted on the identification of common tasks performed by human agents (teachers, students, etc.) involved in collaborative learning processes, and the development of task models [6] using the CommonKADS methodology [14]. The second step has concerned with the design of a software architecture [13] and a distributed handheld-based system [11,13] to mainly satisfy both resource management and collaborative learning requirements.

The paper is organized as follows: section 1 overviews the collaborative learning task models; section 2 explains how the task models were used as the first step to design the software architecture; section 3 introduces the current software architecture; section 4 describes the current implementation; section 4, finally, provides a brief discussion and anticipates future work.

## 1. Task models

### 1.1 The teaching task model

In collaborative learning the teaching task can be considered as a planning problem in which a set of activities to be performed by the students within a certain period of time must be specified. This problem can be approached [6] using a general methodology known as *propose-verify-criticise-modify* (PCVM):

- *Propose*. Based on course information, the teacher defines pedagogical objectives and activities to be developed by the students. Furthermore, the teacher would need to request educational resources and explain them in the classroom as a matter of a planned activity.
- *Verify*. The teacher needs to keep track both student and group performance in the classroom. It can be accomplished by means of observations annotated in the classroom.
- *Criticise*. Based on recorded observations and evaluation forms, the teacher analyses the educational activity performance for each group and student.
- *Modify*. After evaluation, the teacher could decide either (1) to change the pedagogical material used by the students; and/or (2) to propose new teaching initiatives.

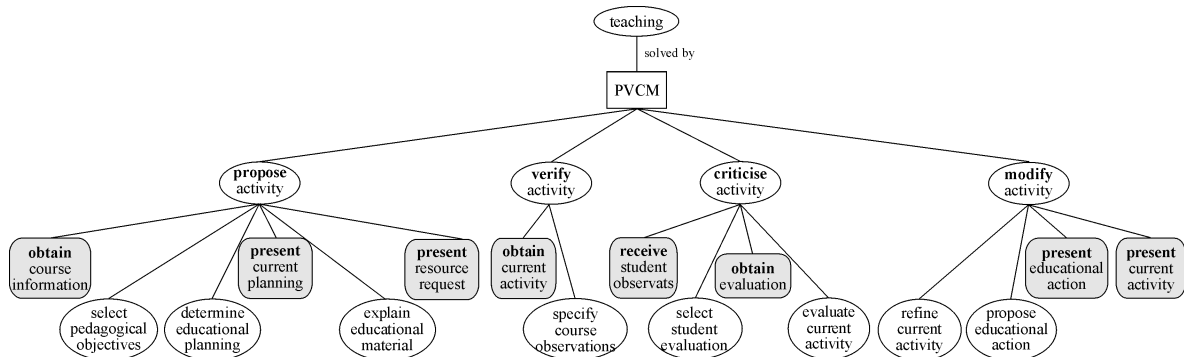


Figure 1. The teaching task model.

Figure 1 shows the breakdown of the general tasks (propose, verify, criticise and modify) into domain subtasks (white ovals) as well as transfer functions (grey rectangles) [2], which

represents interactions between the teacher and the other agents (students and educational system) in the environment. For example, the teacher can either send information to other agents (*present*) or request information from them (*obtain* and *receive*).

### 1.2 The EUME system task model

The task breakdown is based on the transfer functions identified in the teacher task model. These tasks can be initially classified as follows:

- *Manage information request.* It concerns with the manipulation of educational information: student data, list of pedagogical objectives, pre-designed activity templates, educational resources.
- *Manage agent interaction.* This kind of tasks include services to solve teacher-student and student-student interactions: student requesting for additional material or resources in order to work towards the proposed activity, teacher requesting student answers to activity evaluation forms, and teacher reporting changes in the proposed activities to the involved students.
- *Manage resource request.* This is the most important task. It comprises the management of agent calendars, hardware devices as well as application resources.

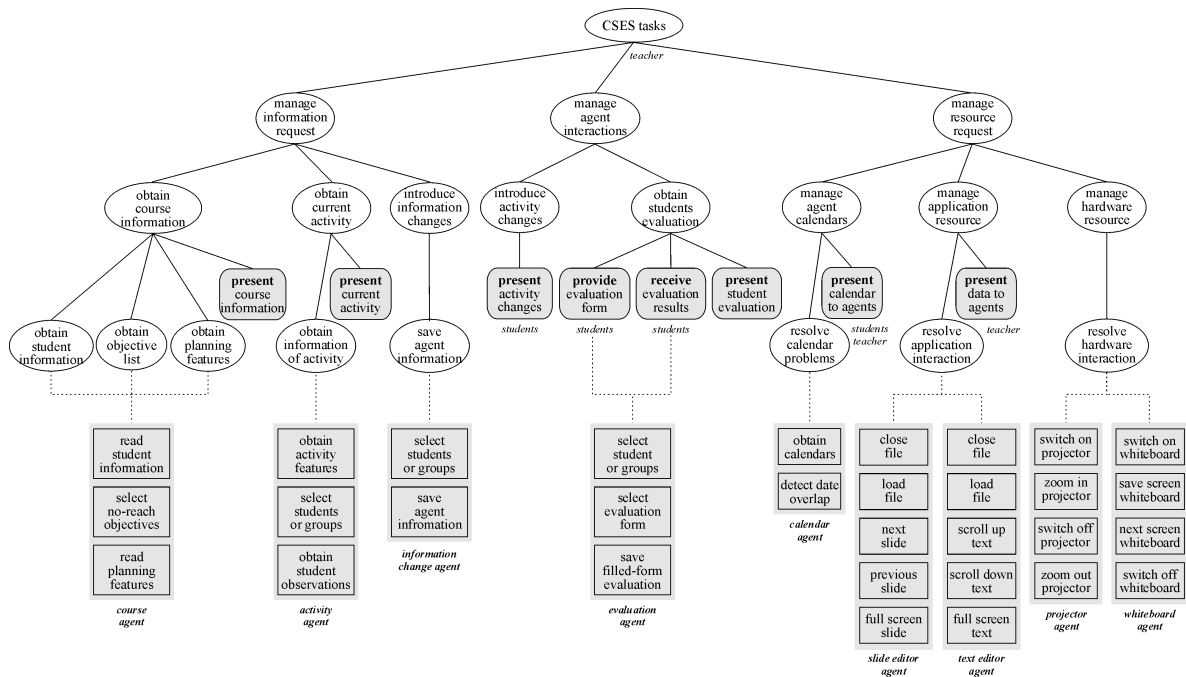


Figure 2. The EUME system task model.

## 2. From the system task model to software agents

The EUME system task model represents most of the functionality that the final system should show in order to satisfy user (professor and students) needs. Therefore we have analysed the subtasks located at the bottom of the task tree in figure 2 and identified the number of services related to each subtask. Each group of services can be further associated to a single software agent, thus defining its functionality. At the end, every general system task is performed by a set of agents: the Manage Information task will be performed by

Course, Activity and Information Change Agents; the Manage Agent Interaction task by an Evaluation Agent; and finally the Manage Resource task by Calendar, Slide Editor, Text Editor, Projector and Whiteboard Agents. Additionally, an agent has different availability conditions depending if it is going to be used in classroom, out of classroom, or in both locations.

The identification of software agents from the system task model has been the first step in the design phase. As we will see shortly, they have constituted the basic building blocks of the key components of our software architecture.

### 3. The software architecture

The software architecture for the EUME project aims to satisfy the requirements that naturally follow from the two main project goals: support traditional teaching methodologies and support the adoption of advanced learning techniques. For the first goal, we considered the following functional requirements: resource management services (including remote control and information capturing services) and authoring course-planning facilities. As a resource we understand every application, device or information related with the learning activity. For the second goal, the additional requirements concern with student/group/teacher communication and collaborative work planning features to be added to the authoring course-planning facility. The current architecture is an enhanced version of that discussed in Sánchez et al. [13].

The architectural style corresponds to a multi-tiered architecture (figure 3) in which every tier is in charge of some specific computation. In what follows we introduce the tiers and the related components and connectors [15]:

- *Client Tier*. It contains the components that interact with the user: POC/I (Professor Out-of-Classroom Interface) allows the professor to obtain course information, to plan educational activities, to design educational resources and in general every authoring-oriented activity; PIC/I (Professor In-Classroom Interface) helps the professor to request needed resources and to communicate with students; A/I, SIC/I and GIC/I (Administrator, Student In-Classroom, and Group In-Classroom Interface) are client interfaces for administrator, student and group, respectively.
- *Interface-oriented Communication Service Tier*. It handles communication between clients. As components we have the ICS/LS (Interface-oriented Communication Services Lookup Service), which provides automatic registering for every user and support for user interface discovery.
- *Resource Management Service Tier*. It contains components offering the services identified in figure 2. RMS/MED (Resource Management Services MEDIator) routes user requests to the appropriate RMS; P/RMS (Professor Resource Management Services) is a key part of our architecture and handles all services available to the professor in order to facilitate every teaching task. Figure 2 describes the agents and corresponding services that are integrated within the P/RMS; A/RMS, S/RMS and G/RMS (Administrator, Student, and Group Resource Management Services) package the services required for system administration, student and group support.
- *Resource Tier*. It packages controllers of (1) applications, (2) hardware devices, and (3) data access. NS (Name Server) provides resource localization; DE/R and APP/R (DEvice and APPlication Resource) represent the controller of an application and

devices available in the classroom, respectively. Finally, DB/R (DataBase Resource) stands for a database manager.

For explanatory purposes, we have considered two general scenarios to describe how the architecture would therefore work:

1. *Resource management.* If a client pretends to request an operation either on a device or an application, the PIC/I components issues a request message that is processed through a *message passing connector* (I-RMS/MED), and subsequently routed by the RMS/MED component to the appropriate RMS. The RMS triggers a suitable internal agent to handle the request that performs the following operations: (1) requests to the NS the location of a convenient device/application controller location, and (2) forwards the user request to it. To perform these operations we need a *procedure call connector* (RMS-NS) to perform the controller look-up operation as well as a *synchronous message passing connector* (RMS-DE/R-APP/R) to handle request-notification interactions between RMS and both DE/R and APP/R.
2. *Human agent communication.* Interaction between human agents (students and professor) implies the use of the Interface-oriented Communication Service Tier. First, when the user logs into the system, a procedure call connector (I-ICS/LS)

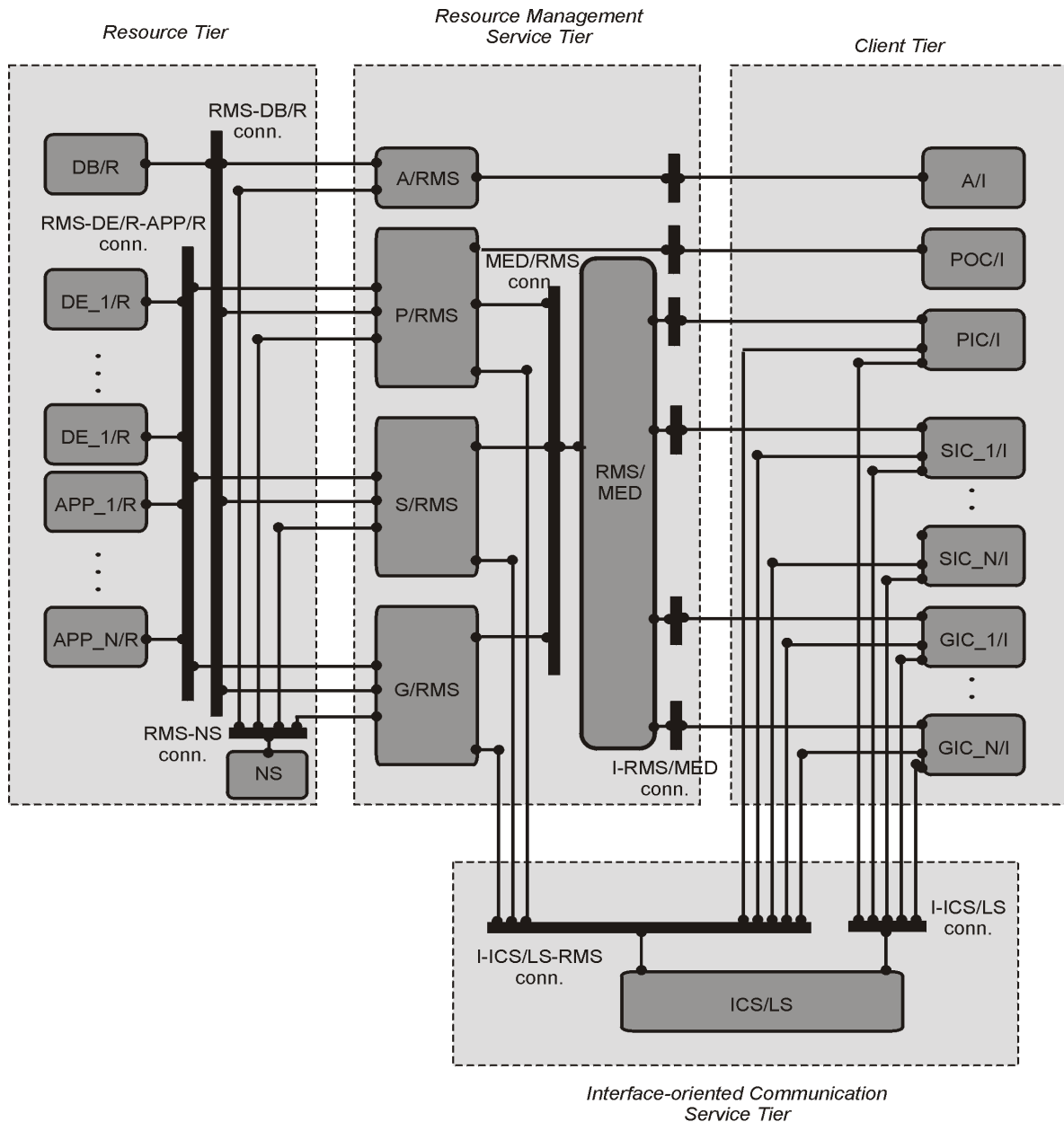


Figure 3. The software architecture.

transparently handles the discovery of other connected users. After that, all message handling is managed by the ICS/LS component. If a user (typically the professor) needs to send any educational resource to another user (typically an student), the sender's RMS is in charge of accessing the requested resource by means of a dedicated I-ICS/RMS connector.

#### 4. Implementation

At the current cycle of the construction plan we have developed most of the components concern with resource management services. This means: the DE/R and APP/R controllers, a basic P/RMS, the RMS/MED, and the PIC/I. The components were implemented in Java and the connectors in Java RMI. For message passing we have used XML-based messages.

After testing, we have deployed the DE/R, P/RMS and RMS/MED in a Classroom Server, running *Windows XP*, and the PIC/I in a *Compaq iPaq* PDA, running Linux (the *Familiar* distribution). The system includes educational hardware devices, such as an LCD projector and a whiteboard capturing system. In the classroom there is a wireless *Ethernet* LAN with an access point to communicate the PDA with the server.

The main services of the resource controllers are summarized in table 1: the LCD Projector controller offers services to switch on/off the device and to zoom in/out the image;

Table 1. Resources: type and associated services.

<i>Resource Name</i>	<i>Controller Type</i>	<i>Main services</i>
LCD Projector	DE/R	Switch_on Zoom_in/Zoom_out Switch_off
Whiteboard Manager	DE/R	Switch_on Open_file Show_next_screen Save_screen Switch_off
Text Editor	APP/R	Open_file Show_full_page Scroll_down Scroll_up Go_Top Go_Bottom Close_file
Slide Editor	APP/R	Open_file Show_full_slide Next_slide Previous_slide First_slide Last_slide Close_file
Internet Browser	APP/R	Open_URL Show_full_screen Scroll_down Scroll_up Go_Top Go_Bottom Select_hyperlink

the Whiteboard Manager controller allows to handle previously saved screens and capture new ones; the Text and Slide Editor controllers provide access to widely used text and slide operations, respectively; and the Internet Browser allows to navigate among websites. These services can be accessed through a common communication service called *message handler* that parses the incoming request and passes the control flow to the appropriate service.

Figure 4 illustrates the look-and-feel of the graphical user interface in the PDA. We have emphasized system usability, what means: (1) providing a task-oriented interface, and (2) using large buttons to facilitate easy touch control. The tasks are grouped related to each device or application resource, and a natural relationship exists between these tasks and the services offered by the resource tier (see table 1).

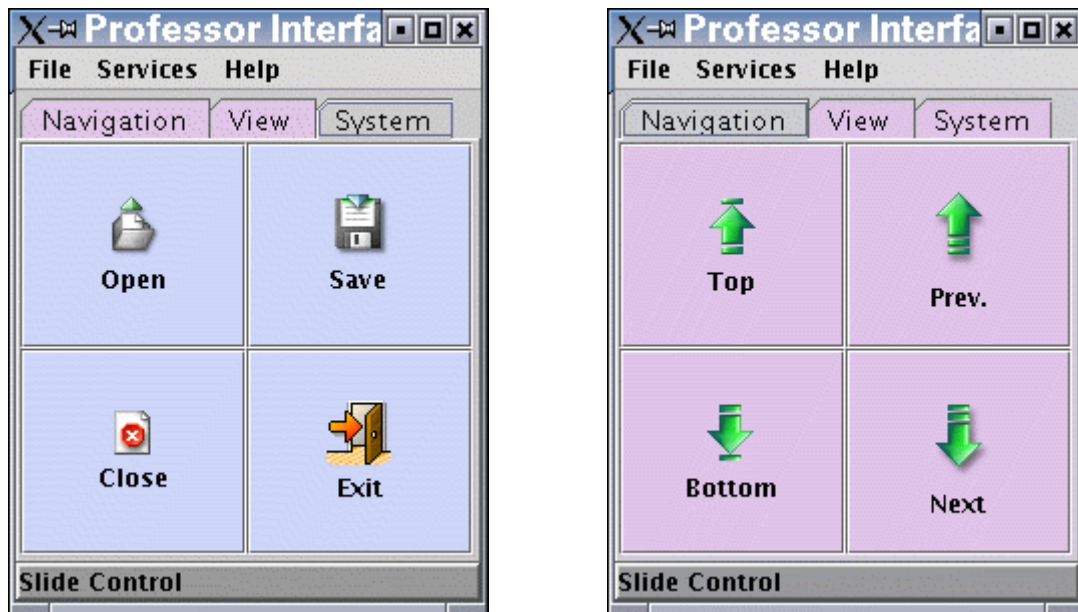


Figure 4. Task-oriented PDA interfaces. The System window provides access to common application tasks (left). The Navigation window shows popular navigation commands for slide control (right).

## 5. Discussion

The proposed software architecture is designed to support resource management (DE/R and DB/R controllers in the Resource Tier; RMS agent-based components in the Resource Management Service Tier), and collaborative learning (Interface-oriented Communication Service Tier). Moreover, the architecture is ready for evolution: distance learning can be provided by means of a new web application tier to access the database through the Internet; and personalized tutoring can be achieved through an extension of the user interface.

Our system include features typically offered by Information Capturing Systems, like the e-class project (formerly Classroom2000), which are designed to capture the most information generated in the classroom as possible [1,4]. Moreover, our interest on mobile devices as control mechanisms is shared with other projects that use them either to support collaborative learning [3,12], or to control applications and educational devices in the classroom [8,10].

Finally, we believe that new models are needed to synthesise in a single framework the description of learning methodologies, teaching activities and cognitive processes. The EUME project is working towards this direction. On one side, we are developing [6] the first knowledge model based on a widely used modelling methodology: CommonKADS. On the other side, the modelling study is being completed with a domain ontology constructed from standards like EML and LOM. Furthermore, this low level ontology will serve as the basis for database design, component communication and future interaction with external systems.

## Acknowledgements

Authors wish to acknowledge support from the Xunta de Galicia through grant PGIDIT02TIC20601PR.

## References

- [1] G.D. Abowd. *Classroom 2000: An experiment with the instrumentation of a living educational environment*. IBM Systems Journal. 38(4):508-530, 1999.
- [2] J. Akkermans, R. Gustavsson, and F. Yagger. *An integrated structured analysis approach to intelligent agent communication*. In J. Cuena (editor). Proceedings of the IFIP 1998 World Computer Congress, IT&KNOWS Conference, Chapman & Hall, 1998.
- [3] A. Danesh, K. Inkpen, F. Lau, K. Shu and K. Booth. *Geney: Designing a collaborative activity for the Palm handheld computer*. Proceedings of CHI, Conference on Human Factors in Computing Systems. Seattle, USA, April 2001.
- [4] J. Flachsbart, D. Franklin, K. Hammond. *Improving human computer interaction in a classroom environment using computer vision*. Conference on Intelligent User Interface. pp. 86-93, 2000.
- [5] T. Koschmann. *CSCL, Theory and Practice of an emerging paradigm*. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 1-23, 1996.
- [6] M. Lama, R. Amorim, E. Sánchez, A. Riera, J. Vila, S. Barro, B. Cebreiro and C. Fernández-Morante. *A task model for a management resource system integrated in a cooperative learning environment*. Proceedings of the International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2002). Crema, Italy, 2002.
- [7] T. Murray. *Authoring Intelligent Tutoring Systems: An analysis of the state of the art*. International J. of Artificial Intelligence in Education 10: 98-129, 1999.
- [8] B.A. Myers. *Using handhelds and PCs together*. Communications of the ACM, 44(11):34-41, 2001.
- [9] S. Ohlsson. *Some Principles of Intelligent Tutoring*. In Lawler & Yazdani (Eds.), Artificial Intelligence and Education, Ablex: Norwood, NJ, 1: 203-238, 1987.
- [10] J. Rekimoto. *A multiple-device approach for supporting whiteboard-based interactions*. Proceedings of CHI'98, Proceedings of CHI, Conference on Human Factors in Computing Systems. Los Angeles, USA, 1998.
- [11] A. Riera, J. Vila and S. Barro. *A PDA classroom computer system*. In C. Montgomerie and J. Viteli (editors). Proceedings of the World Conference on Educational Multimedia, Hipermedia & Telecommunications (ED-MEDIA 2001), 2001.
- [12] J. Roschelle and R. Pea. *A walk on the WILD side: How wireless handhelds change CSCL*. Proceedings of Computer Support for Collaborative Learning (CSCL 2002), Colorado, USA, 2002.
- [13] E. Sánchez., M. Lama, R. Amorim, A. Riera, J. Vila, S. Barro. *A multi-tiered agent-based architecture for a cooperative learning environment*. Proceedings of IEEE Euromicro Conference on Parallel and Distributed Processing (PDP 2003). Genoa, Italy, 2003.
- [14] G. Schreiber, R. de Hoog, H. Akkermans, A. Anjewierden, N. Shadbolt and W. van de Velde. *Knowledge Engineering and Management*, MIT Press. 2000.
- [15] M. Shaw and D. Garlan. *Software architecture*. Prentice Hall. 1996.