A Generic Platform for the Systematic Construction of Knowledge-based Collaborative Learning Applications

Santi Caballé, Thanasis Daradoumis, Fatos Xhafa

Chapter XII
A Generic Platform for the Systematic Construction of Knowledge-Based Collaborative Learning Applications

Santi Caballé, Thanasis Daradoumis
Open University of Catalonia, Spain

Fatos Xhafa
Polytechnic University of Catalonia, Spain

ABSTRACT

This study aims to explore the importance of efficient management of event information generated from group activity in collaborative learning practices for its further use in extracting and providing knowledge on interaction behavior. The essential issue here is how to design a platform that can be used for real, long-term, complex collaborative problem-solving situations and which enables the instructor to both analyze group interaction effectively and provide an adequate support when needed. The achievement of this task first involves the design of a conceptual model that structures and classifies the information generated in a collaborative learning application at several levels of description. This conceptual model is then translated into a computational model that not only allows the efficient management of the knowledge produced by the individual and group activity but also the possibility of exploiting this knowledge further as a meta-cognitive tool for real-time coaching and regulating the collaborative learning process. The computational model becomes the central issue in this contribution while the conceptual model is briefly introduced.

INTRODUCTION

Computer supported collaborative learning (CSCL) is an emerging paradigm (Koschmann, 1996) for research in educational technology that focuses on the use of information and communications technology (ICT) as a mediation tool within collaborative methods of learning. When designing and implementing environments that support online collaborative learning, several issues must be taken into account in order to ensure full support to the online learning activity. One such issue is the representation and analysis of group activity interaction.

Interaction analysis is a core function for the support of coaching and evaluation in online collaborative learning environments. It relies on information captured from the actions performed by the participants during the collaborative process (Dillenbourg, 1999; Martínez, de la Fuente, & Dimitriadis, 2003). The efficient embedding of this information and of the extracted knowledge into applications sets the basis for enhancing monitoring, awareness (Gutwin, Stark, & Greenberg, 1995) and feedback (Zumbach, Hillers, & Reimann, 2003) to achieve a successful learning process in collaborative environments. Therefore, the success of CSCL applications depends to a great extent on the capability of such applications to embed information and knowledge of group activity and use it to achieve a more effective group monitoring (Gutwin, Stark, & Greenberg, 1995).

CSCL applications are characterized by a high degree of user-user and user-system interaction and hence are generating a huge amount of event information. This information can be conveniently collected and automatically processed by computers as a data source to extract relevant knowledge of the collaboration. Note that in this context information refers to the event data generated by the learning group and knowledge refers to the result of the treatment of this information. Knowledge is acquired by means of analysis techniques and interpretations that will be presented to the same group that generated it.

As a result, the event information management is the cornerstone in this context, aiming at achieving three main goals: (1) Provide an analysis of the group's information by obtaining and classifying the necessary information gathered from the collaborative activity into three essential types of categories (Daradoumis, Martínez, & Xhafa, 2006), namely the *outcome of collaboration* (the members' contributing behavior to the task), the *functioning of the group* (the management and organizational processes underlying the collaborative learning activities, such as participation behavior, role playing, etc.), and *individual and group scaffolding* (social support and task- or group functioning-oriented help); (2) Given that the large amount of information generated during online group activity may need much time to be processed, an effective way to collect, analyze and present this information is required; (3) Efficiently embed the information and knowledge obtained into CSCL applications so as to both facilitate tutors to monitor the learning activity and constantly provide group members with as much awareness and feedback as possible.

Achieving a clear and well-structured conceptual model constitutes a principled manner for the design of a computational model that implements the process of embedding information and knowledge into a CSCL application. Indeed, the structuring and classification of the event information into specific collaborative processes can contribute and facilitate the building of a portable, general and reusable collaborative learning ontology for the representation, learning and inference of knowledge about each collaborative process. This allows the design of effective computational models that reflect as accurately as possible task performance, individual and group behavior, interaction dynamics, members' relationships and group support.

To this end, a generic, robust, reusable platform is provided for the systematic construction

of CSCL applications endowed with enriched capabilities for providing more efficient knowledge management and scaffolding. This platform, called the Collaborative Learning Purpose Library (CLPL) (Caballé et al., 2004), acts as a computational model of collaborative learning interaction and can be used to embed information and knowledge into collaborative learning applications in an efficient manner.

This chapter is organized as follows: We first present an overview of other existing collaborative learning platforms which leads us to identify and discuss the main problems to be faced in the construction of our generic platform. Then, we enter the description of design principles that conducted the development of the CLPL and its architecture, which will be described in detail. Finally, the construction of a real application, a structured discussion forum, is discussed to validate the possibilities offered by our platform as regards data analysis and management. We conclude highlighting the main points and remarks covering this chapter and pointing out ongoing and further work.

DISCUSSION ON EXISTING PLATFORMS FOR COLLABORATIVE LEARNING APPLICATIONS

Generic platforms, frameworks and components are normally developed for the construction of complex software systems through the reuse technique (Czarnecki, 2005). This approach has been successfully applied to different domains thus providing applications of increased quality reducing both cost and development time. For this reason, it has attracted the attention of developers from the collaborative learning domain, mainly from Web-based distributed development from which the most popular platforms have arisen.

With the increasing interest in the World Wide Web over the last decade, several proposals have been made for the development of Web-based platforms for collaborative learning applications. These platforms are mainly focused on standardizing learning metadata schemes, course structures, and software interfaces to provide interoperability between applications and learning resources (Anido-Rifón et al., 2001).

We will now briefly review some of these proposals and point to several important considerations related to the collaborative learning domain. Particular attention will be paid to those aspects related to the information and knowledge embedding which have been, to the best of our knowledge, little investigated.

Bacelo et al. (2002), though focusing on collaborative learning environments in general, introduce an interesting approach to component-based architecture to support collaborative application designs. In this study, the authors concentrate on how reuse techniques (components and object-oriented framework) can be applied to collaborative learning domains and make some preliminary proposals for the component-based architecture. At the level of analysis there are some interesting ideas. However, they present certain limitations as only communication, cooperation, and information sharing aspects are considered. Aspects such as awareness and knowledge management need to be taken into account and analyzed at a deeper level.

Other approaches describe an initial attempt to remediate this deficiency. Thus, Anido-Rifón et al. (2001) propose a three-layered component-based framework focused on Web-based interactive and collaborative educational applications. In this approach, events produced by a given component are handled and delivered to remote components in the same application. Thus, an auditing tool registers the actions that each particular user executes (e.g., which application resources are used, when they are used, for how long, and by whom). Another component allows the sharing and distribution of events performed on the shared application's user interface, where users' actions will be forwarded to every other user in the group. Although the

proposed framework supports most scenarios with their particularities, it again fails to focus neither on processing and analyzing the event information from the user actions nor on how to present this information so as to provide users' fundamental needs for groupware environments such as dynamic support to group awareness, specific components for awareness and feedback management.

At this point, one the one hand, we state the importance of the information management as a cornerstone of the collaborative learning environments. The aim is to model different aspects of interaction and thus at helping all the actors involved understand the outcomes of the synchronous and asynchronous collaborative learning process. Indeed, the specification of high-level collaborative learning processes constitutes the first step towards the classification of the many different variables that characterize collaborative interaction. In addition, this allows the identification and measurement of these variables in terms of the user and system specific events (or actions). This conceptualization enables the construction of a computational model to gather information in a structured manner and, consequently, to provide an easier and more efficient further processing and analysis of this information through different techniques (such as statistical and data mining, social network analysis etc.). The ultimate aim is to interpret the analysis results and extract, reveal and provide the actors with valuable knowledge for each of the three high-level collaborative learning processes.

On the other hand, in distributed systems, independence from any platform and interoperability between different technologies are crucial issues. In addition, in the domain of groupware applications, the interoperability between different applications to support collaborative work is fundamental. Many popular Web-based platforms have turned up in the market for the construction of distributed collaborative applications, such as WebCT[1], PhpBB[2], and Moodle[3]. However, even though they support many aspects of collaborative applications, they do not fully support interoperability thus making the applications dependent from the programming language, underlying infrastructure, and so on. It is worth mentioning here some approaches such as Amin, Nijsure, and von Laszevski (2002), and Bote-Lorenzo, Dimitriadis, and Gómez-Sánchez (2003), which point to the use of great-scale, distributed computing environments in the development of components for collaborative learning domains. These approaches aim at meeting many important needs of collaborative learning applications, such as group activity data analysis and management, in a highly effective manner. In this context, interoperability becomes essential to meet these demanding requirements. For instance, Grid computing (Foster & Kesselman, 1998) offers high-throughput and data-intensive computing, which greatly facilitate the process of embedding information and knowledge into these applications making it possible to provide users with constant real-time awareness and feedback.

To sum up, the problems to be faced for knowledge-based collaborative learning applications are: (1) How to efficiently process the large amount of information collected during the group activity in order to facilitate its later analysis and make the extracted knowledge available to the participants even in real time; (2) How information should be analyzed and what kind of knowledge should be extracted to be fed back to the participants in order to provide the best possible support and monitoring of their learning and instructional processes. Finally, there is a need for providing an efficient and robust computational approach that enables the embedding of the collected information and the extracted knowledge into a CSCL application.

In the next section, we take these entire approaches one step further by incorporating a generic, interoperable, distributed point of view

by means of our collaborative learning platform as a computational model focused on data analysis and management.

PRINCIPLES IN DESIGNING A COMPUTATIONAL MODEL FOR DATA ANALYSIS AND MANAGEMENT

CSCL applications are characterized by a high degree of user-system interaction thus generating a huge amount of action events. The management of action events is a key issue in these applications. On the one hand, the analysis of data gathered from real-life online collaborative learning situations would help one understand important issues in group functioning and collaborative learning process. On the other hand, the study of the action events can be used as a guide both in designing more functional workspaces and software components and in developing better facilities such as awareness, feedback, monitoring of the workspace, assessment and tracking of the group's work by a coordinator, tutor, etc. Indeed, by filtering out the data, an adequate event management makes it possible to establish a list of parameters that can be used for analyzing group space activities (e.g., tutor-to-group or member-to-member communication flow, asynchronism within the group space, etc.). These parameters would allow the efficiency of group activities to be improved and group behavior and individual attitudes of its members in the shared workspace to be predicted.

In addition, in designing CSCL applications it is necessary to correctly organize and administer both the resources offered by the system and the users accessing these resources. All of this user-resource and user-user interaction generates events or logs, which are collected in log files and represent the information basis for the performance of statistical processes aimed at obtaining useful knowledge of the system. This knowledge will facilitate the collaborative learning process by keeping users aware of what is going on in the system (e.g., the contributions of others, the new documents created, etc.) and controlling users' behavior in order to provide them with support (e.g., helping students who are not able to accomplish a task on their own). Furthermore, user-user and user-resource interaction is crucial in any learning collaborative application to make it possible for groups of students to communicate with each other and to accomplish common objectives effectively (e.g., a collaborative classroom activity).

To achieve these goals, a generic, robust, interoperable, reusable, component-based and service-oriented Collaborative Learning Purpose Library (CLPL) [4] (Caballé et al., 2004) is proposed. The CLPL is based on the Generic Programming paradigm (Caballé & Xhafa, 2003) as a computational model to embed information and knowledge from group activity into CSCL applications. This platform constitutes the implementation of the above-mentioned high-level types of categories and the conceptual model of data analysis and management (see next section and Daradoumis, Martínez, & Xhafa, 2006 for a complete description of these three categories and a complete understanding of the conceptual model). The ultimate aim is to support an efficient embedding of the information collected from users and the later knowledge acquired into CSCL applications.

In developing the CLPL, we paid attention to distribution, reusability, flexibility and interoperability as key aspects to address the current strong needs for meeting more and more changing and demanding requirements in software development in general and specifically in the e-Learning domain. Indeed, over the last decade, e-Learning needs have been changing in accordance with ever more complex pedagogical models as well as with technological evolution resulting in e-Learning environments with very dynamic and

changing teaching and learning requirements. These requirements represent a great challenge for the latest trends of software development to be completely satisfied.

In order to meet these requirements, we based the development of the CLPL on the model-driven development (MDD) paradigm and the framework supporting it, namely model-driven architecture (MDA). This new development paradigm has been recently attracting a lot of attention given that it allows software developers and organizations to capture every important aspect of a software system through appropriate models (Czarnecki, 2005). MDA provides great advantages in terms of complete support to the whole cycle development, cost reduction, software quality, reusability, independence from the technology, integration with existing systems, scalability and robustness, flexible evolution of software and standardization, as it is supported by the Object Management Group[5] (OMG).

In proposing MDA, two key ideas have had significant influence in OMG aiming at addressing the current challenges in software development: Service-oriented architectures (SOA) and product line architectures (PLA) (see OMG Web site). As to the former, SOA provides great flexibility to system architectures by organizing the system as a collection of encapsulated services. Hence, SOA relies on services which represent the behavior provided by a component to be met and used by any other components based only on the interface contract. As to the latter, PLA promotes developing large families of related software applications quickly and cheaply from reusable components. In PLA, a certain level of automation is provided in the form of generators (also known as component configuration tools) to realize solutions for large parts of the systems being developed (Czarnecki, & Eisenecker, 2000). Taking these approaches into consideration, the CLPL is based on SOA and the Generic Programming paradigm (Czarnecki, & Eisenecker, 2000; Caballe, & Xhafa, 2003) as the central part of the development in MDD.

There are many views and opinions about what MDA is and is not. However, the OMG, as the most authoritative view, focuses MDA on a central vision (Czarnecki, 2005): Allow developers to express applications independently of specific implementation platforms (such as a given programming language or middleware). To this end, OMG proposes the following principles for MDA developments: First, the development of a UML-based platform independent model (PIM), second, one or several models which are platform specific models (PSM). Finally, a certain degree of automation by means of descriptions is necessary for mapping from PIM to PSM. The development of the CLPL fully followed the first and second principles while ongoing work is dealing with the last by introducing a certain level of automation by means of WSDL descriptions.

In particular, in developing the CLPL, we first created our PIM by applying the following Generic Programming ideas (Caballé & Xhafa, 2003): (1) Define the semantics of the properties and domain concepts, (2) extract and specify the common and variable properties and their dependencies in the form of abstractions found in the CSCL domain, and (3) isolate the fundamental parts in the form of abstractions from which the basic requirements were obtained, analyzed and designed as a traditional three-layer architecture (i.e., presentation, business and information). To this end, first, our PIM was expressed using UML as the standard modeling language promoted by the OMG. Second, two different PSM have been constructed so far from the PIM: A Java implementation in the form of a generic component-based library and a collection of WSDL files organized in directories that are automatically turned into generic Web-services implemented in the desired programming language and allowing developers to implement the services according to specific needs. On the one hand, the Java programming language provides great predisposition to the adaptation and correct transmission of generic software design, which make the software highly reusable. On the

other hand, in order to increase flexibility and interoperability, our service-oriented PSM provides great predisposition to be involved in distributed environments supporting different middleware and programming languages. Finally, in order to automate as much as possible the transition from the PIM to the appropriate PSM, the latest research results are leading us to deal with XMI files (see OMG Web site for details), which are XML-tagged files as the result of coding UML diagrams. In combination with XSL style sheets, it is possible to turn the PIM's XMI files into WSDL files, which represent the input for a Web-service working environment to transform them into a specific-language architecture design (PSM). Lack of comply with standards of the existing UML case tools is the major problem to face next as well as how to provide a more complete and detailed realization of the desired PSM.

The design of the user interface in CSCL collaborative applications (e.g., multi-user editors) offers many more challenges than the design of interfaces for single user applications. The user interface must provide information about what others are doing to efficiently support collaborative tasks, and awareness information regarding the effects of other users' activities has to be communicated by visual or audio signals. The user interface is therefore the main way to support awareness in multi-user collaborative environments. Even though in collaborative learning environments the user interface will usually be in graphic mode, our approach considers a generic focus in order to make the logic part of the application independent from the specific design of the graphic user interface.

The design of the persistence in the CLPL is also generic and thus a disk manager abstraction has been considered. The disk manager acts as a bridge between the future application and its data to make the design of the persistence independent from the specific technology that will manage the data. This way, it is possible to treat both ordinary text files and different database system managers during particularization. Furthermore, a complete technology-independent conceptual data model is provided as part of the PIM, which may be realized in different technologies managing generic persistence.

Finally, robustness is offered through a complete hierarchy of error treatment. As a result, a high degree of component quality and reliability is guaranteed without depending on the error treatment of the specific platform supporting the software.

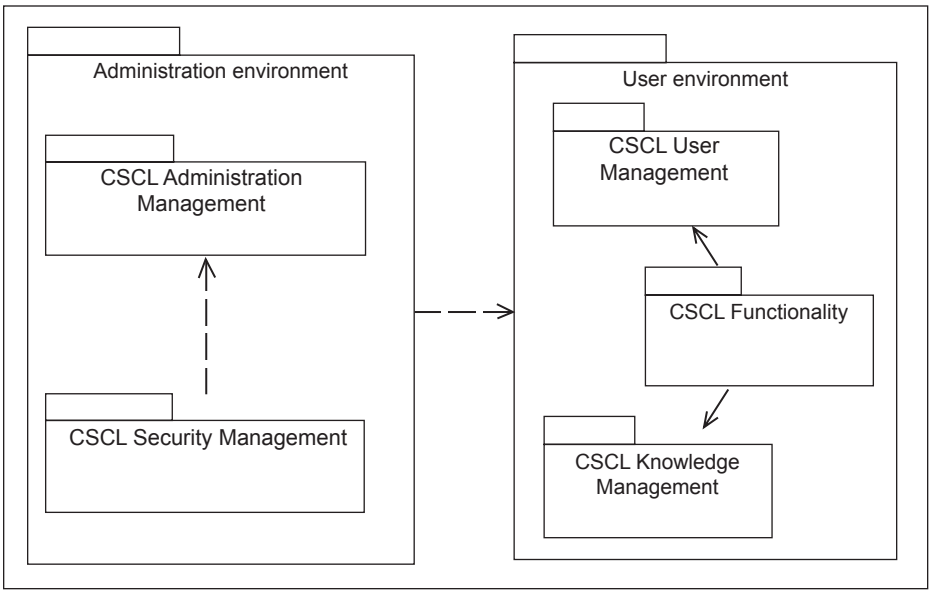
The ultimate aim of the CLPL is to enable a complete and effective reutilization of its generic services and components as the skeleton for the construction of any collaborative learning application, and in particular CSCL applications. Thus, this platform implements the conceptualization of the fundamental needs existing in any collaborative learning experience. In addition, the CLPL is highly interoperable in distributed environments permitting complete flexibility of the services offered in terms of implementation languages and underlying software and hardware platforms.

THE CLPL ARCHITECTURE DESCRIPTION

The CLPL is made up of five components in all (see Figure 1) handling user management, security, administration, knowledge management and functionality. The aim is to map the essential elements involved in any CSCL collaborative learning application. The first three components are briefly described here. Due to its importance for the scope of this chapter, the last two components are explained in great detail later on.

CSCL user management: This component deals with the behavior related to user management encountered in any CSCL applications, that is, who can act as a group coordinator, group member, group-entity and system administrator. It tackles both the basic user management functions in a learning environment (namely registration,

Figure 1. Graphical representation of the CLPL components



de-registration, modifications, joining a group, or meeting group members) and the user profile management. The latter implements the user and group models within a collaborative environment, thus this component provides a generic user profile entity which dynamically allows new user and group needs to be met.

CSCL Security Management: This component contains all the generic descriptions of the measures and rules decided upon to resolve authentication and authorization issues. The aim is to protect the system from both unknown users and the intentional or accidental ill use of its resources. This component's genericity lets programmers implement security issues ad hoc using the latest cryptographic security mechanisms.

CSCL Administration: This component is responsible for managing the specific data coming from log files and those analyses required to perform all the system control and maintenance for the correct administration of the system. The aim is to improve the system functioning in terms of performance and effectiveness. Although user interaction is the most important point to be managed in CSCL applications, it is normally also important to be able to monitor and control the performance and general functioning of the system. This will enable the administrator to continuously track the critical parts of the system and act if necessary. Furthermore, this adds an implicit security layer by monitoring the system (e.g., controlling users' habits make it possible to detect fraudulent use of the system by unauthorized users). Moreover, this component manages the resources of the collaborative workspace, which can be managed by a group member acting as an administrator within the group.

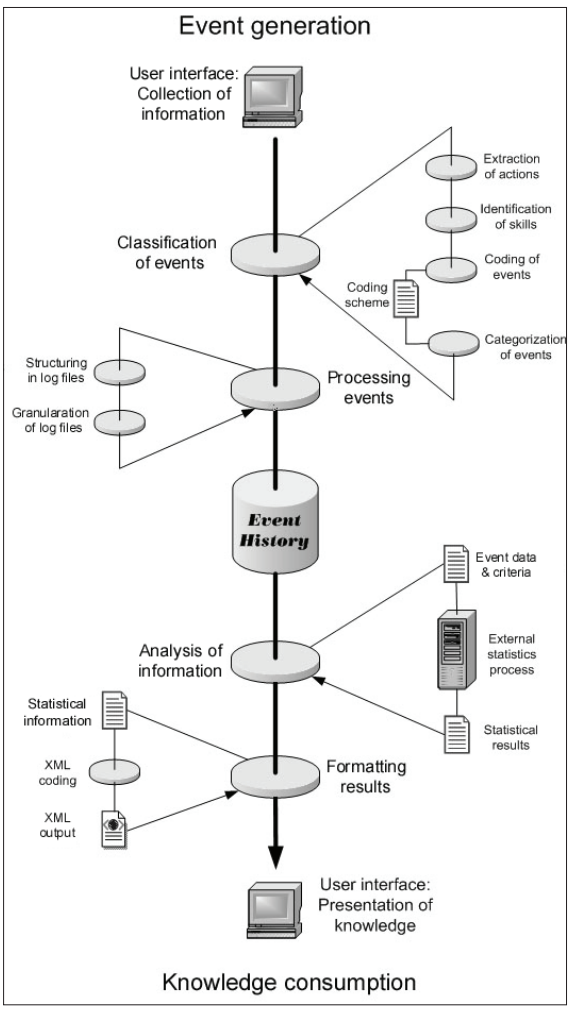
Embedding Information and Knowledge into CSCL Applications

As mentioned previously, our platform represents a computational model that implements the conceptualization of the fundamental needs existing in collaborative learning applications especially for data analysis and management. This

is performed by two specific components related to the knowledge management and functionality support. In the context of our research, the specific aim of this computational model is to entirely cover a process of embedding information and knowledge from group activity into CSCL applications in an efficient and effective manner. This process involves four separate, necessary steps: collection of information, processing, analysis and presentation (see Figure 2 and next subsection for details). The entire process fails if one of these steps is omitted. During the first step, a tight structuring and classification of the generated event information is needed, which is processed in an efficient way in the second step. This information is then analyzed and interpreted in order to extract the desired knowledge. The final step is to provide users with the essential awareness and feedback from the obtained knowledge.

In this section, we are therefore interested in explaining in more detail the last two components of the CLPL, namely CSCL knowledge management and CSCL Functionality. Given that these two components form the core of the computational model for data analysis and management, they are described in great detail.

Figure 2. The process of embedding information and knowledge into applications



CSCL Knowledge Component

The CSCL knowledge management component will manage and analyze all the specific and large user events in order to record user interaction data as information which is crucial for the correct control and administration of the collaborative learning applications. Therefore, this component completely specifies and implements the first two stages (collection of information and analysis) of the mentioned process of embedding information and knowledge into CSCL applications (Figure 2).

The final objective of this component is to extract valuable information from the events processed for later statistical analysis with the aim of revealing useful knowledge from the group activity. This component is made up of the CSCL Activity Management and CSCL Knowledge Processing subsystems, which are explained here.

CSCL Activity Management Subsystem

This subsystem collects, classifies and structures the event information contained in the CSCL application log files so as to make it possible to facilitate its later statistical analysis.

The *log file* is a key entity made up of all the action events occurring in the system over a given period of time and is automatically generated by the system during its usual functioning. This represents the source of information that is later used for the creation of the appropriate statistics.

In CSCL applications there is a need for the classification of all types of user and system events generated according to the above-mentioned three generic group activity parameters o categories (namely, task performance, group functioning, and scaffolding).

Next, we briefly describe each of these three categories. We employ a similar terminology to the one used in the Basic Support for Cooperative Work (BSCW) system (Bentley, Horstmann, & Trevor, 1997) to refer to the actions that can be carried out in an asynchronous groupware platform. However, they are general enough to be abstracted and represent all the typical and basic actions encountered in any asynchronous groupware platform (for a complete description, see Daradoumis, Xhafa, & Juan Pérez, 2005 and Daradoumis, Martínez, & Xhafa, 2006):

Collaborative learning outcome (or task performance) measures those skills that characterize the students who participate in a learning collaborative situation in order to achieve effective group and individual performance of the task and thus obtain a successful learning outcome (see Table 1).

Group functioning indicates the skills that students should exhibit in order to enhance participation accomplish well-balanced contributions, promote better communication and coordination. Moreover, this parameter indicates adequate work load distribution, task management and workspace organization. The purpose of group functioning is to achieve an effective group interaction and functioning in a collaborative learning situation (see Table 2).

Scaffolding shows the different types of social support and help services (McGrath, 1991) that have been identified and accounted for in our model. The participants' actions and contributions aiming at getting or providing help are classified and measured according to whether they refer to the task or group functioning (see Table 3).

Table 1. indicators that model task performance

Skills	**Sub-skills (Learning outcome contribution)**	**Actions (&objects) involved**
Basic active learning skills	Knowledge/info generation	Create doc/note
Supporting active learning skills	Knowledge/info refinement	Edit doc
	Knowledge/info elaboration	Version/Replace doc
	Knowledge/info revision	Revise/Branch doc
	Knowledge/info reinforcement	Create_Noteboard doc/URL /Notes (attach a note to a document, url or debate)
Information processing (perception) skills	Knowledge/info acknowledge	Read event

Table 2. Indicators that model group functioning

Skills	Sub-skills (Group functioning contribution)	Actions (&objects) involved
Active participation behavior and peer involve-ment skills	Participation in managing (generating, expanding and processing) info	Create Event, Change Event, Read Event
Social grounding skills	Well-balanced contributions, adequate reaction attitudes, and role playing	Create Event, Change Event, Read Event, Move Event
Task processing skills	Task planning	Create/Link Appointment Create/ChangeAccess WSCalendar
	Task (and knowledge) management	Create Folder Create Notes (create a debate space)
Workspace processing skills	Workspace organization and maintenance	Move event (cut, drop, copy, delete, forget)
Communication processing skills	Clarification	Change Description/ Change Event doc Change Description url
	Evaluation	Rate document/url
	Description (illustration)	Edit/Change Description Folder Change Description Notes
	Communication improvement	Edit Note Chvinfo/Chvno/Checkin/ Checkout doc Rename Folder/Notes/doc/url/ Appointment/WSCalendar
	Meeting accommodation	ChangeDesc/ChangeDate / ChangeLocation Appointment

Table 3. Indicators that model scaffolding

Social support
Members' commitment toward collaboration, joint learning and accomplishment of the common group goal
Level of peer involvement and their influential contribution to the involvement of the others
Members' contribution to the achievement of mutual trust
Members' motivational and emotional support to their peers
Participation and contribution to conflict resolution
Help Services
Help is timely
Help is relevant to the student's needs
Help is qualitative
Help is understood by the student
Help can readily be applied by the student

At this point, we start introducing and describing step-by-step the above-mentioned process of embedding information and knowledge into CSCL applications (see Figure 2). That will go through the rest of subsystems of the Knowledge Management component and also the Functionality component, which we will explain later.

During the first step of the process, collection and classification of the information, the aim is to correctly classify the users' interaction according to the three generic types of categories described. To this end, a complete and tight hierarchy of events (Figure 3) is provided in this subsystem. In this hierarchy, a certain degree of redundancy is allowed since both the same events to measure different elements are expected and desired. For instance, a group processing event can be simultaneously addressed as both a quantitative parameter to measure group functioning and a qualitative parameter to measure scaffolding.

Furthermore, in a collaborative learning experience, the group activity is driven by participants' actions on the generic collaborative learning resources and these actions are aggregated to the user events to form another hierarchical tree (included in Figure 3). In this hierarchy, at a first level, we can differentiate between active and passive user actions depending on whether or not the student contributes directly to achieving the group objective. At this same level, the support action (i.e., help, motivation and encouragement) is also considered and constitutes another distinct category.

In order to correctly classify the user actions on the resources during group activity according to the event hierarchy, we propose a classification process and a coding scheme (see Table 4) for asynchronous environments based on our conceptual model. In this process, the event information collected from the log files is handled in sequential steps consisting of extraction, identification, coding, and categorization (see Figure 2).

Thus, firstly, we extract from the log files the specific action performed by a user on a resource (e.g., create a note). Secondly, this action is interpreted depending on the type of event that was involved, such as in response to a previous contribution. This represents the essential information

Figure 3. A hierarchy to collect and classify all events generated during the group activity

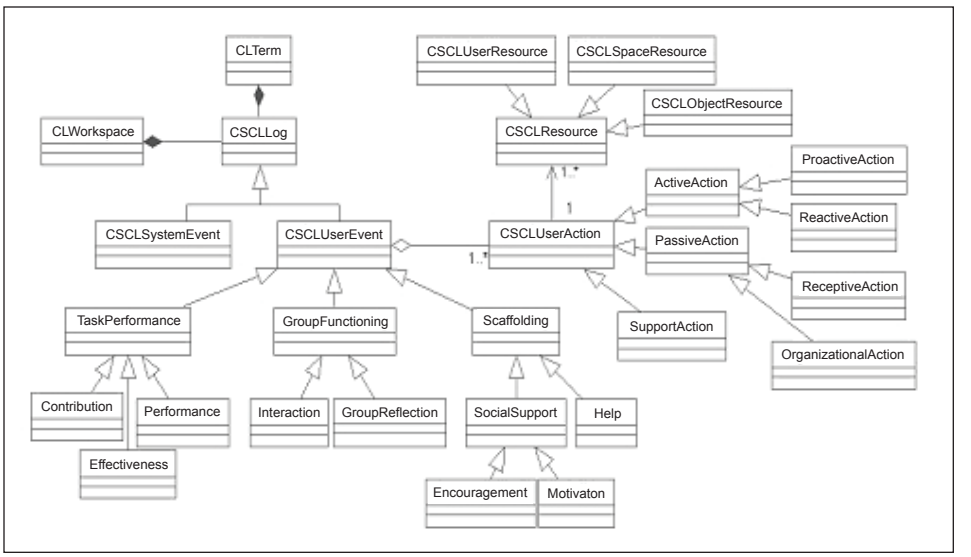


Table 4. Excerpt of a generic coding scheme for asynchronous environments

Code	Action	Event type	Skills	Category
cdc	Create document	Creation	Information generation	Contribution
cda	Create document	Activity	Active participation	Interaction
cdr	Create document	Reply	Information revision	Effectiveness
cde	Create document	Evaluation	Task contribution	Group Reflection
cnc	Create note	Creation	Information reinforcement	Performance
cns	Create note	Support	Members' involvement	Motivation
cnr	Create note	Reply	Information revision	Effectiveness
cna	Create note	Activity	Active participation	Interaction
rdp	Read document	Processing	Information acknowledge	Performance
rda	Read document	Activity	Passive participation	Interaction
mdr	Modify document	Revision	Information revision	Contribution
mda	Modify document	Activity	Active participation	Interaction
mde	Modify document	Evaluation	Task contribution	Group Reflection
rde	Replace document	Elaboration	Information elaboration	Effectiveness
rda	Replace document	Activity	Active participation	Interaction
rde	Replace document	Evaluation	Task contribution	Group Reflection

in the identification of the real intentions or skills shown by the user (e.g., creating a note during a debate can be interpreted as either revision or reinforcement of the information depending on whether the note was created in the context of a reply, an observation, agreement, etc.). Then, during the third step of the process, we uniquely codify the user event according to both the user action performed and the real user skill identified in the context of the action. Thus, for instance, creating a replying note is codified with a unique code. Finally, we categorize the user event into one of the above-mentioned group activity indicators (see Tables 1 through 3).

Given that this classification process is highly generic, we only provide the most abstract form

of categorization based on the above-mentioned generic event hierarchy (see Figure 3). Thus, the specific applications using this process should categorize their event information according to their particularization of this categorization.

Note that although it is possible to use the same classification process for both synchronous and asynchronous environments, we will focus on the latter as this is still the most usual way to collaborate in online collaborative learning environments and permits the complete automation of the classification process. In contrast, in synchronous environments most of this process has to be performed manually and it needs a different coding scheme to codify the user actions.

Once the event information generated in the group activity is collected as log files and correctly classified, CSCL applications need to structure this information. The goal is to make it possible to both prepare information to facilitate its later processing and analysis and allow it to be efficiently addressed in a distributed environment where available (such as in a grid environment). Detailed information about how to structure and parallelize the processing of event log files can be found at Xhafa, Caballé, Daradoumis, and Zhou (2004) and Caballé, Paniagua, Xhafa, and Daradoumis (2005). This forms the second step of the process of embedding information and knowledge into collaborative learning applications (see Figure 2).

At this point in the process, we find the information has been collected, classified and well-structured so that it can easily and efficiently be processed and analyzed during the work of the CSCL Knowledge Processing subsystem.

CSCL Knowledge Processing Subsystem

After the event information from the structured files has been processed, the results of data processing are stored in a database manager system where all the information contained in the structured files should be correctly represented, even if they are distributed in different machines. The aim is to make it possible to consult both the desired data from the database directly (e.g., the number of connected users, the type of documents in a certain workspace) and the computed complex statistical results produced from processing these data as part of the analysis step in the process of information management (Figure 2).

The ultimate objective of this subsystem is to define a bottom-up analysis approach that processes and analyses the user events in order to decode the specific actions of the users describing their interaction during the collaboration activities. This analysis aims at identifying those sequences of actions that can be used to determine typical patterns of interactions (Inaba, Ikeda, & Mizoguchi, 2003).

Thus, at this point in our research our objective is to identify as many best collaborative learning practices as possible, which can then be translated into typical collaborative learning patterns. Based on a model of desired interaction, the system allows us to compare the learners' real interaction processes with the typical interaction patterns in order to infer whether or not the process is effective for the learner. Furthermore, the knowledge revealed by this analysis can enhance self and peer evaluation, which in turn improves the efficiency of group activities, monitoring group behavior and the individual attitudes of its members in the shared workspace. In addition, this knowledge is useful in assisting the tutor by providing the necessary means to support and assess individual and group learning outcomes.

CSCL Functionality Component

This component, which has five subsystems in all, defines the three elemental parts involved in any form of cooperation, namely coordination, communication and collaboration (Caballé et al., 2004). Coordination involves the organization of groups to accomplish the important objectives of

members such as workspace organization and group structure and planning. Collaboration lets group members share any kind of resources while communication represents the basis of the whole component since it enables coordination and collaboration to be achieved by providing them with low-level communication support. Here we describe briefly three subsystems of this component that provide support to the mentioned areas.

CSCL coordination: This subsystem manages both members and resources within a collaborative group so as to both organize and coordinate the learning group and enable tutors to monitor and assess the learning process.

CSCL collaboration: The main purpose of this subsystem is to let participants share resources such as files and applications in a collaborative learning environment. Resource sharing may be in both synchronous and asynchronous modes.

CSCL communication: This subsystem manages all the low-level interactions between two or more participants within a collaborative learning group in both synchronous and asynchronous modes.

The final objective of this component is to provide functional support to CSCL applications in terms of group organization, resource sharing, user interaction, and so on. Moreover, this component implements the last stage of the process of embedding information and knowledge into CSCL applications (Figure 2) by presenting the knowledge generated to users in terms of immediate awareness and constant feedback of what is going on in the system. Due to the importance of this component, we describe here in more detail the specific subsystems of this component, namely *CSCL Awareness* and *CSCL Feedback* that explicitly provide support for awareness and feedback.

CSCL Awareness Subsystem

Awareness is essential for any of the three forms of cooperation seen. It allows implicit coordination of collaborative learning, opportunities for informal, spontaneous communication, and it keeps users informed as to what is happening in the system (Gutwin, Stark, & Greenberg, 1995). On the one hand, when awareness is synchronous, users know in real time what other co-participants are doing (e.g., during a multi-user editor session, who is editing and what is being shown) and which documents are being used by others. On the other hand, when awareness is asynchronous, users receive delayed knowledge of who, when, how and where shared resources have been created, changed or read by other users.

In order to provide the essential awareness information to support collaboration, communication and coordination effectively, this subsystem defines three generic entities respectively, namely *resource state*, *user status* and *group memory*. Each of these abstractions acts as a vehicle so that awareness information can be classified and presented to users in the correct form depending on the type of activity involved. Thus, first, in resource sharing (e.g., a multi-user editor session), participants are continuously modifying the state of the shared application (e.g., writing a new text comment, deleting somebody else's sketch, etc.). This way, the current application state has to be continuously propagated to the users as a news warning signal. Second, it is essential to show the current participants' status so as to be aware of the availability of them for communication (e.g., before sending a message to others it is crucial to know whether or not they are available). Finally, the persistent storage of awareness information is needed during coordination since it allows us to access documents and data, which are commonly stored for later retrieval, and also the context in which they were created. Thus, being aware of others' activities is essential for coordination (e.g., in decision-making, group organization, social engagement, etc.).

Furthermore, as regards the presentation format, this subsystem defines a *flag* as a single abstraction supporting the presentation of awareness

information to users through the user interface by any means: Ranging from a visual and simple signal for warning purposes to complex visual and audio effects to keep participants aware of what is happening in the group activity.

The ultimate objective of this subsystem is to present awareness information to users in a correct, effective and immediate fashion as the presentation step in the process of embedding knowledge into CSCL applications we have been carrying out so far (Figure 2).

CSCL Feedback Subsystem

Feedback in Web-based collaborative learning environments is receiving a lot of attention due to its positive impact on the motivation, emotional state, and problem-solving abilities of groups in online collaborative learning (Zumbach, Hillers, & Reimann, 2003). It aims to influence group participants in a positive manner by means of a steady tracking of parameters outside the task itself (such as motivation and emotional state) and by giving a constant feedback of these parameters to the group. Therefore, when users participate in a CSCL application, they may enhance their abilities by increasing their knowledge about others in terms of motivation, interaction behavior and so on.

Feedback goes one step further than awareness by providing exhaustive information of what is going on in the group over a long period of time (e.g., constantly showing to each group member the absolute or relative amount of the contributions of others). Furthermore, feedback may be obtained about the emotions and motivation of participants through asking them about these states. In all cases, feedback implies receiving information simultaneously both synchronously and asynchronously since the history information shown is continuously updated.

During the feedback process, all new information communicated to the users will have been previously collected, classified and analyzed by the CSCL knowledge management component. As a consequence of the complex knowledge provided to participants in form of feedback (e.g., group's member relative and absolute amount of contributions, group's members variation in motivation and emotional state during last two hours, etc.) this subsystem makes a strong use of the statistical analysis and need to show the results obtained in complex graphical formats.

In this subsystem we define certain generic entities such as *history*, *pool* and *diagram* and functions such as *sorting*. Based on these abstractions it is possible to dynamically gather and store great amounts of history data and statistical results from the group activity in order to constantly update and present them to participants in the appropriate diagrammatic form (e.g., pie chart, histograms, etc.).

AN APPLICATION EXAMPLE: THE DEVELOPMENT OF A STRUCTURED DISCUSSION FORUM

To illustrate the approach, a prototype of a Web-based structured discussion forum was developed to validate the possibilities offered by our computational platform during data analysis and management.

We describe the main guidelines that conducted the design of this prototype that gives new opportunities to learning methodologies, such as learning by discussion, and is applied to new learning scenarios. To this end, a complete discussion and reasoning process is proposed. This application provides significant benefits for students in the context of project-based learning, and in education in general.

Pedagogical Background and Requirements

In collaborative learning environments, the discussion process forms an important social task

where participants can think about the activity being performed, collaborate with each other through the exchange of ideas arising, propose new resolution mechanisms, and justify and refine their own contributions and thus acquire new knowledge (Caballé et al., 2004).

To this end, we propose a complete discussion and reasoning process based on three types of generic contributions, namely specification, elaboration and consensus. Specification occurs during the initial stage of the process carried out by the tutor or group coordinator who contributes by defining the group activity and its objectives (i.e., statement of the problem) and the way to structure it in sub-activities. Elaboration refers to the contributions of participants (mostly students) in which a proposal, idea or plan to reach a solution is presented. The other participants can elaborate on this proposal through different types of participation such as questions, comments, explanations and agree/disagree statements. Finally, when a correct proposal of solution is achieved, the consensus contributions take part in its approval (this includes different consensus models such as voting); when a solution is accepted the discussion terminates.

In a discussion process, participants perform a role according to their profile (e.g., coordinator, member, guest, etc.), have personal collaborative preferences (e.g., language) and must set up environment features (e.g., sound or visual effects, text or voice warnings, etc.) according to their personal characteristics. Participant needs are not static and they evolve as the discussion moves forward.

The Design of the Application

During the design of this application, the generic types of contributions mentioned above were supported by allowing the application to take advantage of the CLPL components. Certain correspondences are described here.

In designing the specification phase, coordination needs to be supported by essential elements such as an *agenda* and a *calendar* so as to perform all the typical tasks in this initial stage of the discussion process (such as group formation, definition of objectives, structuring the task in sub-activities and labor division). During this phase, the CSCL Coordination subsystem gave support through certain generic entities that were particularized into specific needs of this application and as a result the mentioned essential entities and processes were provided. In order to enable the tutor to both monitor and assess the discussion process, the application took advantage of the generic *report* system provided by this subsystem so as to keep track of the performance of participants and assess their contributions.

The application design includes certain thematic annotation cards (such as idea, evaluation, reply, etc.—see Figures 4 and 5) that structure the elaboration phase and can offer full help as well. All events generated are recorded as user actions, analyzed and presented as information to participants either in real time (to guide directly students during the learning activity) or after the task is over (in order to understand the collaborative process). To this end, the CSCL knowledge management component provided full support to the event management. In particular, during the elaboration phase, a complete treatment of the structured task performance events generated enables the system to keep participants aware of the contributing behavior of others, to check certain argumentative structures during discussion and also to open up the possibility to provide feedback based on the data produced. Equally, group analysis outcomes produced by the treatment of group functioning events constitute an important data source that can assist in achieving a more satisfactory solution to the problem during the consensus phase. Furthermore, the coordinator can use this same information to organize well-balanced groups during the specification phase.

Figure 4. The inclusion and setting up of parameters or indicators in the form of labels to classify the participants' interaction in a specific workspace

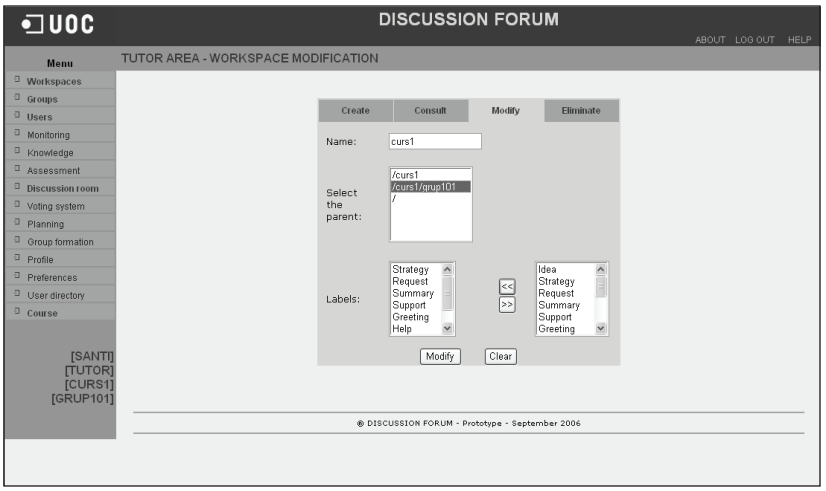
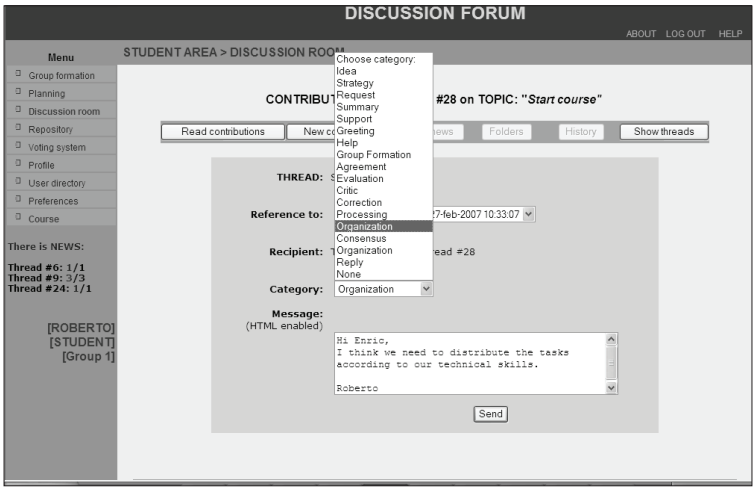


Figure 5. The discussion forum: A list of labels to categorize a contribution



Personal features of the discussion group participants (their role, collaboration preferences and so on) were taken into account and a user and group model were designed so as to allow participants to add new services while their needs evolve as the discussion moves forward. All these user features were included by the CSCL user management component through the CSCL user profile management subsystem, providing a solid support for building and maintaining the user and group model.

Therefore, on the one hand, the structured discussion forum supports a complete discussion process through the realization of three generic contribution types and an open user and group model. On the other hand, this application con-

stitutes a valuable resource that takes advantage of the computational platform to greatly improve essential features of a discussion process such as awareness of participant contributions and enhance the abilities of users by increasing their knowledge of each other in terms of motivation, interaction behavior and so on.

Implementation Issues

This prototype is currently working as a typical client-server Web-based application at the Open University of Catalonia and evolving rapidly to be completed. Taking advantage of the flexibility of the service-oriented approach, we used different languages for the development of the client and the server sides. Thus, on the one hand, PHP resulted in a very suitable programming language to implement the Web pages forming the user interface on the client side. Indeed, the ease of use and create forms and other graphical objects on the client side as well as being supported by most of existing Web servers, convinced us to use this popular programming language. On the other hand, the generic Web-services supporting the business and persistence layers on the server side were implemented in Java as a powerful and experienced language offering great characteristics with regard to robustness, portability, ease of use and extensibility, which create an ideal context for the implementation of the server side.

Experimental Results

From our experience at the Open University of Catalonia, the collection of structured information from an asynchronous discussion forum is highly desired for supporting the learning process. Indeed, the analysis results of this information provide the appropriate knowledge to be presented to participants in terms of awareness and feedback as well as for monitoring purposes. Our prototype allowed us to achieve these goals by supporting the following process: (1) participants are urged to label their contributions according to certain indicators (see Figure 5) based on the generic group activity parameters introduced in Section 1; (2) all information generated during the discussion is collected in log files and processed according to different criteria such as participants and time and type of contribution; (3) this information is then analyzed in order to extract desired statistics such as percentage of each participant's contributions and most active participants; (4) these analysis results are presented to participants in terms of flags showing new contributions that are pending to read, bar-charts with updated statistical information about the relative amount of contributions of each participant, etc. (see Figure 6).

In order to measure the impact of knowledge-based collaborative learning applications, such as our prototype, on the real learning experience at the Open University of Catalonia, a report is usually conducted on several groupware-based courses. Table 5 shows the results of a structured and qualitative report conducted at the end of each term in a course called "Management of Information Systems," which is formed by about 300 students distributed into 4 classrooms and 60 working groups. In this report, students are requested on evaluating the results of using the set of collaborative learning tools provided by our university to support both synchronous and asynchronous group activity.

CONCLUSION AND FUTURE WORK

This chapter describes an architecture solution that provides an efficient management of information that comes from online collaborative learning activity in order to enhance the collaborative learning process. To this end, a computational model of collaborative learning interaction in the form of a generic platform was described in detail that can be used for constructing collaborative learning applications that are enabled to manage information and knowledge in an efficient manner. This

Figure 6. Snapshot of the application showing the current list of discussion threads. Awareness of what is happening and where is presented by news and flags. Updated feedback is presented in the form of numeric and graphical quantitative statistics that allows participants to compare their performance to that of their group mates.

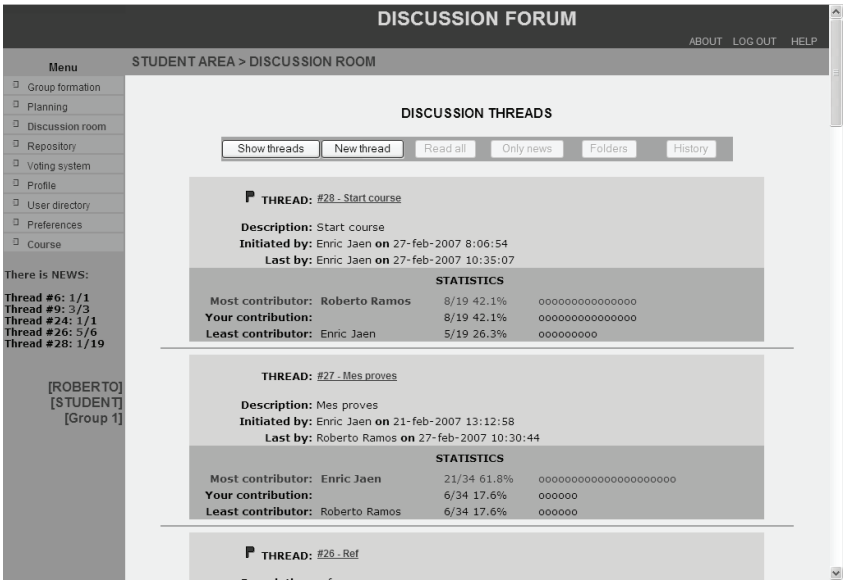


Table 5. Excerpt of a report of using knowledge-based collaborative learning applications

Selected questions	**Average of structured responses (Strong positive, Positive, Neutral, Negative)**	**Excerpt of selected students' comments**
Assess the collaborative learning tools (CLT) used.	Positive	"Apart from technical problems with the server, the CLT run smoothly and fulfilled my expectations" " Despite the distance, CLT achieved to support our work"
Did the CLT help you achieve the course goals?	Positive	"We failed not because of having problems with the CLT but the lack of engagement of certain members" "The CLT eased the group discussion, which enhanced our work."
Evaluate how the CLT fostered your active participation?	Strong positive	"It is always hard to work form the distance and keep in touch with the other group members. The CLT made this part easier by constantly providing information about what the other group mates are doing"
Describe problems and conflicts found in using the CLT.	Neutral	"I needed to make an effort to learn how to deal with the CLT but after a while I got used to it."

computational approach implements a conceptual model of data analysis and management that was briefly introduced. Finally, in order to validate the possibilities offered by this platform, a prototype of a structured forum was developed and used in experimental online collaborative learning activities. The experience gained gave us very useful insights and the confidence to use this application further in real collaborative learning situations in the Open University of Catalonia.

Currently, we are working on how to automatically describe WSDL files from our PIM model as part of the MDA-based development so that it is possible to generate PSM implementations of our collaborative learning platform in different programming languages and middleware. The next step for our prototype is to install the service-oriented PSM in the form of Web services in the nodes of a real distributed platform such as PlanetLab turned into a Grid environment. The aim is to greatly increase the application's performance by parallelizing the clients' requests and thus making it possible to provide costly functionality, such as the constant presentation of complex knowledge in real time. Our experience (Xhafa et al., 2004; Caballé et al., 2005) in dealing with distributed environments makes us confident of being successful.

Further work focuses on investigating how to integrate a portable, general and reusable collaborative learning ontology into our generic platform as a declarative representation of the knowledge embedded into collaborative learning applications with the aim of both describing how these systems are built and understanding how real groups work.

ACKNOWLEDGEMENT

This work has been partially supported by the Spanish MCYT project TSI2005-08225-C07-05. We would also like to thank our colleagues at the Open University of Catalonia, the Polytechnic University of Catalonia, and the University of Valladolid in Spain for their human and scientific support as well as all the students who eagerly participated in this experience.

REFERENCES

Amin, K., Nijsure, S., & von Laszevski, G. (2002). Open Collaborative Grid Services Architecture (OCGSA), In *Euroweb 2002 Conference, The Web and the GRID: From e-Science to e-Business*. (pp. 101-107). Oxford, UK: The British Computer Society.

Anido, L., Llamas, M., Fernández, M.J., Caeiro, M., Santos, J., & Rodríguez, J. (2001). A component model for standardized web-based education. *ACM Journal of Educational Resources in Computing 1(2)*, 1-21.

Bacelo P.T.A., & Becker, K. (2002). A component-based architecture tosupport collaborative application design, In Haake and J. Pino (Eds.), *Groupware: Design, Implementation and Use*. (LNCS 2440, pp. 134-143). London, UK: Springer-Verlag.

Bentley, R., Horstmann, T., & Trevor, J. (1997). The World Wide Web as enabling technology for CSCW: The case of BSCW. *Computer-Supported Cooperative Work: The Journal of Collaborative Computing, 6(2)*, 111-134.

Bote-Lorenzo, M. L., Dimitriadis, Y. A., & Gómez-Sánchez, E. (2003). Grid characteristics and uses: A grid definition. In F. Fernández et al. (Eds.), *European Across Grids Conference* (LNCS 2970, pp. 291-298). Berlin: Springer-Verlag.

Caballé, S., & Xhafa, F. (2003). A study into the feasibility of generic programming for the construction of complex software. In *Proceedings of the 5th Generative Programming and Component Engineering/Net.Objectsdays 2003*, (pp. 441-446). Retrieved March 10, 2007, from

http://www.old.netobjectdays.org/pdf/03/papers/ws-yrw/441.pdf

Caballé, S., Xhafa, F., Daradoumis, T., & Marquès, J.M. (2004). Towards a generic platform for developing CSCL applications using grid infrastructure. In *Proceedings of the First International Workshop on Collaborative Learning Applications of Grid Technology*, Chicago: IEEE Computer Society

Caballé, S., Paniagua, C., Xhafa, F., & Daradoumis, Th. (2005). A grid-aware implementation for providing effective feedback to on-line learning groups. In R. Meersman et al. (Eds.), *On the Move to Meaningful Internet Systems 2005* (LNCS 3762, pp. 274-285). Berlin: Springer-Verlag.

Czarnecki, K. (2005). Overview of Generative Software Development, In J.-P. Banâtre et al. (Ed.), *Unconventional Programming Paradigms (UPP) 2004* (LNCS 3566, pp. 313-328). Berlin: Springer-Verlag.

Czarnecki, K. & Eisenecker, U.W. (2000). *Generative programming: methods, techniques, and applications.* Boston, MA: Addison-Wesley.

Daradoumis, T., Xhafa, F., & Juan Pérez, A. (2005). A framework for assessing self peer and group performance in e-learning. In T.S. Roberts (Ed.), *Self, Peer, and Group Assessment in E-Learning.* (pp. 279-294). Hershey, PA: Idea Group.

Daradoumis, T., Martínez, A. & Xhafa, F. (2006). A layered framework for evaluating online collaborative learning interactions. *International Journal of Human-Computer Studies. Special Issue on "Theoretical and Empirical Advances in Groupware Research," 64*(7), 622-635.

Dillenbourg, P. (1999). Introduction; What do you mean by "Collaborative Learning?" In P. Dillenbourg (Ed.), *Collaborative learning. Cognitive and computational approaches* (pp. 1-19). Oxford: Elsevier Science.

Foster, I. & Kesselman, C. (1998). *The Grid: Blueprint for a future computing infrastructure*. San Francisco: Morgan Kaufmann.

Gutwin, C., Stark, G., & Greenberg, S. (1995). Support for workspace awareness in educational groupware. In J. L. Schnase & E. L. Cunnius (Eds.) *ACM Conference on Computer Supported Collaborative Learning* (pp. 147-156). Mahwah, NJ: Lawrence Erlbaum Associates.

Inaba, A., Ikeda, M., & Mizoguchi, R. (2003). What learning patterns are effective for a learner's growth?—An ontological support for designing collaborative learning. In *Proceedings of the International Conference on Artificial Intelligence in Education* (pp. 219-226). Sydney, Australia.

Koschmann, T. (1996). Paradigm shifts and instructional technology: An introduction. In T. Koschmann (Ed.) *CSCL: Theory and practice of an emerging paradigm* (pp. 1-23). Mahwah, NJ: Lawrence Erlbaum Associates.

Martínez, A., de la Fuente, P., & Dimitriadis, Y. (2003). Towards an XML-based representation of collaborative interaction. In B. Watson et al. (Eds.), *Proceedings of the International Conference on Computer Support for Collaborative Learning 2003*, Bergen, (pp. 379–384). Dordrecht: Kluwer Academic Publishers.

McGrath, J.E. (1991). Time, interaction and performance: A theory of groups. *Small Group Research*, *22*(2), 147-174.

Watson, P. (2003). Databases and the grid. In F. Berman et al. (Eds.), *Grid computing: Making the global infrastructure a reality.* (pp. 363-384). Chichester, UK: John Wiley & Sons, Inc.

Xhafa, F., Caballé, S., Daradoumis, Th., & Zhou, N. (2004). A grid-based approach for processing group activity log files. In R. Meersman et al. (Eds.), *On the Move to Meaningful Internet Systems 2004,* (LNCS 2392, pp. 175-186). Berlin: Springer.

Zumbach, J., Hillers, A., & Reimann, P. (2003). Supporting distributed problem-based learning: The use of feedback in online learning. In T. Roberts (Ed.), *Online Collaborative Learning: Theory and Practice* (pp. 86-103), Hershey, PA: Idea Group Press.

ENDNOTES

[1] WebCT Learning Systems is found at: http://www.webct.com (Web page as of March 2007).

[2] PhpBB Community Building Software is found at: http://www.phpbb.com/ (Web page as of March 2007).

[3] Moodle is found at: http://moodle.org/ (Web page as of March 2007).

[4] The Java API of the CLPL is found at: http://cv.uoc.edu/~scaballe/clpl/api/ (Web page as of March 2007).

[5] Object Management Group: Model-Driven Architecture is found at: http://www.omg.com/mda (Web page as of March 2007).