SCISPACE
formerly Typeset

Open access • Proceedings Article • DOI:10.1109/CCGRID.2004.1336567

Towards a generic platform for developing CSCL applications using Grid infrastructure — Source link

Santi Caballé, Fatos Xhafa, Thanasis Daradoumis, Joan Manuel Marquès

Institutions: Open University of Catalonia

Published on: 19 Apr 2004 - Cluster Computing and the Grid

Topics: Grid computing, Grid, Application software and Generic programming

Related papers:

- Collaborative learning : cognitive and computational approaches
- A Grid-Based Approach for Processing Group Activity Log Files
- Gridcole: A tailorable grid service based system that supports scripted collaborative learning
- An enabling framework for master-worker applications on the Computational Grid
- Groupware: Design, Implementation, and Use

Share this paper:

View more about this paper here: https://typeset.io/papers/towards-a-generic-platform-for-developing-cscl-applications-2toin6zpzd

Towards a Generic Platform for Developing CSCL Applications Using Grid Infrastructure

S. Caballé F. Xhafa T. Daradoumis J.M. Marquès
Open University of Catalonia, Department of Information Sciences
Av. Tibidabo,39-43, 08035 Barcelona, Spain
{scaballe,fxhafa,adaradoumis,jmarquesp}@uoc.edu

Abstract

The goal of this paper is to explore the possibility of using CSCL component-based software under a Grid infrastructure. The merge of these technologies represents an attractive, but probably quite laborious enterprise if we consider not only the benefits but also the barriers that we have to overcome. This work presents an attempt toward this direction by developing a generic platform of CSCL components and discussing the advantages that we could obtain if we adapted it to the Grid. We then propose a means that could make this adjustment possible due to the high degree of genericity that our library components is endowed by being based on the Generic Programming paradigm. Finally, an application of our library is proposed both for validating the adequacy of the platform which it is based on and for indicating the possibilities gained by using it under the Grid.

1. Introduction

Computer-Supported Collaborative Learning (CSCL) is one of the most influencing research paradigms dedicated to improve teaching and learning with the help of modern information and communication technology [1]. Collaborative or group learning refers to instructional methods where students are encouraged to work together on learning tasks. Project-based collaborative learning proved to be a very successful method to that end [2].

CSCL applications must provide both advanced support for distributed collaborative activities and the necessary functionalities to all participants in a collaborative learning experience. One of the main challenges in the development of CSCL applications is to achieve a high degree of adaptability and extensibility in order to deal with system evolution. We achieve this goal by proposing a generic platform that assembles different software components into a library, called Collaborative Learning Purpose Library (CLPL). Our ultimate objective is to use CLPL to develop robust, reusable, integrated, useful and innovative CSCL applications.

The synergy between component-based software technologies and CSCL has already been explored in some research work [3, 4, 5]. Our research takes this previous work one step further by proposing an approach that aims at developing generic software components (such as, user management, security management, administration, knowledge management and functionality components) that can be integrated into a generic platform which makes possible to work with all of them as a whole and widely reuse them for constructing specific CSCL applications.

Furthermore, Grid technology provides a set of interesting features which represent an ideal context for supporting and producing major benefits for CSCL applications [6, 7]. Such important features include: large scale of Grid infrastructures, wide geographical distribution of resources, multiple administrations from different organizations, transparent and dependable access as well as the capability of granting access to shared, heterogeneous resources in very dynamic environments [8].

Considering the benefits provided by Grid it is possible for educational organizations to make use of true collaborative learning environments that enable the involvement of large number of single/group participants (teachers, students, tutors among others) who can potentially belong to many different organizations, possibly situated at very different locations, and transparently share a huge variety of both software and hardware resources while enhancing human-to-human interaction (synchronously or asynchronously) through a friendly 3D-based user interface. Leveraging the inherent performance potential of Grid infrastructure for CSCL applications

0-7803-8430-X/04/$20.00 ©2004 IEEE

makes it possible to greatly enhance the collaboration between users in terms of both participant scalability (adding as many participants/groups as necessary) and resource availability (replicating and executing them in multiple Grid nodes) enabling collaboration as the most important learning method.

In the context of Project-based Collaborative Learning, it is important to provide resources that facilitate the realization of a complex software project by virtual learning groups. We may need resources such as data repository machines, a data base server, a groupware tool, a facility for performing remote operations, a web page server, a discussion forum, etc. To that end, a Grid infrastructure proves to be a necessary technology for providing support to two main aspects:

a. Offer a transparent and user-friendly management of the shared resources, relieving the students of the responsibility of administering the resources by themselves and thus gain flexibility in adding or removing resources.
b. Accomplish the needs of the CLPL itself in a more effective way. This includes among others: provide an easy and efficient maintenance of data structures; enable faster calculation processes, for example facilitate the processing of log files which may contain a very large amount of data (hundreds of thousands of recorded events).

The rest of the paper is organized as follows. Section 2 presents an overview on existing platforms for CSCL applications. Section 3 studies the domain requirements of the CLPL, whereby section 4 describes the development of the CLPL components. Section 5 discusses our effort toward merging CSCL, component-based software and Grid technologies. Finally section 6 presents an initial proposal of a possible application example that can be constructed over the proposed infrastructure.

2. Overview on existing platforms for CSCL applications

The development of generic platforms, frameworks and components is largely used for the construction of complex software systems through the reuse technique. This approach has been successfully applied to different domain applications thus providing applications of increased quality and reduced cost and development time. Due to this, this approach has attracted the attention of developers from the CSCL domain. In the last years, with the increasing interest to the CSCL applications, several proposals have been made in the literature making a first attempt towards the development of platforms for CSCL applications.

We now turn to briefly review some of these proposals and point out several important aspects related to the CSCL domain as well as to the used technology that have not been taken into account in these approaches.

Bacelo et al. [9] introduce a component-based architecture to support collaborative application design. In this work the authors concentrate on how the reuse techniques (components and object-oriented framework) can be applied to the CSCL domain and just some initial ideas are presented for the component-based architecture. At an analysis level, there are some interesting ideas, however they present some limitations since only the communication, cooperation and information sharing aspects are considered. Clearly, there are several other aspects of the CSCL domain needed to be taken into account, such as awareness and knowledge management, and also the above-mentioned aspects need to be analyzed at a deeper level. Another issue not present in this approach is that, in a dynamic environment, CSCL applications will need to use resources in a transparent manner.

As mentioned in Section 1, some approaches describe an initial attempt to remediate this deficiency [7, 8]. In addition, they point out the use of the Grid computing in the development of components for the CSCL domain. The Grid technology has been also used for specific CSCL environments. For instance, the *Access Grid Collaboration Environment* [10] supports group-to-group collaboration for remote visualization or interactive applications, using a high-bandwidth environment for virtual meetings and events.

It is worth mentioning here that in any CSCL platform the communication infrastructure will play an important role; due to this, the use of Grid technology is receiving a lot of attention. Ochoa et al. [11] deal with the design of the communication infrastructure for a groupware system by identifying several layers in the communication of a groupware system.

In this paper we take all these approaches one step further, on the one hand, by better analyzing and taking into account most of the CSCL requirements and, on the other, by incorporating the Grid technology as an underlying infrastructure for our CSCL Library.

3. CSCL Library domain requirements

All CSCL applications have their users and resources which must be managed. In addition, in this domain the applications have a strong necessity to control both users and resources in order to restrict user access to the valuable system's resources. Security and communication issues are very important in this

context as well as to obtain knowledge about what is happening in the system.

If we ask ourselves how users and resources from different CSCL applications are managed, we can see that their management and, most importantly, their implementation are very different. Even so, all of these systems are not so different when passing from a particular system into another; this means that there is a common basis among them which avoids a new CSCL application being developed from scratch.

3.1. A common basis for the CSCL applications

In the last years there has been an explosion of new CSCL applications aiming to create collaborative learning environments where students, teachers, tutors, etc., are able to cooperate with each other in order to accomplish a common learning goal. To achieve this goal, the collaborative applications must provide support to three essential aspects: coordination, collaboration and communication; with communication being the base for reaching coordination and collaboration [11]. Collaboration and communication might be *synchronous* or *asynchronous*. The former means cooperation at the same time with typically fine-grained notifications giving immediate feedback about the activities of other participants whereby the shared resource (such as a text document and a message) will not have a lifespan beyond the sharing. The latter means cooperation at different times and the shared resource will be stored in a persistent support.

The different areas overlap each other (Fig.1), and any collaborative system must support all the three aspects. We call them *CSCL functionality*.

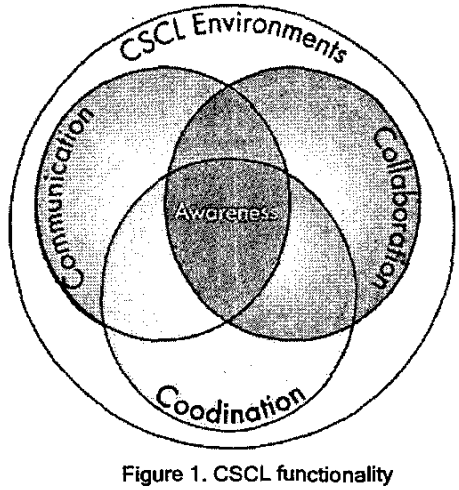

Figure 1. CSCL functionality

Coordination is an important aspect of any collaborative activity. It entails the combination and sequencing of otherwise independent work toward the accomplishment of a larger goal. In a collaborative learning environment, coordination mostly refers to the tasks toward the learning group formation and the definition and planning of the group objectives. Moreover, the group coordinator may track task status, deadlines, resource usage, working results, or other critical process parameters to correctly lead the group.

Collaboration relies on students sharing all kind of documents. The sharing of resources between several participants is therefore a central functionality of CSCL systems. Sharing may be synchronous, with several participants accessing the same resource at the same time (that is, they work on the same copy of the document), or asynchronous, with different participants accessing the same resource at different times (each of them works on a different copy of the same document).

Communication is another functional aspect of collaboration systems aiming to support the communication between two or more collaborative learning participants. Communication includes text messages, spoken interactions, or non-verbal exchanges like gestures in a video conference [12]. Communication may take place asynchronously (different participants communicate at different times such as email, debate, etc.) or synchronously (participants communicate at the same time such as chat, videoconference, etc.). The communication support is based on four elements involved: a *message* as the information carrier between a *sender* process and a *recipient* process (which receives and possibly process the message) through a *channel* [11]. Moreover, in this context, it is necessary to implement different ways of message addressing such as point-to-point, multicast and broadcast.

Awareness is essential for any of the three forms of cooperation seen above. It allows implicit coordination of collaborative learning, opportunities for informal, spontaneous communication and gives users the necessary feedback about what is happening in the system. In particular, on the one hand, synchronous awareness lets users know exactly what other co-participants are doing (e.g. during a shared editing session shows who is editing what) and when documents are in use by others. On the other hand, asynchronous awareness determines who, when,, how and where shared resources have been created, changed or read by others.

In order to improve the collaboration within a group it is important to take into account both current and future behavior of all user types and the fact that user objectives or intentions may change as they interact with the system. To that end, it is essential to design some kind of *user and group models* describing, for example, the user characteristics, intentions, beliefs,

knowledge, skills, roles and collaborative activities. Moreover, the user and group models should be open enough to let add new services and collaborative activities to them according to the participant needs.

The design of the CSCL user interface offers many more challenges than the design of interfaces for single user applications (e.g. multi-user editors). The user interface must provide information about what others are doing to efficiently support collaborative tasks and additional information has to be presented. The latter refers to the effects of other users' activities which must be communicated by visual or audio signals. Therefore, the user interface is the main way to support awareness in multi-user collaborative environments.

Although most research efforts in CSCL areas have been dedicated to developing distance learning environments, most learning activities still take place in the traditional face-to-face classroom [12]. To that end, given that our library is highly generic, it supports the common basis from both scenarios and it is possible to instantiate CSCL applications both for virtual learning (i.e. most of participants are physically in different places) and for traditional learning (i.e. all the participants are physically found in the same place, usually in a classroom). In this paper, though we will mostly refer to virtual CSCL environments on the Grid, the principles are the same for both scenarios.

3.2. The importance of the event management

The CSCL applications are characterized by a high degree of user-system interaction thus generating a huge amount of action events. The management of the action events is a key issue in CSCL applications since, on the one hand, the analysis of data gathered from real life CSCL situations would help understand important issues in group functioning and collaborative learning process; on the other hand, this can guide us both in designing more functional workspaces and CSCL components and in developing better facilities such as awareness, 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 allows us to establish a list of parameters that can be used for analysing group space activities (e.g., tutor-to-group or member-to-member communication flow, asynchronism within the group space, etc.). These parameters would allow to improve the efficiency of group activities and to predict group behaviour and individual attitudes of its members in the shared workspace. Due to its importance, we pay a special attention to event management, by developing a specific component in our CLPL, called *CSCL Knowledge Management.*

4. Development of the CLPL components

Our objective is to construct a Collaborative Learning Purpose Library[1] (CLPL) based on the Generic Programming paradigm [13] so as to enable its generic software components a complete and effective reutilization for the construction of specific CSCL applications. Robustness will be offered through a complete hierarchy of error treatment and so component quality and efficiency will be also guaranteed. In developing the library, we have first designed and implemented a more generic one, namely a Generic Purpose Library (GPL), whose domain requirements are a high degree of user-user and user-system interaction as well as the optimal management of system's resources. By including the CSCL domain into the GPL domain, the CLPL will represent a particularization from the GPL[2].

4.1. Architecture description

The CLPL is mainly made up of five components (Fig.2) of which we will mainly concentrate on those that are closely related to the CSCL domain.

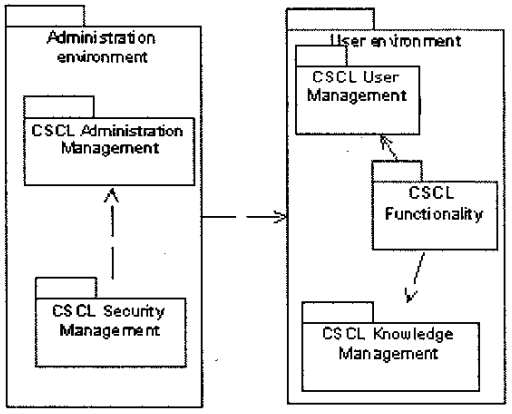

Figure 2. CLPL package diagram

CSCL User Management component: this contains all the logics related to the CSCL system user management which can act as a group coordinator, group member, group-entity and system administrator. It will tackle both the basic user management functions in a learning environment (namely registration, drop

[1] See CLPL API at: http://cv.uoc.edu/~scaballe/clpl/api/index.html

[2] See GPL API at: http://cv.uoc.edu/~scaballe/gpl/api/index.html

out, modifications and 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 the generic *ProfileElement* entity which dynamically allows adding new user and group needs.

CSCL Security Management component: this contains all the generic descriptions of the measures and rules decided to carry out the authentication and authorization issues and so protecting the system from both the unknown users and the intentional or accidental bad use of its resources. Its genericity lets programmer implement them with the ultimate cryptographic security mechanism existing.

CSCL Administration Management component: this contains those specific data (through log files) and processes (statistical computations) to carry out all system's control and maintenance with the aim of correctly administering the system and to improve it in terms of performance and security. Moreover, it will manage the system's resources (the collaborative workspace's resources can be managed by a group member who will act as an administrator within the group).

CSCL Knowledge Management component: this manages all the specific and large user events in order to record the data of users' interaction as crucial information. This allows us both to inform and keep the user aware of what is going on in the system and to control the user behavior as well as the system resources. To that end, it has been split into the *activity management* and *knowledge processing* modules. The former includes a complete hierarchy of user events classified into three main event types to be managed: *task performance* (i.e., collaborative learning process), *group functioning* (i.e., interaction behaviour) and *scaffolding* (i.e., social support such as support, help and encouragement). The latter also includes another generic hierarchy that contains those statistical criteria which are most common in these environments (e.g., the number of students connected over a period of time, the average student working session, etc.). Furthermore, it will enable log information to be exported and extracted in different formats for a posterior statistical analysis. The final objective of this component is to extract valuable information from the events processed for later analysis with the aim to reveal useful knowledge.

CSCL Functionality component: this forms, along with the previous component, the basis of the CSCL environments by defining the already mentioned three basic elements involved in any groupware application: coordination, communication and collaboration. Regarding *coordination*, this component offers the basic tools to facilitate group organization in planning and accomplishing the members' objectives as well as group monitoring by modeling the awareness of its participants. The *communication* support module involves the above-mentioned four basic elements, namely, sender, message, channel and receiver, and can be implemented in several ways depending on the means of message addressing such as: point-to-point, multicast and broadcast. Moreover, each one can be delivered asynchronously (such as an email, where the message is made persistent) or synchronously (such as a chat, where the message is not made persistent and thus it cannot be restored once the conversation ends). Finally, the *collaboration* support module lets members share any kind of resources (both software and hardware) also in both modes synchronous (such as real-time editors) and asynchronous (such as file sharing). Both communication and collaboration modules will generate many events which, previously processed by the *CSCL Knowledge Management* component, will be communicated to the users.

Once these components have been analyzed and designed, their implementation is done in Java due to its great predisposition to adapting and correctly transmitting generic software design and making the software highly reusable.

4.2. User interface, persistence and exceptions

The *user interface* is generically focused so as to make particularization in graphic and text modes possible. Even though, in CSCL environments, the user interface will usually be in graphic mode, it is necessary to consider this abstraction in order to make the logical part of the application independent from the specific design of the graphic user interface.

The design of the *persistence* is also generic and so a *disk manager* abstraction has been considered to act 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 and allow the treatment of both ordinary text files and the database during particularization.

Finally, a complete hierarchy of exception provides a high degree of reliability without depending on the error treatment of the specific platform supporting the software.

5. Use of the Grid infrastructure

Most of the potential benefits provided by Grid have two crucial aspects in common, which software applications greatly take advantage of. These are *transparency* and the final client *simplicity*. The former refers to making the system users independent from the

complex, heterogeneous and dynamic resources and services (i.e. *ubiquitous computing*) that Grid provides. The latter refers to the capabilities of the Grid provider to support the whole system load, thus facilitating the final client the use of very simple terminal devices to access to the Grid.

5.1. CSCL on the Grid

The advantages of the above-mentioned aspects when applied to the CSCL context are multiple and highly important. The *transparency* benefit allows learning virtual environment participants (i.e. students, teachers, tutors, etc.) to access to a huge amount of resources (both software and hardware) without having to worry where the resources are located and how they are managed and even though the location of the node or nodes supporting them are dynamically changing all the time (due to performance reasons of the Grid).

With regard to the final client *simplicity*, it lets CSCL users access to the entire Grid resources and services through a very simple terminal device (as those desktops found in a classroom or at home), however, from the point of view of the CSCL users it seems as if their machines are very powerful and they have got a huge repository of educational resources. Furthermore, the users have a great variety of heterogeneous, common, low-power devices to use, such as traditional desktops, laptops, PDAs and so on letting everyone be able to access to the Grid from everywhere (e.g. for a practical work the classroom can be moved outside the building without missing basic elements such as blackboard, books, notebooks, pencils and so on).

There are other many benefits of the Grid that the specific CSCL applications can take advantage of. One of the most important is the ability to implement *reality scenarios* aiming to achieve both user learning environments and user task outcomes as close to the *reality* as possible. For instance, on the one hand, the large computing support and high network bandwidth of a Grid may enable the designer to base the application's graphical user interface on virtual reality, providing complex and highly cost 3D graphical worlds which helps virtual environment learning participants feel as being at a more real context. On the other hand, the Grid's high-throughput computing support can let individual students avoid their particular computing limitations and carry out real practical tasks with high-throughput computing needs (e.g. computer science students can take advantage of the Grid to carry out, through their personal computer, a class activity in which it is necessary to process a real database made up of millions of rows).

5.2. The CLPL and the Grid

Using the Grid as the underlying infrastructure of our generic CLPL, it would allow us to exploit its performance potential to smartly satisfy our library's specific needs in terms of system log treatment, user and resource management, system administration, user interface and so on. As a consequence, the final objective is to use Grid in CSCL environments to greatly improve the resource availability, participant scalability, security, user awareness, user interaction with the system among others.

More specifically, taking advantage of the large scale of the Grid and its wide geographical distribution, all the software resources of a learning organization could be replicated and stored in multiple Grid nodes (for example, according to the distribution of students and learning centres). This enables us to reduce the response times when accessing the system resources and allow our platform to provide a complete resource management (e.g. a specific application could automatically push all the group updates to each participant each time they access to the system).

In addition, the exhaustive event management provided by the components of our library is based on the system's log file and will contain a huge amount of system events generated by the user interaction. These events are processed and analyzed in order to obtain system knowledge from which it is possible both to keep the user informed about what is happening and to monitor the users' interaction. Therefore, since the log file treatment needs both *high-throughput computing* support and *data-intensive computing* support, the use of the Grid would greatly facilitate these tasks, letting the administrator make a deeper analysis of the system users (e.g. a teacher can monitor individual behaviour of each group member more effective). As a result, the administrator is able to take appropriate measures to handle the specific situations generated (i.e. to better orientate the single/group participants if necessary).

5.3. How to incorporate Grid into the CLPL

Multi-computers on the Grid need new data structures fully taking advantage of the distributed memory and processing capacity and allowing files to span for processing in distributed Grid nodes that provide access performance and are accessible from multiple autonomous and heterogeneous clients, including the mobile ones not requiring any centralized data access computation or directories to avoid single points of failure. One solution is the Distributed Scalable Data Structures (SDDSs) [14] capable of scaling to thousands of Grid nodes with constant

access performance, very low search times and transparent distribution. Our GP-based generic platform is well suited to support the use of such data structures as a crucial requirement of any CSCL application to be adapted to the powerful and dynamic Grid infrastructure. Indeed, the above-mentioned *disk manager* abstraction and its available specialisations represent a bridge that keeps the logic of the application independent from its data in order to particularize the persistence treatment in different models, among them, the SDDSs so as to permit the specific CSCL applications to work conveniently on a Grid infrastructure.

The *grid services*, defined by OGSA and specified by OGSI [15], include the *notification* service that allows clients to be *notified* of changes that occur in a grid service. In addition, as seen before, maintaining user awareness of others' participation is a key point in any CSCL environment, becoming the cornerstone of our generic platform whose genericity lets programmer use the grid services to effectively notify users of the new events occurred in their workspaces. Therefore, as an example, the CLPL *communication* subsystem, which is in charge to communicate the events to the users, is able to implement the *observer/observable* design pattern and through the notification grid service may inform the users in either *pull* or *push* modes.

Another critical point in a Grid infrastructure is that of security. As explained before, the *CSCL Security Management* component includes highly generic security issues that can be implemented by any specific cryptographic system such as X.509 certificates which is the current mechanism the Globus consortium [15] uses as a way to authenticate and authorize users to utilize Grid system resources.

6. A CLPL application: development of a structured discussion forum

As an example to validate the possibilities offered by the integration of the CLPL under a Grid infrastructure we propose the design and construction of a structured discussion forum that may provide new opportunities to learning methodologies, such as learning by discussion, and be applied to new learning scenarios. This could provide significant benefits for students in the context of project-based collaboration learning, and in education in general.

In these environments, the discussion process forms an important social task where participants can think about the activity they are carry out, collaborate with each other exchanging ideas that may arise, propose new resolution mechanisms, and justify and refine their own contributions and thus acquire new knowledge.

To that 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 professor or the 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 participant contributions (mostly students) in which a proposal, idea or plan to get to the solution is brought up. The other participants can elaborate to this proposal according to the different types of communicative acts the proposal can be structured, such as questions, comments, explanations and agree/disagree statements. Finally, when a correct proposal of solution is achieved, the *consensus* contributions take part to approve it (this includes different consensus models such as voting); when a solution is accepted the discussion closes up.

These interesting features will be supported by allowing this application to take advantage of the CLPL components. For instance, the *CSCL Knowledge Management* component can provide full support to event management and thus, during the elaboration phase, a complete treatment of task performance events will enable the system to keep participants aware of the others' contributing behaviour in each type of communicative act. 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.

In a similar way, we may need to account for the personal features of the discussion group participants such as their role, collaboration preferences and so on that constitute the user and group model and let participants add new services whilst their needs are evolving as time goes by. All these user features are already included in the *CSCL User Management* component through the user profile management subsystem, providing a solid support for building and maintaining a user and group model.

Therefore, on the one hand, our application supports a complete discussion process through the realization of 3 generic contribution types and an open user and group model. On the other hand, it constitutes a valuable resource that, functioning under a Grid infrastructure, takes advantage of most of the benefits of Grid to greatly improve essential features of a discussion process such as the participant awareness of the others' contributions by a complete and highly cost log file treatment.

Indeed, developing such an application aims at:

- Going beyond just online discussions, by giving support to purposeful groups focused on advancing knowledge, elevating personal or organizational effectiveness, and generating tangible results.
- Providing a powerful tool for facilitating meaningful and productive interaction among group members.
- Contributing to the construction of enriched and useful member and group models.

7. Conclusions and further work

In this paper we have shown how CSCL applications can take advantage of the benefits provided by the Grid to enable both advanced support for distributed activities and the necessary functionalities in a collaborative learning experience. Moreover, based on our study of the main needs of any specific CSCL application, we proposed a common basis for all of them as a generic platform called CLPL to develop robust, reusable, integrated, useful and innovative specific CSCL applications and allowing a Grid infrastructure to be incorporated into it. Finally, a CLPL application that validates the use of our platform and offers an innovative resource to be distributed to multiple Grid nodes was described; to that end, we plan to use the application in real collaborative environments to gain experience with it.

Furthermore, we would like to study the possibility to add a P2P architecture to a Grid infrastructure so as, on the one hand, to improve even more the Grid potential of robustness and resource availability and on the other hand, to allow participants to include computers at home, schools and businesses, and to scale to several millions of concurrent participants. The inherent decentralization of P2P systems provides interesting social benefits [16] that we think the CSCL environments could fully take advantage of, such as not to depend on exclusive information, decision capacity, or central control as well as to support auto-organized groups, spontaneous groups and so on.

Acknowledgments

This work has been partially supported by the Spanish MCYT project TIC2002-04258-C03-03.

8. REFERENCES

1. Dillenbourg, P.: *Collaborative Learning: Cognitive and Computational Approaches*. Elsevier Science, Oxford, UK (1999).
2. Daradoumis T., Xhafa, F. and Marquès J.M: A Methodological Framework for Project-based Collaborative Learning in a Networked Environment. *Internat. Journal of Continuing Engineering Education and Life-long Learning*. Special Issue on "CL in Networked Environments". Inderscience (Eds). Geneve, Switzerland. Vol. 12, Nos. 5/6, pp.389-402, 2002.
3. Roschelle, J., DiGiano, C., Koutlis, M., Repenning, A., Phillips, J., Jackiw, N., and Suthers, D: Developing Educational Software Components. Computer. 32 (9) 50-58, 1999.
4. Dimitriadis, Y. A., Asensio, J. I., Toquero, J., Estébanez, L., Martín, T.A., and Martínez, A.: Towards a component based system for the CSCL domain (in Spanish). In: Proc. of the Symposium on Informatics and Telecommunications, Sevilla, Spain, 2002.
5. Amor, M., Fuentes, L., Jiménez D., and Pinto, M. Adaptability in virtual collaborative environments: A Component and Aspect based approach (in Spanish). In: Proc. of the Spanish WS on Group Work and CL: Experiences and Perspectives. November 11, San Sebastián, pp. 21-30, 2003.
6. Foster, I. Computational grids. In: The Grid: blueprint for a future computing infrastructure, eds. Foster, I. and Kesselman, C. San Francisco, CA, USA: Morgan Kaufmann, pp. 15-52, 1998.
7. Amin, K., Nijsure, S., and von Laszevski, G.: Open Collaborative Grid Services Architecture (OCGSA), In: Proc. of the W3C EuroWeb 2002 Conference, Oxford, UK, 2002.
8. Bote-Lorenzo, M. L., Dimitriadis, Y. A., Gómez-Sánchez, E.: Grid Characteristics and Uses: a Grid Definition. Proc. of the 1st European Across Grids Conference (CD), Santiago de Compostela, Spain, 2003.
9. Ana P.T. Bacelo Blois, Karin Becker: A Component-Based Architechture to Support Collaborative Application Design, In Haake and J. Pino Eds., Groupware: Design, Implementation and Use. LNCS, Vol. 2440, pp.134-143
10. The Access Grid Collaboration Environment. http://www.syllabus.com/
11. Sergio F. Ochoa, Luis A. Guerrero, David A. Fuller, and Oriel Herrera: Designing the Communications Infrastructure of Groupware Systems. In Haake and J. Pino Eds., Groupware: Design, Implementation and Use. LNCS, Vol. 2440, pp.114-133.
12. Baloian, N., Berges, A., Buschmann, S., Gaßner, K., Hardings, J., Hoppe, U. and Luther, W.: Document Management in a Computer-Integrated Classroom. In Haake and J. Pino Eds., Groupware: Design, Implementation and Use. LNCS, Vol. 2440, pp.35-44.
13. Santi Caballé, Fatos Xhafa: A Study into the Feasibility of Generic Programming for the Construction of Complex Software. 5th YRWS. In: Proc. of the GPCE/NODe'03 pp.441-446, Erfurt, Germany, 2003.
14. Litwin, W., Neimat: a High Performance Multi-attribute Scalable Distributed Data Structure. HPL-DTD & Paris 9 Technical Report, 1994.
15. The Globus Alliance. http://www.globus.org/
16. Hari Balakrishnan, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica: Technical and social components of peer-to-peer computing: Looking up data in P2P systems. Communications of the ACM, Volume 46, Issue 2. February 2003.