A Grid-Based Approach for Processing Group Activity Log Files

Fatos Xhafa[1], Santi Caballé[2], Thanasis Daradoumis[2] and Nan Zhou[3]

[1]Dept. of Languages and Informatics Systems, Polytechnic University of Catalonia
Jordi Girona Salgado 1-3, 08034 Barcelona, Spain
fatos@lsi.upc.es
[2]Open University of Catalonia, Department of Information Sciences
Av. Tibidabo, 39-43, 08035 Barcelona, Spain
{scaballe,adaradoumis}@uoc.edu
[3]College of Information Science & Technology, Drexel University
3141 Chestnut Street, Philadelphia, PA 19104-2875
nan.zhou@cis.drexel.edu

Abstract. The information collected regarding group activity in a collaborative learning environment requires classifying, structuring and processing. The aim is to process this information in order to extract, reveal and provide students and tutors with valuable knowledge, awareness and feedback in order to successfully perform the collaborative learning activity. However, the large amount of information generated during online group activity may be time-consuming to process and, hence, can hinder the real-time delivery of the information. In this study we show how a Grid-based paradigm can be used to effectively process and present the information regarding group activity gathered in the log files under a collaborative environment. The computational power of the Grid makes it possible to process a huge amount of event information, compute statistical results and present them, when needed, to the members of the online group and the tutors, who are geographically distributed.

1 Introduction

In the online collaborative learning teams, monitoring, awareness and feedback during the group activity are key factors in determining group functioning and task performance and, hence, the success of the learning outcome. Indeed, it is crucial to keep the group members informed of the progress of their peers in performing the learning exercise both as individuals and as a group. It is also important for group members to be aware of the extent to which other members are participating in the collaborative process as this will influence their decision making [1]. Collaborative learning also involves a tutor, who is responsible for acquiring information about students' problem-solving behavior, group processing [2] and performance analysis [3]. To this end, researchers have tried to provide learning teams and tutors with tools and approaches that facilitate monitoring and providing awareness and feedback to support the group activity. Such approaches [4], [5] usually rely on processing group activity data from different sources.

Xhafa, F., Caballé, S., Daradoumis, T., Zhou, N. A grid-based approach for processing group activity log files. A: On The Move Federated Conferences & Workshops. "On the Move to Meaningful Internet Systems 2004: OTM 2004 Workshops: OTM Confederated International Workshops and Posters, GADA, JTRES, MIOS, WORM, WOSE, PhDS, and INTEROP 2004: Agia Napa, Cyprus, October 25-29, 2004: proceedings". Berlin: Springer, 2004, p. 175-186.
The final authenticated version is available online at https://doi.org/10.1007/978-3-540-30470-8_35

Computer Supported Collaborative Learning (CSCL) applications are characterized by a high degree of user-user and user-system interaction and hence generate a huge amount of information usually maintained in the form of event information. In order to make this information useful to the group activity, it must be appropriately collected, classified and structured for later automatic processing by computers as part of a process of embedding information and knowledge into CSCL applications (Fig. 1). The aim is to extract essential knowledge about the collaboration and to make it available to users as awareness and feedback.

Asynchronous collaboration is an important source of group activity data. Data collected during the online collaborative learning activity is then classified and structured into group activity data log files. In order to constantly provide group participants with as much awareness and feedback as possible, it is necessary to efficiently process these log files so that the extracted data can be used for computing statistical results, which can be presented to group members whenever needed.

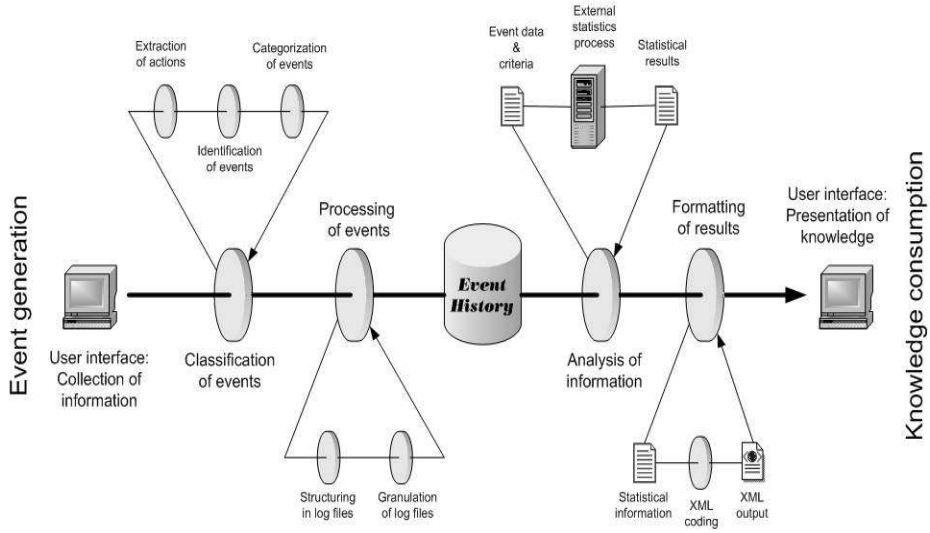

Fig. 1. The process of embedding information and knowledge into CSCL applications

Due to the huge amount of event information gathered in data log files, the processing requires considerable computational resources beyond those of a single computer. In order to make the extracted information available in real time it is necessary to reduce the computational time to acceptable levels. As there may be hundreds or even thousands of students distributed in teams as well as many tutors conducting the learning exercises, the amount of data produced will place great demands on the computational resources given that, in such situations, we need to process the information related to particular classrooms, specific teams within a classroom and even to certain phases of the learning exercise. Our experience at the Open University of Catalonia [6] has shown the need to monitor and evaluate real, long-term, complex, collaborative problem-solving situations through data-intensive applications that provide efficient data access, management and analysis. The Virtual Math Teams

(VMT) Project[1] at Drexel University [7] aims to develop the first application of digital libraries to small group collaborative learning, which requires the processing of a large volume of collaborative activity log files.

The lack of sufficient computational resources is the main obstacle to processing data log files in real time and in real situations this processing tends to be done later, which as it takes place after the completion of the learning activity has less impact on it. With the emerging Grid technology such a handicap can be overcome by using its computational power. The concept of a computational Grid [8] has emerged as a way of capturing the vision of a network computing system that provides broad access not only to massive information resources, but to massive computational resources as well. There is currently a lot of research being conducted on how to use the Grid computing paradigm for complex problem solving [9], processing huge amount of data in biology and medicine, simulations, and collaborative systems. For such problems, putting together distributed computing and storage resources is clearly of great value. Moreover, different technologies such as Globus [10], MPI-Grid2 [11], Condor-G [12], NetSolve [13] and frameworks such as Master-Worker Framework on computational Grid [14] as well as infrastructures for data-intensive Grid applications [15] have been proposed to support the development of Grid-based applications.

In this paper we propose a Grid-based approach for processing group activity log files in order to make the processed information available to the group members in an efficient manner, to compute statistical results and to present the results to the group members and tutors, who are in different locations, as a means of facilitating the group activity, decision making, task accomplishment, and assessment of the progress of the group etc. Our starting point is the definition of an appropriate structure for the log files designed as a part of a more generic platform for supporting CSCL applications [17]. The purpose is to define the structure of event information to be stored in order to permit the structuring of the event information in log files of different degrees of granularity, e.g. corresponding to a single group or to a whole classroom made up of several groups and/or for a given period of time in the group activity. Later, we show how to use Grid infrastructure through the Master-Worker paradigm for processing the log files resulting in a database ready to be used for statistical computations. Furthermore, the natural parallelism inherent in our data log files, as well as in our analysis procedures, makes it feasible to use distributed resources efficiently. Indeed, a Grid computing environment includes computing and storage resources with diverse capabilities and hence different degrees of granularities in our log files, allowing an efficient use of available resources.

The rest of the paper is organized as follows. We give in Section 2 the context that motivated this research and make reference to other studies in the field. In Section 3 we show the structure of data log file used for gathering the event information generated during group activity. In Section 4 we present the Master-Worker approach for processing log files using the Grid infrastructure and, finally, in Section 5 we draw conclusions and outline ongoing work.

[1] For more information, see http://www.cis.drexel.edu/faculty/gerry/vmt/index.html

2 Context and Related Work

Our real context refers to group activity at the Open University of Catalonia (Spain) and the Math Forum at Drexel University (USA). The former results from applying the Project-Based Collaborative Learning paradigm to model several online courses, such as "Software Development Techniques". These courses involve hundreds of students, a dozen of tutors and are characterized by intensive collaboration activity due to the complexity of the learning practices.

To implement the collaborative learning activities and capture the group interaction we use the Basic Support for Cooperative Work (BSCW) system, a groupware tool that enables asynchronous collaboration over the web [16]. BSCW records the interaction data into log files, which can be used for interaction analysis and knowledge extraction. However, its centralized architecture does not allow data access, management and the analysis of BSCW log files.

In particular, BSCW does not incorporate functionalities to process the log files nor provides the means to calculate and present statistics results. Moreover, BSCW generates a unique log file at the end of the day, which includes a large volume of data describing the activity of all virtual groups. Given that log files do not classify or structure information in any way, there is no possibility of scaling them up. As a result of this there is no way to access data related to separated workspaces, specific groups, or phases of the learning practice.

In recent years the popularity of distant collaborative learning among students has increased enormously. This semester, for example, we have had more than 500 students distributed in more than 100 virtual groups composed of 4 to 6 members. All the groups worked, mainly asynchronously, during 4 months. Due to the large volume of interaction data generated, the wide geographical distribution of the students, and the limitations of BSCW, a Grid solution for data-intensive applications and data analysis becomes imperative to overcome the above-mentioned problems and provide a more effective service to our students and tutors.

Similarly, the VMT Project has been investigating how small groups of students meet online and solve mathematical problems collaboratively using Synergeia [18] (an extension of BSCW) and other similar systems. This study gives great importance to the processing of transaction logs of the collaboration activities, which is currently done manually. As the size of transaction logs increases, it will become even more necessary to develop a means for the automatic processing of data.

In the context of our research, Grid computing [19], [20] has been used to support the real-time requirements imposed by human perceptual capabilities as well as the wide range of many different interactions that can take place as one of the most challenging issues of collaborative computing support. For instance, Grid computing offers high-throughput and data-intensive computing [17], which greatly facilitate the process of embedding information and knowledge into CSCL applications making it possible to provide users with real-time awareness and constant feedback. In the literature, however, there has been, to the best of our knowledge, little study aimed at achieving these objectives. As an initial approach, the OCGSA framework [21] proposes an event archiving service, which logs the messages or events communicated between online users of a group instance into a persistent database. However, in pro-

posing the implementation of the functionality, this framework does not offer any methodology which takes advantage of the distributed nature of Grid computing to partition the generated event information for efficient parallel processing.

3 The Structure of Group Activity Log Files

In collaborative learning systems, usual group activity results in a lot of interaction which generates a huge amount of events. Therefore, CSCL applications have to be designed to permit the pre-structuring, classification and partitioning of these large amounts of event information into multiple log files to meet different criteria (e.g. group or time) in order to correctly capture the group activity and increase the efficiency of data processing.

The existing CSCL applications have several drawbacks in structuring the log files that prevent efficient processing. To overcome this, we firstly propose a definition and classification of event information generated in a CSCL system and, secondly, we explain how to store this information in log files according to different criteria with the aim of facilitating its later processing in a Grid infrastructure (see also Fig. 1).

3.1 Definition and Classification of Event Information

The most important issue while monitoring group activity in CSCL applications is the collection and storage of a large amount of event information generated by the high degree of interaction among the group participants. Such a large amount of informational data may need a long time to be processed. Therefore, collaborative learning systems have to be designed in a way that pre-structures and classifies information in order, on the one hand, to correctly measure the group activity and, on the other hand, to increase the efficiency during data processing in terms of analysis techniques and interpretations.

The classification of information in CSCL environments is achieved by distinguishing three generic group activity parameters: task performance (i.e. collaborative learning product), group functioning and scaffolding [17]. Furthermore, in a collaborative learning experience, the group activity is driven by the actions of the participants on the collaborative learning resources, which are aggregated to the user events to form another taxonomy in which we can differentiate, at a high level of abstraction, between active, passive and support user actions (see Fig. 2). Therefore, in CSCL applications there is a strong need for the classification of all types of events generated by user actions according to the three generic parameters mentioned. To this end, a complete and tight hierarchy of events (Fig. 2) is provided to collect and categorize the identified events generated by user actions during the collaborative learning activity.

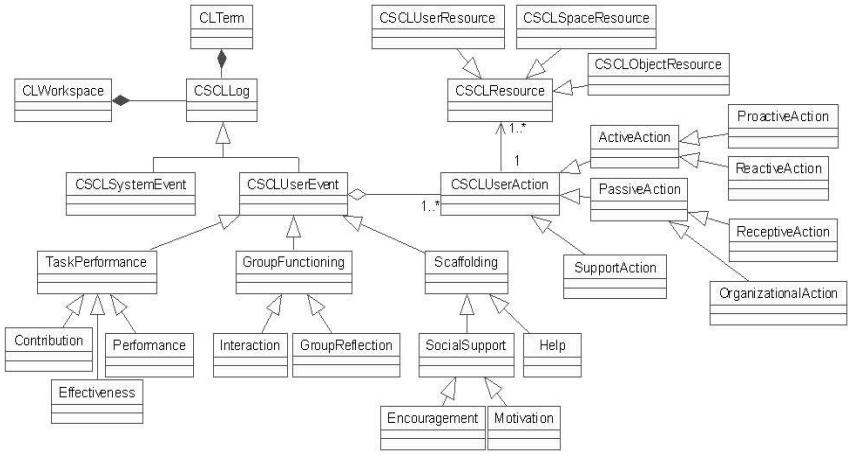

Fig. 2. A hierarchy to collect and classify all events generated during the group activity

In order to implement this classification, we developed a generic, reusable, component-based library for the construction of specific CSCL applications [17] in which a specific component called *CSCL Knowledge Management*[2] was designed representing the formalization of this hierarchy of events. Note that *CLWorkspace* in Fig. 2 refers to the log file aggregating event information that is generated in a given workspace. Such a workspace may correspond to a whole group or to a phase within a group activity.

3.2 The Structure of the Log Files

In order to prepare the event information for efficient processing, as soon as we classified and turned it into persistent data, we store it in the system as log files, which will contain all the information collected in specified fields. Next, we intend to predefine two generic types of log files according to the two basic criteria, time and workspace, that characterize group collaboration. These log files will represent as great a degree of granularity as possible regarding both criteria and they will be parameterized so that the administrator can set them up in accordance with the specific analysis needs. Thus, the finest grain or the smallest log file should be set up to store all events occurring in each group for the shortest time interval. Therefore, every single workspace will have its own log file made up of all the events occurring within the workspace for a given period of time.

During data processing it will be possible to concatenate several log files so as to obtain the appropriate degree of granularity thus making it possible for a distributed system to efficiently parallelize the data processing according to the characteristics of

[2] CSCL Knowledge Management component is found at: http://cv.uoc.edu/~scaballe/clpl/api/

the computational resources. The aim is to efficiently process large amounts of information enabling the constant presentation of real-time awareness and constant feedback to users during the group activity.

Thus, concatenating several log files and processing them in a parallel way, it would be possible to constantly show each group member's absolute and relative amount of contribution, which would provide participants with essential feedback about the contribution of others as a quantitative parameter supporting the production function. In a similar way, real-time awareness is possible by continuously parallelizing and processing each and every single fine-grained log file of each workspace involved at the same time in order to permanently notify all workspace members of what is going on in their groups. Finally, showing the results of complex statistics after longer periods of time (e.g. at 12 hour intervals) is very important for the group's tutor to be able to monitor and assess the group activity as a qualitative parameter supporting acquisition of information about students' problem-solving behavior, group processing and performance analysis.

4 A Master-Worker Approach for Processing Log Files

The Master-Worker (MW) model (also known as Master-Slave or Task Farming model) has been widely used for developing parallel applications. In the MW model there are two distinct types of processors: master and workers. The master processor performs the control and coordination and assigns tasks to the workers. It also decides what data will be sent to the workers. The workers typically perform most of the computational work. The MW model has proved to be efficient in developing applications using different degrees of granularity of parallelism. Indeed, it has several advantages such as flexibility and scalability (the worker processors can be implemented in many different ways and they can be easily added if needed) as well as separation of concerns (the master performs coordination tasks and the worker processors carry out specific tasks). This paradigm is particularly useful when the definition of the tasks to be completed by the workers can be done easily and the communication load between the master and workers is low.

4.1 Master-Worker Paradigm on the Computational Grid

The MW paradigm has been used in developing parallel applications in traditional supercomputing environments such as parallel machines and clusters of machines. Over the last few years, Grid computing has become a real alternative for developing parallel applications that employ its great computational power. However, due to the complexity of the Computational Grid, the difficulty encountered in developing parallel applications is higher than in traditional parallel computing environments. Thus, in order to simplify the development of Grid-aware applications several high-level programming frameworks have been proposed, among which is the Master-Worker Framework (MWF) [14].

MWF allows users to easily parallelize scientific computations through the master-worker paradigm on the computational grid. On the one hand, MWF provides a top level interface that helps the programming tasks to distribute large computations in a Grid computing environment; on the other hand, it offers a bottom level interface to existing grid computing toolkits, for instance, using the Condor system to provide Grid services. The target applications of MWF are parallel applications with weak synchronization and reasonably large granularity. As we show next, this framework is appropriate for processing log files of group activity since we have different degrees of granularity available so as to guarantee efficiency and, furthermore, there is no need for synchronization or communication between the worker processors. Moreover, in our application, the communication load between the master and workers is very low.

4.2 The Architecture of the Application

The architecture of the application (Fig. 3) is made up of three parts: (1) the Collaborative Learning Application Server, which is in charge of maintaining the log files and storing them in specified locations; (2) the MW application for processing log files and, (3) the application that uses the resulting information in the data bases to compute statistical results and present them to the final user.

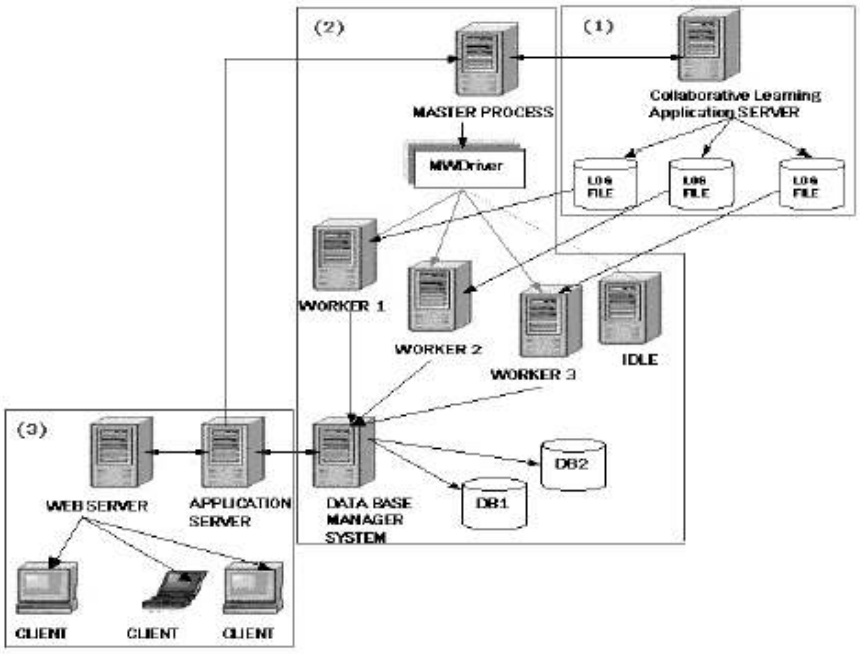

Fig. 3. The architecture of the application for processing log files

The Master-Worker Application for Processing Log Files. We proceed now to present more details of the MW application, basically how the master and worker processors are programmed. The master is in charge of generating new tasks and submitting them to the MWDriver for distributing them to the worker processors while the worker processors run in a simple cycle: receiving the message describing the task from the master, processing the task according to a specified routine and sending the result back to the master. The MW framework, which schedules the tasks, manages the lists of workers and of tasks to be performed by the MWDriver. Tasks are assigned to workers by giving the first task on the list to the first idle worker on the worker list. We take advantage of the fact that the MWDriver's interface allows the task list to be ordered according to a user's criteria and the list of workers to be ordered according to their computational power. Thus, we order the task list in decreasing order of log file size and the machines in decreasing order of processing capacity so that "good" machines have priority in receiving the largest log files.

Furthermore, we have a unique type of task to be performed by the workers that consists in processing a log file. We assume that the workers have the processing routine available; otherwise, the worker would take a copy of the routine on receiving a task for the first time and then use a flag to indicate whether it must receive a copy of the routine or not. The task is described as follows:

Task description:

```
address of the location of the log file;
name of the log file;
size of the log file;
address of the location where the processing routine is
 found.
url of the database where the processed information
 will be stored;
```

The master processor is programmed as follows:

```
while (true) do
  check for new log files generated from the
    Collaborative Learning Application Server;
  update the list of the <log file description>
    for the new incoming log files;
  for each new log file generate a task;
  submit the newly generated tasks to the MWDriver;
```

We note that the log files generated by the Collaborative Learning Application Server can be stored either in disk spaces of the same server or at different locations (machines) available in the Grid. Furthermore, the processed information by the workers can be stored either in unique or different databases that can be found at different machines as specified in the tasks to be realized by the worker processors.

The worker processor is programmed as follows:

```
receive the task;
receive the specified log file from the specified
 location in the task description;
run the processing routine on the log file
send to the master the task's statistics (execution
 time, number of events processed...) upon
 completion of the task;
```

Efficiency issues of the MW Application. It should be observed that the communication takes place between master and the workers at the beginning and the end of the processing of each task. Therefore, our application has weak synchronization between the master and the workers, which ensures that it can run without loss of performance in a Grid environment. Moreover, the number of workers can be adapted dynamically so that if new resources appear they can be incorporated as new workers in the application; in addition, if a worker in the Grid becomes unavailable while processing a task, the task can be reallocated to another worker. Finally, by having different degrees of granularity of the log files it is possible to efficiently distribute the load balance among workers and minimize the transmission of the data log files from their original locations to the worker machine.

4.3 The Design of the Resulting Database

Once the event information from the log files has been processed, the workers (see Fig. 3) send back the task reports (e.g. processing time, number of processed events, etc.) to the collaborative learning application server through the master so as to verify the results achieved. The results of data processing, which workers send to the database manager system, should have correctly represented all the information contained in the log files so as to make it possible to consult both the desired data from the database directly (e.g. number of connected users, type of documents in a certain workspace, etc.) and the computed complex statistical results from the database. These statistical results should be obtained by the application server as fast as possible and presented to group members and tutors in different formats.

Thus, based on the premises argued in [22], we provide[3] a logic design of the database, which is generic, efficient and independent from any specific database manager. We have designed the database in a way to satisfy all of these requirements and to allow users to consult data regarding the basic entities that take place in any CSCL environment (users, objects, workspaces, connections, etc.).

4.4 XML representation of the statistical results

The third part of our application uses the resulting information in the databases to compute statistical results and present them to the members of the online collabora-

[3] See the design of the database at: http://cv.uoc.edu/~scaballe/GADA04/DBDesign.pdf

tive group and the tutors In this context we are studying an XML coding of the statistical results in order to make it possible to present this information to final users in different forms (see Fig. 1). Considering the fact that the data is highly structured and the design of the relational database [23], we propose that application be designed as a middleware [24], which performs the following functions: to extract necessary information from the databases, to compute statistical measurements as desired, and to convert the results into XML output. This design will provide sufficient flexibility as to allow ad hoc statistical measurements to be obtained as well as permitting the creation of user-specified document type definitions (DTD) to accommodate the different needs of information representation.

5 Conclusions and Ongoing Work

The efficient embedding of information and knowledge about the ongoing group activity into collaborative learning environments is crucial to the success of the online collaborative learning activity. Moreover, disposing of such information as fast as possible makes the use of more computational resources indispensable in order to process a huge amount of event information generated during the ongoing group activity. In this study we have shown a Grid-aware approach for processing log files of group activity in an efficient yet simple manner. Our approach is based on the Master-Worker paradigm on the Computational Grid and shows its feasibility by satisfying several conditions such as weak synchronization, dynamic adaptation of resources, efficient load balancing etc., which ensures that our application can run without loss of performance in a Grid environment. We have achieved this through a careful definition of the event information generated during group activity and an adequate structure for log files that collect the event information. This allows us to dispose of log files according to different criteria as well as different degrees of granularity.

We are currently implementing this application that we will test under a real environment using a grid infrastructure formed by machines at three Spanish Universities (Open University of Catalonia, Polytechnic University of Catalonia and University of Valladolid) within the scope of a joint Spanish Research Project (CICYT). Doing so, we are using real data coming from workspaces of online collaborative learning groups at the Open University of Catalonia and those from the VMT Project at Drexel University.

Acknowledgements. This work has been partially supported by the Spanish MCYT project TIC2002-04258-C03-03 and NSF VMT project (IERI grant #0325447).

References

1. Dillenbourg, P. (ed.) (1999): Collaborative Learning. Cognitive and Computational Approaches. Elsevier Science Ltd. 1-19.

2. Kiesler, S. and Sproull, L.S. (Eds.) (1987). Computing and change on campus. New York: Cambridge Press
3. Daradoumis, T., Xhafa, F. and Marquès, J.M. (2003) Exploring Interaction Behaviour and Performance of Online Collaborative Learning Teams, 9th Int. Workshop on Groupware, CRIWG'03, France. Lecture Notes in Computer Science, Vol. 2806, pp. 126-134.
4. Martínez, A., de la Fuente, P., Dimitriadis, Y. (2003) Towards an XML-Based Representation of Collaborative Action. In: Proc. of the CSCL 2003, Bergen, Norway.
5. 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
6. Open University of Catalonia http://www.uoc.edu (web page as of August 2004)
7. Virtual Math Team Project http://mathforum.org (web page as of July 2004)
8. Foster, I. and Kesselman, C. (Eds) (1999) The Grid 2e, 2nd Edition Blueprint for a New Computing Infrastructure. Morgan-Kaufman
9. Arnold, D., Agrawal, S., Blackford, S., Dongarra, J., Miller, M., Seymour, K., Sagi, K., Shi, Z. and Vadhiyar, S. (2002) Users' Guide to NetSolve V 1.4.1, Univ. of Tennessee, Technical Report, ICL-UT-02-05.
10. Globus: http://www.globus.org (web page as of July 2004)
11. MPICH-G2: A Grid-enabled MPI. http://www3.niu.edu/mpi/ (web page as of July 2004)
12. Condor-G: http://www.cs.wisc.edu/condor/condorg/ (web page as of July 2004)
13. Casanova, H. and Dongarra, J. (1998): NetSolve: Network enabled solvers, IEEE Computational Science and Engineering, 5(3) pp. 57-67.
14. Goux, J.P., Kulkarni, S., Linderoth, J. and Yoder, M. (2000): An enabling framework for master-worker applications on the computational grid. In 9th IEEE International Symposium on High Performance Distributed Computing (HPDC'00). IEEE Computer Society.
15. Pérez, M., Carretero, J., García, F., Peña, J.M., and Robles, V. (2003) MAPFS: A Flexible Infrastructure for Data-Intensive Grid Applications, In 2003 Annual CrossGrid Project Workshop and 1st European Across Grids Conference, Santiago de Compostela, Spain
16. Bentley, R., Appelt, W., Busbach. U., Hinrichs, E., Kerr, D., Sikkel, S., Trevor, J. and Woetzel, G. (1997) Basic Support for Cooperative Work on the World Wide Web. Int. J. of Human-Computer Studies 46(6) 827-846.
17. Caballé S., Xhafa, F., Daradoumis, T. and Marquès, J.M. (2004) Towards a Generic Platform for Developing CSCL Applications Using Grid Infrastructure. In: Proc. of the CLAG/CCGRID'04, Chicago, USA
18. Stahl, G. (2002) Groupware Goes to School, 8th Int. Workshop on Groupware. CRIWG'02, La Sirena Chile. LNCS, Vol. 2440, pp. 7-24. ISBN: 3-540-44112-3.
19. Foster, I. and Kesselman, C. The Grid: Blueprint for a Future Computing Infrastructure. pp. 15-52. Morgan Kaufmann, San Francisco, CA, 1998.
20. Bote-Lorenzo, M. L., Dimitriadis, Y. A., Gómez-Sánchez, E.: Grid Characteristics and Uses: a Grid Definition. In: Proc. of the 1st European Across Grids Conference (CD), Santiago de Compostela, Spain, 2003.
21. 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.
22. Watson, P. (2003) Databases and the Grid In: Grid Computing: Making The Global Infrastructure a Reality, Berman, F. et al. (eds.) Wiley
23. Fernández, M., Kadiyska, Y., Suciu, D., Morishima, A., and Tan, W. (2002) SilkRoute: A framework for publishing relational data in XML. In ACM Transactions on Database Systems (TODS), Vol. 27, Issue 4.
24. Kyung-Soo, J. A design of middleware components for the connection between XML and RDB. Proceedings of IEEE ISIE 2001, Vol. 3, pp. 1753 - 1756.