HAL
open science

A Grid Approach to Provide Effective Awareness to On-line Collaborative Learning Teams

Santi Caballé, Thanasis Daradoumis, Claudi Paniagua, Fatos Xhafa

To cite this version:

Santi Caballé, Thanasis Daradoumis, Claudi Paniagua, Fatos Xhafa. A Grid Approach to Provide Effective Awareness to On-line Collaborative Learning Teams. 1st International Kaleidoscope Learning Grid SIG Workshop on Distributed e-Learning Environments, Hotel Oriente, Vico Equense (Naples), Italy. 14 March 2005, 2005, Vico Equense, Italy. pp.9. hal-00190583

HAL Id: hal-00190583

https://telearn.hal.science/hal-00190583

Submitted on 23 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Grid Approach to Provide Effective Awareness to On-line Collaborative Learning Teams

S. Caballé, T. Daradoumis, C. Paniagua and F. Xhafa
Open University of Catalonia, Department of Information Sciences
Av. Tibidabo,39-43, 08035 Barcelona, Spain
{scaballe, adaradoumis, cpaniaguam, fxhafa}@uoc.edu

Providing collaborative learning applications the possibility to offer continuous awareness to on-line collaborative learning teams is a challenging issue since it is essential to keep participants informed of both what is happening and what happened in the learning group. Moreover, awareness is the basis to enhance collaboration and improve the individual and group performance since it allows to track and assess the collaborative learning process more objectively and to model students' behavior more efficiently. In order to provide students and tutors with effective awareness, we consider the collaborative actions registered in log files. It is necessary to process large and considerably complex event log files in a constant manner, and thus it may requires processing capacity beyond that of a single computer. To that end, in this paper we show how a Grid approach can considerably decrease the time of processing group activity log files and thus allow students and tutors to be aware of what is going on in the group activity even in real time. Our approach is based on the master-worker paradigm and is implemented using Globus technology running on the Planetlab platform. To test our application, we used event log files from the Basic Support for Collaborative Work (BSCW) system.

Computer-supported collaborative learning (CSCL), awareness, group activity log files, grid technology.

1. INTRODUCTION

Awareness [1] in Computer-Supported Collaborative Learning (CSCL) environments, is essential for any form of cooperation, namely coordination, communication and collaboration [2]. It allows implicit coordination of collaborative learning, opportunities for informal, spontaneous communication, and gives users partial feedback [3] about what is happening during collaboration. Indeed, it is crucial 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 [4].

When users communicate and collaborate with each other and also when they are coordinating their activity, they must always be aware of both what other users are doing at the same time and what other users did in the past. This implies receiving information from both synchronous and asynchronous modes and being aware of both the resources used and the users who interact with the system. In addition, the persistent storage of awareness information as *group memory* [5] is essential for both students and tutors since, on the one hand, it allows participants not to access only the latest documents and data, which are commonly stored for later retrieval, but also the context in which they were created. On the other hand, it allows tutors to track the collaborative learning process for several purposes such as scaffolding and assessment of the learning outcome. Thus, being aware of the activities of others is essential for supporting group activity and improving learning as it enhances the collaboration in great deal in terms of decision-making, group organization, social engagement, support, monitoring and so on.

When awareness is synchronous, users should be aware of the current activity in the group (the contribution of other members, their location and availability, the users working on a shared document at the same time and so on) and should 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). In an asynchronous context, on the other hand, users are aware of the activities performed by receiving deferred information of who, when, how and where shared resources have been created, changed or read by others as a basic requirement to solve the task at hand and support asynchronous communication.

The supply of efficient and transparent awareness to users in both synchronous and asynchronous modes is a significant challenge. Users are continuously interacting with the system (creating documents, reading others' contributions, etc.), generating events, which once collected, they are classified, processed, structured and analyzed. The information derived from this analysis will be represented in appropriate forms (messages, quantitative

information, visualizations, etc.) to support students' awareness, metacognition and self-regulation of their learning activity. This process is triggered by the system automatically and so is transparent to non-participating users.

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 type information. In order to capture the interaction for awareness purposes correctly, this event information can be classified into many different categories such as work sessions, messages, workspaces, documents and other objects and thus may generate large size of information, especially, in real online collaborative learning that comprise complex learning activities to be carried out during a rather long period of time and involve a considerable number of participants.

This information may include a great variety of types and formats and hence tends to be large in size [5]. Indeed, at a first level of classification, we can find group activity log files of CSCL applications associated with synchronous (e.g. multi-user editors) and asynchronous collaboration (e.g. discussion forums). These applications generate different types of information depending on their specific needs and functions (e.g. a discussion forum can generate event-type information so as to capture the participants' contributions). This information can be then stored in different formats.

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. As a result, there is a strong need for powerful tools that record the large volume of interaction data and can be used to perform an efficient interaction analysis and knowledge extraction.

Given the real needs of an online collaborative learning situation, in order to provide different types of awareness, we need to capture all and each type of possible data that could result to a huge amount of information that is generated and gathered in data log files. Moreover, the need to make the analyzed information available in real time entails that we may come across with processing requirements beyond those of a single computer. As a matter of fact, most of the existing approaches in the literature consider a sequential approach mainly due to three reasons: (i) processing for a specific purpose (i.e. limiting the quantity of information needed for that purpose); (ii) processing the information afterwards (i.e. not in real time) and (iii) processing of small data samples, usually for research and testing purposes (i.e. not for real learning needs). Yet, the lack of sufficient computational resources is the main obstacle for processing large amounts of data log files in real time. In real situations this processing tends to be done later, after the completion of the learning activity, thus having less impact on it [7].

Recently, Grid technology is increasingly being used to reduce the overall, censored time in processing data by taking advantage of its large computing support. The concept of a computational Grid [8] has emerged as a way of capturing the vision of a networked computing system that provides broad access not only to massive information resources, but to massive computational resources as well. Thus, in this paper, we show how a Grid approach can be used to match the time processing requirements.

A study was conducted to show that a Grid approach can increase the efficiency of processing a large amount of information from group activity log files. In this study, we use an *ad hoc* application called *EventExtractor* to process a sample of BSCW activity log files from both a sequential and Grid approach. This allows us to show first the gain provided by the Grid approach in terms of relative processing time and, second, the benefits of using the inherent scalable nature of Grid while the input log files are growing up in both number and large size. It is worth mentioning here that both approaches are independent from the type of log file since we encapsulate it in the *EventExtractor* and thus make it possible to carry out the Grid approach and conduct a comparative study.

The rest of the paper is organized as follows. In Section 2 we give the context that motivated this research and make some reference to other studies in the field. In Section 3 we show the process of creating awareness and exemplify a real situation. Section 4 provides a Grid-based approach to present both a sequential and parallel processing of event log files to be used by Grid nodes whereby Section 5 shows the most representative computational results achieved. We conclude in Section 6 by drawing some conclusions and outlining ongoing work.

2. CONTEXT AND RELATED WORK

In a real context, the Open University of Catalonia provides several on-line courses involving hundreds of under-graduated students and a dozen of tutors in a collaborative learning environment. The complexity of the learning practices entails intensive collaboration activity generating a great amount of group activity information. To implement the collaborative learning activities and capture the group interaction we use the Basic Support for Cooperative Work (BSCW) [9] as a shared workspace system which enables collaboration over the Web by supporting document upload, group management and event service among others features. BSCW event service provides awareness information to allow users to coordinate their work [10]. Events are triggered whenever a user performs an action in a workspace, such as uploading a new document, downloading (i.e. reading) an existing document, renaming a document and so on.

The system records the interaction data into large daily log files and presents the recent events to each user. In addition, users can request immediate email messages whenever an event occurs and the so-called daily activity reports which are sent to them daily and inform them about the events within the last 24 hours.
BSCW provides neither log file processing nor tools for analyzing the processed information, thus it can not provide enriched awareness to users to support collaboration and learning (e.g., model the users' behavior, and track and assess the collaborative learning process). Moreover, BSCW records the interaction as line-structured text files which are scarcely commented while they produce a high degree of redundancy. As a result, the processing and analysis of this huge amount of ill-structured information requires an efficient data processing system.

Research work in the Virtual Math Teams Project at the University of Drexel [11], has also raised the necessity for efficient processing of a large volume of collaborative activity log files resulting from collaborative learning activities in small groups. To this end, Synergeia [12] (an extension of BSCW) and other similar systems are used, however, they present similar drawbacks as BSCW.
There exist several approaches in the literature that deal with the processing and analysis of data for awareness purposes so that to keep track of the collaborative learning activities. Martinez et al [13] presented a tool for processing event logs of the BSCW using a social network analysis method. Other approaches, focus on processing event log files for analyzing the dialogue and interaction (Avouris et al. [14]), or for using them in query systems to increase awareness (Hardings [15]). On the one hand, these approaches consider the processing of the data just for a specific purpose, limiting thus the scope of the developed tools. On the other hand, they do not address the issue of processing time requirements that might result from the huge amount of data that are to be processed, which is a common issue in collaborative learning environments.

3. THE PROCESS OF CREATING AWARENESS

Creating useful, effective, heterogeneous, yet structured awareness is a complex process. As part of the process of embedding information and knowledge into CSCL applications (Figure 1) [7], this consists of four separate, indispensable stages: *collection of information*, *processing, analysis* and *presentation*. The entire process fails if one of these stages is omitted.

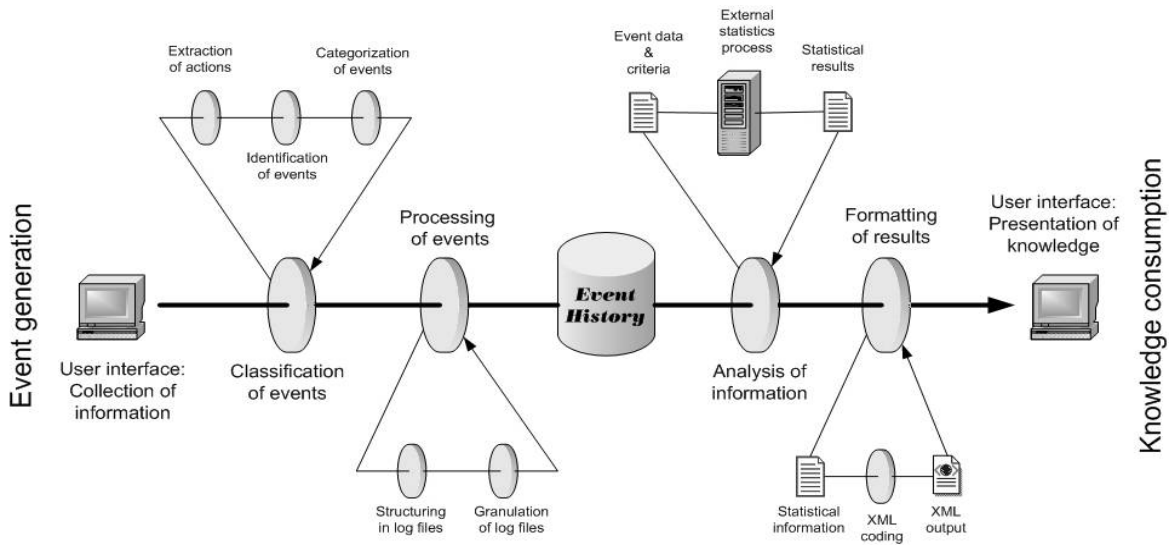

FIGURE 1: The process of embedding information and knowledge into CSCL applications

During the first stage, the information is collected from the collaborative actions triggered by the students and is classified in terms of specific events. Due to its conceptual importance, we now describe this stage in great detail. In particular, during the information collection, the objective is to gather as much information as possible generated during group activity to use it as a basis for the next three stages. To this end, we propose a solution consisting of classifying information by means of three generic group activity parameters [16], namely *collaborative learning product*, *group functioning*, and *scaffolding* (social support and task or group functioning help oriented services), which represent high-level collaborative learning processes.

The *collaborative learning product* parameter features the production function and task performance of on-line groups. It is characterized by the type of actions (events) that capture and describe the functional knowledge, cognitive processes and skills of the students and the group as a whole in solving problems and producing learning outcomes in a collaborative learning practice. It is used to analyze and evaluate the individual and group effectiveness as far as task achievement concerns. It can be measured as a qualitative and quantitative parameter by the type of user task-based actions that represent contributions which express basic and supporting active learning skills as well as perception skills.

The *group functioning* parameter is made up of the type of actions (events) that represent and are used to measure and analyze the individual and group effectiveness regarding participation and interaction behavior that facilitate the group's well-being function. As a quantitative parameter, it enables us to measure important participant contributions (in terms of specific types of user actions) which indicate skills related to: active or passive participation, well-balanced contributions and role playing, participation quality and communication flow among group members, as well as the necessary skills that facilitate and enhance group interaction, namely active processing skills (such as task, workspace and communication processing skills). In addition, interaction behavior can also be measured as a qualitative parameter by group reflection (i.e. group and individual self-evaluation).

The *scaffolding* parameter is specified by the type of events that refer to *social support* among members as well as to *task- or group functioning-oriented help* provided to a participant who is not quite able or ready to achieve a task on his or her own. As for the former, we look at event information that includes actions which support and promote group cohesion, such as motivational and emotional support, conflict resolution, etc. As for the latter, we focus on those specific actions designated to provide effective help to the peers when they need it during the collaborative learning activities. The participants' actions aiming at getting or providing help are classified and measured according to whether they refer to the task or group functioning.

In the next stage of *processing*, a tight structuring of the generated event information is performed and stored in log files. During the *analysis* stage, this information is analyzed and interpreted in order to extract knowledge according to different criteria. The final stage, *presentation*, is to show the students the knowledge obtained, thus providing them appropriate awareness and metacognition.

Therefore, in order to provide users with continuous awareness, once the event information generated in group activity is correctly collected and classified, this information needs to be structured in a way that facilitates its later processing and analysis and can be addressed in a distributed environment such as a Grid.

To that end, during the processing stage, we propose the following generic steps so as to structure the event information correctly for later processing [7]: we first classify the event information collected from the collaborative actions and turn it into persistent data; then we store it in the system as structured log files. Next, we predefine the structured files according to appropriate criteria (such as time and workspace) depending on the desired awareness information. For each of these criteria these files represent as great a degree of granularity as possible so that we can concatenate several log files and obtain the appropriate degree of granularity during data processing. This makes it possible for a distributed system such as Grid to parallelize data processing efficiently according to the characteristics of its computational resources.

Thereby, CSCL applications can take a great advantage of the inherent parallelism of a Grid environment to process several files (e.g. all the groups in a classroom) at the same time and thus considerably reduce the overall computational time. As a result, it is possible for these applications to process a large volume of collaboration activity data and make the results available in real time. The aim is to process large amounts of information efficiently enabling the constant presentation of both single awareness information in (almost) real-time and complex, in-depth deferred awareness.

At this point, in order to clarify the process of creating complex and useful awareness in a real situation, we briefly exemplify the case of monitoring the users' behavior by processing BSCW log file information involving time and workspaces. We take all the daily single BSCW log files as a starting point within the desired period of time that are to be analyzed. Each of these log files records all the event information generated each day and thus they collect all the users' activity in the system. This information is first classified according to users, time and workspaces so that it is only related to the user who generated events, the time when they occurred and the workspace where they took place. Then, the resulting log file is partitioned into multiple log files of a finer grain, each one storing all the workspaces that a certain user has visited during the shortest time interval. Consequently, several of these log files will be concatenated so as to obtain the appropriate degree of granularity and thus fit their size to the characteristics of the computational resources of Grid's nodes. The aim is to distribute the workload among the Grid's nodes correctly and as a result to parallelize the data processing efficiently. Finally, the processed results are stored in a database in a way that they can be easily analyzed by statistical packages so as to extract knowledge about the user's behavior: usual time to enter

and leave a certain workspace, time average staying in each workspace and number of times per day entering the same workspace are, among others, some analysis needs. At the end of the process, this obtained knowledge can be presented to the user as structured awareness.
The benefits of providing awareness for monitoring the students' behavior and performance are multiple and highly important as they can greatly enhance essential issues of collaboration and learning such as assessment, support, security, and so on. As an example, by monitoring the user's habits in the system it is possible to detect some irregular behavior of certain users and thus it can act as additional security layer to protect the system from unauthorized users.

4. A GRID APPROACH USING PLANETLAB

In order to perform our experiment, we developed a simple processing routine in Java, called *EventExtractor,* which runs on a single machine and extracts event information from the BSCW event log files. This application converts the event information into well-formatted data to be stored in a persistent support. Specifically, the *EventExtractor* first receives the BSCW log files as an input, then it extracts specific data about users, objects, sessions, etc. and finally it stores the extracted data in a persistent support as a data structure. It should be noted that the processing is done with the aim of extracting the whole information on events and without any specific criteria for later analysis; that is, processing is independent of the analysis methods that could be later applied to the extracted information. However, when executing sequentially, the *EventExtractor* needs a lot of time to process such amount of event information and hence it is not possible to constantly process a large size of data log files in real time.
In this section, on the one hand, we use this processing routine in a Grid prototype, namely, we run it in Grid nodes in a parallel manner; on the other hand, we measure the processing time of the sequential approach when executing this processing routine in a sequential mode by Grid nodes. This later result is used to compare the time results of both the sequential and parallel processing of event log files.
The Master-Worker (MW) [17] 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 decomposes the problem in tasks and assigns them to the workers taking into account the dependencies between them. The workers typically perform most of the computational work by just executing those tasks. The MW model has proved to be efficient in developing applications with different degrees of granularity of parallelism. This paradigm is particularly useful when the partitioning of the problem in tasks to be completed by the workers can be done easily and the dependencies between these tasks are low.
In this point, we show how the MW paradigm is appropriate for processing log files of group activity in a Grid environment, since we have different degrees of granularity available and, furthermore, there is no need for synchronization between the worker processors as tasks are completely independent from one another. To this end, we have written a minimal Grid implementation prototype using the Globus Toolkit 3 and have deployed it on the Planetlab platform. The aim is to demonstrate the viability of a Grid implementation of the MW paradigm for our problem domain. Both GlobusToolkit 3 and Planetlab are first described here briefly. Then, we present our application in more detail.
The Globus Toolkit (GT) [18] is the actual *de facto* Grid middleware standard. Version 3 of GT (GT3) is a refactoring of version 2 in which every functionality is exposed to the world via a Grid service (i.e. basically, stateful web services). The core of the GT is a Grid service container implemented in Java that leverages and extends the Apache's AXIS [19] web services container.
Planetlab [20] is an open platform for developing, deploying and accessing planetary-scale services. It is, at the time of this writing, composed up of 506 nodes hosted in 245 different sites. Each Planetlab node runs the same base software, basically a modified Linux operating system offering services to create virtual isolated partitions in the node, which look to users as the real machine.
In order to test our Grid prototype, we have turned Planetlab into a Grid fabric by installing the GT3's Grid service container. Moreover, we have implemented the worker as a simple Grid service that we have deployed on the GT3's container and have written a simple Java client that plays the role of the master by dispatching tasks just by calling the operations exposed by the worker Grid services, as follows:

- the **worker** Grid service publishes an interface with only one operation that the master calls in order to dispatch a task to the worker. This operation, which is implemented by wrapping the Java code of the *EventExtractor* routine, passes as an input a textual representation of the events to be processed by that task and returns a data structure containing performance information about the task executed (i.e. elapsed time, number of events processed and number of bytes processed).

- the **master** is just a simple Java application that reads from a configuration file (1) the folder that contains the event log files to process, (2) the available workers, (3) the number of workers to use, and (4) the size of the task to be dispatched to each worker expressed in number of events. The master then proceeds as follows: it picks as much workers as needed from the configuration file and puts them all in a queue of idle workers. Then it enters a loop reading the events from the event log files and, each time it has read a number of events, it either waits for a worker if the queue is empty or calls the worker's operation. Once the call to the worker returns, the worker is put back into the queue of idle workers. The master exits the loop when all events in the event log files have been read and all the tasks that were dispatched have finalized.

As we have stated, this is not a real GT3 Grid implementation of the MW paradigm but a proof-of-concept prototype, thus important features in a real environment such as fault-tolerance and dynamic discovery of available workers, are missing.

5. EXPERIMENTAL RESULTS

In order to carry out a comparative study between the sequential and Grid approaches, we designed a specific test battery in which we used both large amounts of event information and well-stratified short samples. Thus, on the one hand, in order to carry out certain tests we used all the existing daily log files making up the whole group activity generated during the spring term of 2004 in the course "Software Development Techniques" at the Open University of Catalonia. This course involved two classrooms, with a total of 140 students arranged in groups of 5 students and 2 tutors. On the other hand, other tests involved a few log files with selected file size and event complexity forming a sample of each representative stratum. This allowed us to obtain reliable statistical results using an input data size easy to use.

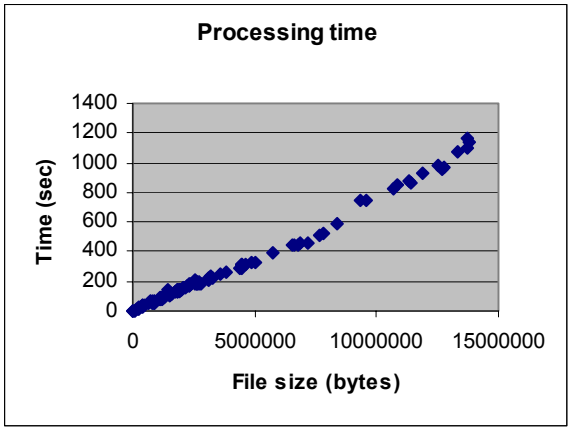

FIGURE 2: Sequential processing time for each event log file size

All our test battery was processed by the *EventExtractor* routine executed in our Grid prototype on single-processor nodes involving usual configurations. The battery test was executed several times with different work load in order to have more reliable results in terms of statistical data involving file size, number of events processed and execution time along with other basic statistics.

The experimental results from the sequential processing are summarized in Figure 2 which presents the processing results of over one hundred event log files involving file size and processing time. This shows that the processing time is linear with respect to the number of events processed.

This linearity found in processing time allowed us to greatly simplify the experiment by using the same event log file as input for all the Grid tests in the experiment. Then, we left to vary the parameters regarding both the number of workers and the size of the tasks (expressed in number of events) which were to be executed by the workers. We run tests for a different number of workers with different task sizes. Figure 3 shows the maximum speed-ups for the observed bandwidth between our master processor and the Planetlab nodes at the time of running the experiment and for the different number of workers we tested.

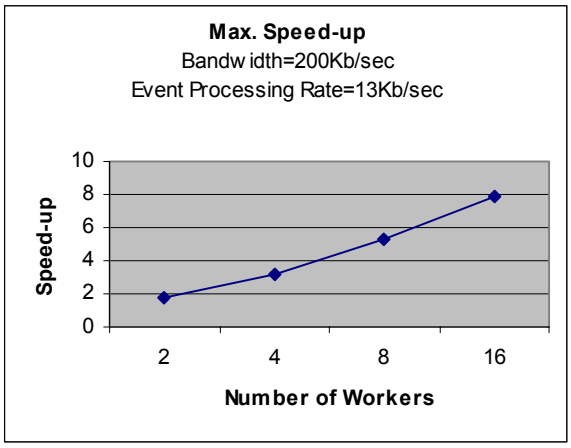

FIGURE 3: Maximum speed-up vs. number of workers

As mentioned before, the worker returns the elapsed time of its execution, whereas the master executes all the events found up to the input event log files have been completely parsed and all dispatched tasks have been completed. We computed the observed speed-up for the test by dividing (1) the sum of all the elapsed times returned by each invocation of the worker into (2) the elapsed time the master run multiplied by a normalization factor to compensate the different speed between the machine running the master and the Planetlab nodes running the workers.
Therefore, the main experimental results from the parallel processing of log files are given in terms of how much close each set of workers is to achieve its theoretic maximum speed-up (see Figure 3) for different task size processed and, thus, providing the best processing time possible while parallelizing the data processing. To this end, Figure 4 shows the graphical representation of an extract of these results in relative terms for a sample of a specific 5-event size task.

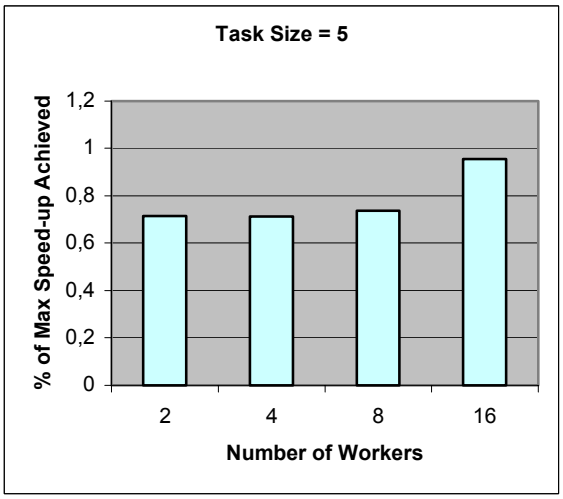

FIGURE 4: Relative speed-up vs. number of workers for a task size of 5 events

5.1. Analysis of the results

Analyzing the experimental results, we found that, on the one hand, from certain values of the task size, the speed-up observed was very close to the theoretic maximum achievable. This allows us to conclude that only for a very small value of the task size (i.e. small tasks involving less than 5 events to be processed) the impact on the speed-up can be great due to the cost of the transmission overhead. However, we also observed that the more workers we used in our tests the closer to the theoretic maximum was the speed-up achieved by the small tasks, and this increased quickly up to the point that, given a sufficient number of workers, even the smallest tasks (i.e. one-event task size) achieved considerable speed-up.

On the other hand, the homogenous behavior observed in Planetlab nodes justified our decision of testing with the same task size for all workers. However, in a real grid environment, task sizes should be adjusted per worker node case to fit the dynamically changing workloads the nodes may be experimenting and to account for different machine speeds.
We note, however, that though the results of this experiment are dependent on the form of the BSCW event log files, any other collaborative learning environment involving a great amount of online courses, groups, students and tutors along with the identification of the main parameters (i.e. the above-mentioned task performance, group functioning and scaffolding) is expected to generate a much larger amount and more complex events than BSCW. This scenario will take much more advantage of the benefits provided by a Grid environment, given that the BSCW system identifies only a very few parameters and as a result generates a very few different types of events.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have first argued how the provision of continuous awareness to on-line teams in CSCL environments can greatly improve the group activity in terms of decision-making, group organization, social engagement, support, monitoring and so on. To this end, large amounts of log information generated from the collaborative interaction need to be efficiently processed and analyzed. Moreover, in order to make the knowledge extracted from the analysis be useful for awareness purposes, users should be provided with both single information delivered fast in (almost) real-time and complex, exhaustive, yet structured deferred information, thus, stressing even more the processing requirements beyond those of a single computer.
Then, we proposed a Grid approach to overcome these demanding requirements by improving the processing time of a large amount of complex event information of group activity log files and we have shown how this approach could be useful even for the case of BSCW log files which are limited in terms of complexity of information.
According to the results obtained in our study the question whether the Grid is beneficial or not will heavily depend on the volume and structure of information being processed. Therefore, these results encourage us to keep up working on the development of a real working Grid implementation to address the problem of processing group activity event log files.
As ongoing work, we plan to improve the communication between master and workers in our Grid prototype by exploring the convenience of using other features provided by the GT3 such as the Reliable File Transfer service and OGSI notification service. We believe that all these enhancements can not but increase the performance of our grid approach and that our current prototype, thanks to its simplicity, succeeds at establishing a lower bound on the expectation on the performance gain that we expect to achieve by grid-enabling our *EventExtractor* application. Moreover, we also plan to address other necessary improvements on our Grid prototype in future iterations such as fault-tolerance and dynamic discovery of available workers.

Acknowledgements

This work has been partially supported by the Spanish MCYT project TIC2002-04258-C03-03, NSF VMT project (IERI grant #0325447) and Kaleidoscope NoE IST-507838 programme.

REFERENCES

[1] Gutwin, C., Stark, G. and Greenberg, S. (1995) Support for Workspace Awareness in Educational Groupware. *Proceedings of the ACM Conference on Computer Supported Collaborative Learning*, Bloomington, Indiana, USA October 17-20.
[2] Caballé S., Xhafa, F., Daradoumis, T. and Marquès, J.M. (2004) Towards a Generic Platform for Developing CSCL Applications Using Grid Infrastructure. *Proceedings of the CLAG/CCGRID'04*, Chicago, USA.
[3] 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.
[4] Dillenbourg, P. (ed.) (1999): *Collaborative Learning. Cognitive and Computational Approaches*. Elsevier Science Ltd. 1-19.
[5] Conklin, J. *Capturing Organization Memory*. Groupware 92. (1992) David D. Coleman (editor). San Mateo, CA: Morgan Kaufmann Publishers.
[6] Open University of Catalonia http://www.uoc.edu (web page as of February 2005).
[7] Xhafa, F., Caballé, S., Daradoumis, Th. and Zhou, N. (2004). A Grid-Based Approach for Processing Group Activity Log Files. *Proceedings of the On The Move Federated Conferences (OTM'04), First International Workshop on Grid*

Computing and its Application to Data Analysis (GADA'04), October 25 - 29, Larnaca, Cyprus. Lecture Notes in Computer Science, Heidelberg-Berlin: Springer-Verlag.
[8] Goux, J.P., Kulkarni, S., Linderoth, J. and Yoder, M. (2000): An enabling framework for master-worker applications on the computational Grid. *Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing (HPDC'00).* IEEE Computer Society.
[9] 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.
[10] Appelt, W. (2001): What Groupware Functionality do Users Really Use? *In Proceedings of the 9th Euromicro Workshop on PDP 2001*, Mantua, February 7-9. IEEE Computer Society, LosAlamitos.
[11] VMT: The Virtual Math Team Project, College of Information Science & Technology, Drexel University http://mathforum.org/wiki/VMT (web page as of February 2005).
[12] Stahl, G. (2002) Groupware Goes to School, *Proceedings of the 8th Int. Workshop on Groupware, CRIWG'02*, La Sirena Chile, LNCS, Vol. 2440, pp. 7-24. ISBN: 3-540-44112-3.
[13] Martínez, A., Dimitriadis, Y., Rubia, B., Gómez, E., Garrachón, I. and Marcos, J. A. (2003) *Combining qualitative evaluation and social network analysis for the study of classroom social interactions.* Computers & Education, Volume 41, Issue 4 , December 2003, Pages 353-368.
[14] Avouris, N., Komis, V., Fiotakis, F., Margaritis, M., Tselios, N. (2003) On tools for analysis of collaborative problem solving. *Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03)*. Athens, Greece, 9-11 July,. IEEE Computer Society 2003, ISBN 0-7695-1967-9.
[15] Hardings, J. (2003) An XML-based Query Mechanism to Augment Awareness in Computer-integrated classrooms. Proceedings *of the 11th International Conference on Artificial Intelligence in Education*, Australia.
[16] T. Daradoumis, A. Martinez and F. Xhafa (2004) An Integrated Approach for Analysing and Assessing the Performance of Virtual Learning Groups, *Proceedings of the 10th Int. Workshop on Groupware, CRIWG'04*, San Carlos, Costa Rica. Lecture Notes in Computer Science, Vol. 3198, pp. 289-304.
[17] Master-Worker: http://www.cs.wisc.edu/condor/mw/ (web page as of February 2005).
[18] Globus: http://www.globus.org (web page as of February 2005).
[19] Apache Axis: http://ws.apache.org/axis/ (web page as of February 2005).
[20] Planetlab: http://www.planet-lab.org (web page as of February 2005).