Efficient Interaction Analysis for an Effective Provision of Knowledge about the Discussion Process to CSCL Practices

Santi Caballé[1], Fatos Xhafa[2] , Thanasis Daradoumis[1], Ajith Abraham[3]

[1]Open University of Catalonia, Department of Computer Science, Multimedia, and Telecommunication
Barcelona, Spain
{scaballe, adaradoumis}@uoc.edu
[2]Polytechnic University of Catalonia, Department of Languages and Informatic Systems
Barcelona, Spain
fatos@lsi.upc.es
[3]Center of Excellence for Quantifiable Quality of Service, Norwegian University of Science and Technology
Trondheim, Norway
ajith.abraham@ieee.org

Abstract

The discussion process plays an important social task in Computer-Supported Collaborative Learning (CSCL) where participants can discuss about the activity being performed, collaborate with each other through the exchange of ideas that may arise, propose new resolution mechanisms, as well as justify and refine their own contributions and thus acquire new knowledge. Indeed, learning by discussion when applied to collaborative learning scenarios can provide significant benefits for students in collaborative learning, and in education in general. However, the discussion process in the context of distance education presents high dropout in comparison to traditional programs due chiefly to a sense of isolation of participants who do not have knowledge about others nor they can compare their own progress and performance to the group. To alleviate this problem, the provision of appropriate knowledge from the analysis of on-line interaction is rapidly gaining popularity due to its great impact on the discussion performance and outcomes. This implies a need to capture and structure all types of information generated by group activity and then to extract the relevant knowledge in order to provide participants with efficient awareness and feedback as regards group performance and collaboration. As a result, it is necessary to process and analyze complex event log files from group activity in a constant manner, and thus it may require computational capacity beyond that of a single computer. To this end, in this paper we show how a Grid approach can considerably increase the overall efficiency of processing group activity log files and thus allow discussion participants to receive effective knowledge even in real time. The context of this study is a real on-line discussion experience that took place at the Open University of Catalonia.

1. Introduction

When developing Computer-Supported Collaborative Learning (CSCL) [1] environments that support online collaborative learning, several issues must be taken into account in order to ensure full support to the online learning activity. One such key issue is interaction management and analysis to support awareness, coaching and evaluation, based on information captured from the actions performed by participants during the collaborative process [1]. The success of CSCL applications depends to a great extent on the capability of such applications to embed information and knowledge extracted from group activity interaction and use it to achieve a more effective group monitoring [1], [9].

The real context of this study is the virtual learning environment of the Open University of Catalonia (UOC) , which offers full distance education through the Internet. Part of UOC's courses' curricula includes the participation of students in on-line discussions with the aim of sharing and discussing their ideas. Indeed, the discussion process plays an important social task where participants can discuss about the activity being performed, collaborate with each other through the exchange of ideas that may arise, propose new resolution mechanisms, as well as justify and refine their own contributions and thus acquire new knowledge [2].

The provision of effective knowledge extracted from the information collected in CSCL environments is essential for any discussion process [2]. It allows implicit coordination of collaborative learning, opportunities for informal, spontaneous communication, and gives users awareness [3] and feedback [4] about what is happening during discussion. It is indeed crucial for group members to be aware of others' participation process as this may enhance the discussion process a great deal in terms of decision-making, group organization, social engagement, support, monitoring and so on [5], [9].

These ideas have been incorporated in the design of a collaborative tool called Communities of Learning Practice Environment (CoLPE), which was developed at the UOC to facilitate both the construction of knowledge among learners and the development of cognitive-acquisition skills, such as problem-solving abilities as well as the provision of an adequate multi-support framework so that tutors and peers can provide a suitable scaffolding when needed. CoLPE pursues these objectives by means of seeing discussion as a medium through which the building and distribution of cognition is effected.

CoLPE [5] is a web-based collaborative system designed to enable “democratic” collaborative learning that involves sets of on-line learners who share a learning activity to engage in collaborative production but who do not have a formal workflow for this collaboration. It also envisions enabling informal collaborative learning among non-technical learners or those who lack of the necessary resources to acquire such systems. To this end, CoLPE provides, among other features, hierarchical threaded discussion, support for a range of choices on discussion and voting methods and enables group coordinators and tutors without IT expertise to customize their discussion environments. Finally, this system implements the above-mentioned fundamental requirements to sustain collaborative learning applications by the representation and analysis of group activity interaction to facilitate coaching and evaluation as well as awareness and feedback about what is happening during the collaboration.

The first results of using CoLPE drawn from real collaborative learning show very promising benefits for students in a real context of learning and in education in general [5]. However, from the evaluation of CoLPE and its effects in the discussion process we came across important repercussions derived from certain non-functional requirements that by now are hard to be met, such as performance, scalability, fault-tolerance, and interoperability [6]. Concerning the first three issues, participants (i.e., students and tutors) reported many problems when trying to participate in the discussion by using CoLPE, which influenced the whole learning experience negatively.

Indeed, system's poor performance is one of the most frustrating aspects during the on-line collaborative learning experience as it makes participants’ requests be waiting for long periods of time to be served [6]. In order to keep on providing a high level of quality of service, a learning system should seamlessly scale to new resources of both hardware and software at the same pace as the workload increases. To this end, we show in this paper how a Grid approach can increase the overall efficiency of the system while processing a large amount of information from group activity log files [7].

Experimental results from previous research [7], [11] allow us to be confident with the gain provided by our Grid approach in terms of relative processing time and the benefits of using the inherent scalable nature of Grid while user concurrency is high and the input log files are growing up in both number and large size. The ultimate aim of this study is to show the feasibility of Grid technology to achieve an effective provision of the relevant knowledge to the discussion process.

This paper is organized as follows. Section 2 presents the experimental setting and data gathered using CoLPE to support a discussion process and above all its effects in the learning experience that motivated this study. Section 3 proposes a generic model and a Grid-based realization to efficiently manage group activity information. Section 4 summarizes the paper and points out future work.

2. Centralized approach to support a discussion process

An experience using CoLPE took place last term at the UOC involving 86 graduated students enrolled in the course Methodology and Management of Computer Science Projects. Students were equally distributed into two classrooms and participated in the experience at the same time. Students from one classroom were required to use the standard asynchronous threaded discussion forum offered by the virtual campus of the UOC while the other group of students used the new CoLPE outside the campus to support the same discussions with the same rules during the same time.

The whole experience consisted in carrying out a class assignment in the form an on-line discussion activity for 3 weeks. The students enrolled in the course were free to open discussion threads at convenience where they proposed strategies, ideas, etc., to appropriately deal with the topic of to the discussion (i.e, “Change management: necessity or virtue?”). Any student could contribute in a discussion thread as many times as needed so as to provide new argumentations with regards to the issue addressed. The only requirement was to submit at least one post to any thread.

The whole experience was supported by a Zope server [8] on the server side, which run on a single node (i.e., Linux SuSE 2.4.21-99 machine, Intel Pentium 4 CPU 2.00 GHz, 256MB RAM).

Table1. Main statistics results from the class assignment using both discussion tools.

Statistics	Standard tool	CoLPE
Number of students	43	43
Number of threads	29	17
Total of posts	174	93
Mean number (posts/thread)	M=6.0 SD=2,7	M=5,5 SD=4,5
Mean number (posts/student)	M=4,0 SD=1,6	M=2,2 SD=3,8

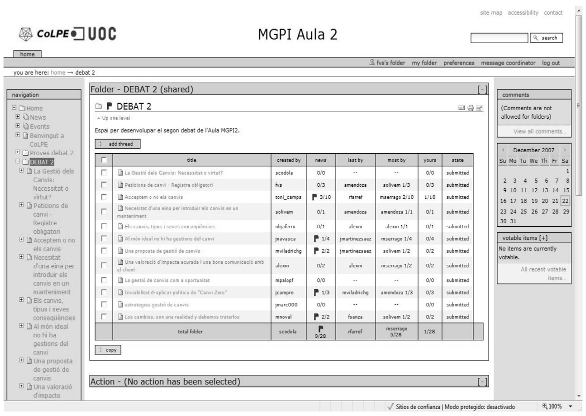
A statistical analysis of the results of the discussion comparing both the standard and CoLPE tools is shown in Table 1.

Table2. Excerpt of a questionnaire's results on CoLPE's evaluation to support the discussion process.

Selected questions	Average of structured responses (0 – 5)	Excerpt of students' comments
Asses CoLPE as a collaborative tool	1	"The system performed very slowly, I don't understand why the university is not able to provide us with a more powerful server!" "The standard tool is a chaos for large debates (…). Apart from many technical problems, CoLPE encouraged me to participate" "CoLPE is a powerful tool but most of times I couldn't even accede because of timeout problems"
Evaluate how the CoLPE fostered your active participation	1	
Did CoLPE help you acquire knowledge on the debate's issue?	2	
Compare CoLPE to the campus' standard discussion tool	2	

Despite successful previous experiences at the UOC [9] using similar *ad hoc* knowledge-based collaborative tools that impacted positively on the discussion process, the statistical results of this last experience showed that the discussion using CoLPE was poorly participative (see Table 1). Moreover, the results (see Table 2) of a structured and qualitative report conducted at the end of the discussion confirmed a negative effect of CoLPE in the learning experience.

In particular, the problems were originated as follows. First, Zope is a powerful server that demands a fairly amount of hardware resources to run. Second, the need to process and analyze the complex information collected from users' interaction and present the knowledge extracted (see Figure 1) in (almost) real time caused CoLPE to perform very poorly. Third, during the rush hours, the growing number of users who concurrently requested CoLPE's knowledge-related data-intensive functionalities generated noticeable performance repercussions on the underlying hardware supporting the system. Finally, the server was down once for a few hours during the rush time due to maintenance of the internal network.


site map
accessibility
contact
CoLPE UOC
MGPI Aula 2
search
home
fvs's folder
my folder
preferences
message coordinator
log out
you are here: home → debat 2
navigation
Home
News
Events
Benvingut a CoLPE
Proves debat 2
DEBAT 2
La Gestió dels Canvis: Necessitat o virtut?
Peticions de canvi - Registre obligatori
Acceptem o no els canvis
Necesitat d'una eina per introduir els canvis en un manteniment
Els canvis, tipus i seves conseqüències
Al món ideal no hi ha gestions del canvi
Una proposta de gestió de canvis
Una valoració d'impacte
Folder - DEBAT 2 (shared)
DEBAT 2
Up one level
add thread
copy
Action - (No action has been selected)
comments
(Comments are not allowed for folders)
View all comments...
December 2007
votable items [+]
No items are currently votable.
All recent votable items...
Sitios de confianza | Modo protegido: desactivado
100%

Espai per desenvolupar el segon debat de l'Aula MGPI2.

title	created by	news	last by	most by	yours	state
La Gestió dels Canvis: Necessitat o virtut?	scodola	0/0	--	--	0/0	submitted
Peticions de canvi - Registre obligatori	fvs	0/3	amendoza	solivem 1/3	0/3	submitted
Acceptem o no els canvis	toni_camps	3/10	rfarref	mserrago 2/10	1/10	submitted
Necesitat d'una eina per introduir els canvis en un manteniment	solivem	0/1	amendoza	amendoza 1/1	0/1	submitted
Els canvis, tipus i seves conseqüències	olgaferro	0/1	alexm	alexm 1/1	0/1	submitted
Al món ideal no hi ha gestions del canvi	jnavasca	1/4	jmartinezsaez	mserrago 1/4	0/4	submitted
Una proposta de gestió de canvis	mviladrichg	2/2	jmartinezsaez	solivem 1/2	0/2	submitted
Una valoració d'impacte acurada i una bona comunicació amb el client	alexm	0/2	alexm	mserrago 1/2	0/2	submitted
La gestió de canvis com a oportunitat	mpalopf	0/0	--	--	0/0	submitted
Inviabilitat d-aplicar politica de "Canvi Zero"	jcampre	1/3	mviladrichg	amendoza 1/3	0/3	submitted
estrategies gestió de canvis	jmarc000	0/0	--	--	0/0	submitted
Los cambios, son una realidad y debemos tratarlos	mnoval	2/2	fsanza	solivem 1/2	0/2	submitted
total folder	scodola	9/28	rfarref	mserrago 5/28	1/28	

Figure 1. Partial feedback presented to all participants.

As a consequence of this centralized approach, important non-functional requirements could not be completely satisfied in terms of fault-tolerance, scalability and performance. Despite the negative impact on the discussion process caused by the lack of fault-tolerance and user scalability, in this study we concentrate and focus on the performance repercussions caused by the large amount of complex information about group activity to be processed. Indeed, the information stored in very large log files and databases is often found with a certain degree of redundancy, tedious and ill-formatted as well as incomplete as at some cases certain user actions do not generate any log entry (e.g. user may leave CoLPE by either closing or readdressing the browser) and have to be inferred. As a consequence, treating this information is very costly in terms of time and space needing a great processing effort. This is certainly the first issue to be addressed so as to improve the overall system's performance. To this end a parallel approach is proposed next to process log files efficiently.

3. Efficient processing of group activity information

This section presents first a generic treatment model of how to structure log files for its later parallelization. Based on previous research [7], [11], a Grid approach is then incorporated following the Master-Worker paradigm so as to realize the approach. Finally, we give some guidelines of how to leverage real Grid infrastructure for the processing of log files.

3.1 A general model to structure log files

In a order to prepare the information collected from group interaction 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 occur-ring 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 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, for instance, 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.

3.2 A Grid-based processing of log files

Over the last years, Grid computing has become a real alternative for developing parallel applications that employ its great computational power [10]. 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) [11].

The Master-Worker (MW) [11] model (also known as Master-Slave or Task Farming model) has been widely used for developing parallel applications in traditional supercomputing environments such as parallel machines and clusters of machines. 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.

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.

The architecture of the application (see Figure 2 and [11]) 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.

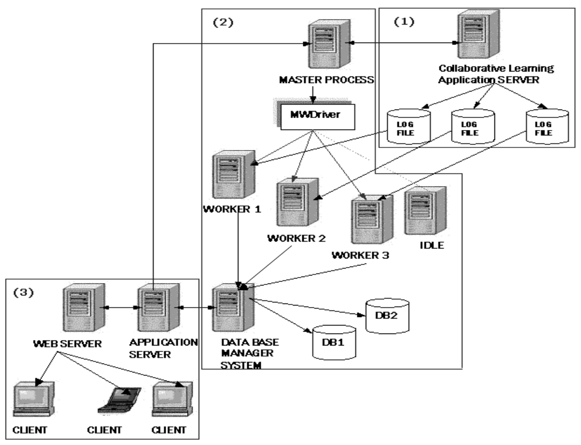


Figure 2. Generic architecture of the application for processing log files

Next subsection introduces a realization of this general approach based on the architecture showed in Figure 2 in the form of Grid middleware to efficiently parallelize the processing of logs files.

3.2.1 The Master-Worker application

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 complete description of the algorithms for the task, and master and worker processors can be found at [11].

3.2.2 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.

3.3 Adding Grid infrastructure

We show here 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, moreover, there is no need for synchronization between the worker processors as tasks are completely independent from one another. To this end, we provide the guidelines for a minimal Grid implementation prototype using the standard Globus Toolkit [12] middleware as well as point out how to deploy it on the "real" grid context of the Planetlab [13] platform.

3.3.1 Using Globus Toolkit

The Globus Toolkit (GT) [13] is the actual de facto Grid middleware standard. The core of the GT is a Grid service container implemented in Java that leverages and extends the Apache's AXIS [14] web services container.

Planetlab is turned into a Grid fabric by installing the GT Grid service container. The worker is then implemented as a simple Grid service and deployed on the GT3 container. Finally, a master is in charge of 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 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 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.

Please note this is not a real GT Grid implementation of the MW paradigm but a proof-of-concept, thus important features in a real environment such as fault-tolerance and dynamic discovery of available workers, are still to be considered.

4. Conclusions and future work

In this paper, we have first argued how the provision of continuous information about the discussion process in on-line CSCL practices can greatly improve the group activity in terms of decision-making, group organization, social engagement, support, monitoring and so on. However, from our experience at the UOC certain requirements are especially frustrating when they are not fulfilled appropriately during the discussion process, such as fault-tolerance, scalability, and performance. As a solution to alleviate this problem we have presented a general 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

We plan to implement this general approach by developing both an *ad hoc* processor for the CoLPE's log files and a Java-based MW application, which will be deployed on the PlanetLab's nodes and turned into Grid by using GT middleware. In addition, we plan to make an in-depth analysis through data mining techniques to provide tutors with ongoing progress of students learning during the discussion activity.

5. REFERENCES

1. Dillenbourg, P. (ed.) (1999): Collaborative Learning. Cognitive and Computational Approaches. Elsevier Science Ltd. 1-19.
2. Schellens, T. & Valcke, M. (2006). Fostering knowledge construction in university students through asynchronous discussion groups. Computers & Education. 46(4):349-370, Academic Press: Elsevier Ltd.
3. Gutwin, C., Stark, G. and Greenberg, S. Support for Workspace Awareness in Educational Groupware. in Proceedings of the ACM Conference on Computer Supported Collaborative Learning, Bloomington, Indiana, USA October 17-20, 1995.
4. 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.
5. Caballé, S., Feldman, J. (2008). CoLPE: Communities of Learning Practice Environment. In proceedings of DIAC-2008/OD2008. Berkeley, CA, USA. To appear.
6. Caballé, S., Xhafa, F., Daradoumis, Th. (2007). A Service-oriented Platform for the Enhancement and Effectiveness of the Collaborative Learning Process in Distributed Environments. In Proc. of GADA 2007. Vilamoura, Algarve, Portugal. LNCS, vol. 4804. ISBN: 978-3-540-76835-7.
7. Caballé, S., Xhafa, F., Daradoumis, Th. (2008). A Grid Approach to Efficiently Embed Information and Knowledge about Group Activity into Collaborative Learning Applications. Book: The Learning Grid Handbook. pp. 173-197. Amsterdam, The Netherlands: IOS Press. ISBN: 978-1-58603-829-8.
8. Zope: http://www.zope.org (web page as of April 2008).
9. Caballé, S., Daradoumis, Th., Xhafa, F. (2008). Providing an Effective Structured and Context-aware Asynchronous Discussion Forum for Collaborative Knowledge Building. In Proceed. of ED-MEDIA 2008. Vienna, Austria. AACE Press. To appear.
10. Foster, I. and Kesselman, C. The Grid: Blueprint for a Future Computing Infrastructure. pp. 15-52. Morgan Kaufmann, San Francisco, CA, 1998.
11. Xhafa, F., Caballé, S., Daradoumis, Th., & Zhou, N. (2004). A Grid-Based Approach for Processing Group Activity Log Files. In Proceed. of GADA'04, Agia Napa, Cyprus. LNCS, vol. 3292, Springer 2004, ISBN 3-540-23664-3.
12. Globus: http://www.globus.org (web page as of April 2008).
13. Planetlab: http://www.planet-lab.org (web page as of April 2008).
14. Apache Axis: http://ws.apache.org/axis/ (web page as of April 2008).