Social Networking Technologies for Free-Open Source E-Learning Systems

Francesco Di Cerbo, Gabriella Dodero, and Giancarlo Succi
CASE – Center for Applied Software Engineering
Free University of Bolzano/Bozen
Piazza Domenicani, 3
I-39100 Bolzano-Bozen,
{name.surname}@unibz.it
WWW home page: http://www.unibz.it

Abstract. In this paper, we illustrate our methodology for academic teaching, based on cooperative learning paradigm, which also relies on cutting edge e-learning techniques. We use Web 2.0 resources to fulfill requirements for an interactive-constructivistic "learning space", where blended-teaching paradigm may be proficiently applied. We extend an existing Free/Open Source Software Learning Management System, to create a cooperative and community-based learning space adherent to our proposal.

1 Introduction

Much emphasis has been put in the importance of a constructivistic approach to achieve effective learning at all levels, including the academic one. The so-called "Dublin descriptors" adopted in many European universities as common educational indicators stress the importance of coupling the "knowledge" with the "applied knowledge" within reference environments, and across Europe, academic teaching methodologies are now promoting deep restructuring of curricula. In facts, the development of the so-called "soft-skills" has become an indispensable complement to any discipline, be it technology oriented or in the humanities. Pedagogues have already identified methodologies which may be applied in order to achieve such results; they are collectively called "active learning". Examples are "problem-based learning" ([1]), "troubleshooting" ([2]) or "case-based teaching" ([3]).

Several pilot experiences already support this claim (see [4] and [5]), even if an exhaustive experimentation involving e-learning specialists and students at various levels and disciplines is not yet completed.

The development of a suitable technological infrastructure to support such activities has been mostly concentrated on the possibility to pass from the learning environment concept (the status-quo of the market) to that of "dynamical learning space" (DLS). In a DLS, people involved in learning are identified by their specificities, and meet each other to collaborate to the negotiation and construction of knowledge. The implementation of a DLS, giving importance and attention to the individuals inside a distributed process of learning, is our main goal.

Please use the following format when citing this chapter:

Di Cerbo, F., Dodero, G. and Succi, G., 2008, in IFIP International Federation for Information Processing, Volume 275; *Open Source Development, Communities and Quality*; Barbara Russo, Ernesto Damiani, Scott Hissam, Björn Lundell, Giancarlo Succi; (Boston: Springer), pp. 289–297.

The concept of “dynamical learning space” is not originating in the virtual environment by itself. Indeed, in a real (not virtual) learning space, both teacher and his/her class participate to the creation of a shared knowledge; they build up meanings and concepts where every individual has own importance inside the process. Teachers have the lead of the community, steering global effort towards learning targets, and every student may/must contribute.

Such a process requires very skillful teachers, able to involve learners’ community, letting everyone get his/her own role inside the teamwork. In case of a virtual teaching, such as in e-learning scenarios, a teacher should rely on a set of software tools, letting him involve his/her class, like during lessons in presence. Our software helps the teachers to coordinate such virtual communities, specifically at interaction level, in order to form a social network.

We developed our software in order to make possible this co-construction of learning materials and concepts by means of virtual experiences. We find that such approach can be useful not only in virtual (e-learning) classes, but when lessons are given both in presence and in virtual systems (blended learning). Such software also gives a precise and “automatic” track of activities conducted, without the distractive effect caused by recording them by hand. In this way, students and teachers may concentrate on learning process, without caring too much about activities tracking. A more detailed description of the pedagogical design is available in [6].

In Section **Error! Reference source not found.** we describe the choice of tools, and interfacing functionalities to augment them according to the “DLS” paradigm. In Section **Error! Reference source not found.** the underlying technologies are briefly described. Conclusive remarks are presented in Section **Error! Reference source not found.**.

2 Tools

Inside an e-learning scenario, the concept of DLS naturally translates into that of a virtual environment, where interactions are welcomed and eased, and where every community service, like wikis and forums, contributes to the creation of a common knowledge as part of a structured learning process. Here, students and teachers are free to operate and move in a sort of virtual reality: a place, coherently with new Web 2.0[7] issues, where to put opinions or contents, to meet the class, even to find amusement, without a fixed interaction stereotype.

To this aim, previous research in Human-Computer Interaction (for instance, the work of Gräther and Prinz [8]) on community and presence awareness, and especially the concept of *social translucence* ([9]), were exploited. Functional interaction metaphors and schemes of services oriented to social networks were analyzed, where theories of cooperative learning and social constructivism can be put into practice.

But should such a new concept like the DLS be implemented from scratch, or can it be superimposed on top of some existing LMS? Existing infrastructures and services were evaluated with respect to suitability to social constructivism. Free/Open Source Software solutions were especially considered because of the availability of particular license agreements, which allow users to legally and easily extend software, under certain conditions (see [10],[11]).

What is important to remark, here, is the freedom of use that such software license gives to licensee; a user is entitled to use, to study, to modify, and to redistribute a derivative work made upon original one. Our analysis included both a technical evaluation and a qualitative-strategic one, with the aim to understand the extensibility, the quality of the implementation, size, and support provided by the international community. As a result, we chose the Learning Management System Moodle [12] as our privileged software environment.

For Moodle, as well as for any other LMS, the introduction of a DLS implements a new social networking model. since every user is associated to an avatar, free to move in a web page, interacting with other users and with resources (for instance, opening course material or a real time chat). Logical proximity of activities is naturally mapped into physical proximity of the avatars in the virtual space. The possibility of using a spatial metaphor has proven succesful in the past (e.g. [13]).

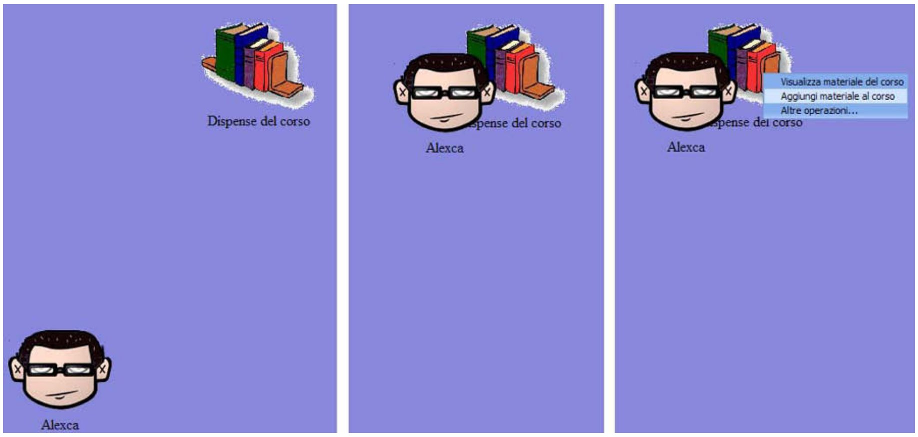

Fig. 1 Example of an interaction between an avatar and course material

A number of services of interest, like forums and wikis, are already found in Moodle. We added a videoconferencing system based only on Java (applets and services) in a client/server architecture, and an AJAX ([14]) whiteboard, shared among the community. Both are browser independent and do not require additional software installations at client side.

Figure 1 shows an example of an interaction between an avatar and a course material, for instance a book available to students. The avatar moves close to the icon representing materials, and a pop-up menu appears, showing next possible actions. This happens also in case of interactions between avatars, in order to open a private chat session. The software development process we followed includes strong involvement of pedagogues in the choices taken by developers. Early feedbacks from them guarantees the highest quality and usability for the customers of the final product.

3 Implementation

Our software consists on two main subsystems: one implements all the operations at client side, while the other is responsible for managing all the interactions between all the users and the system. We defined a set of system behaviors, called *actions*, regarding possible interactions between clients and server. Then, we also defined a set of widgets and utility objects to be used at client-side. Each object is associated to a number of actions that are needed to perform its own tasks. These definitions have been formalized in XML documents, in order to be independent from any programming language (neither client- nor server-side). We refer to these elements as “XML Interfaces”. A representation of the two subsystems as well as XML interfaces is available in Figure 2.

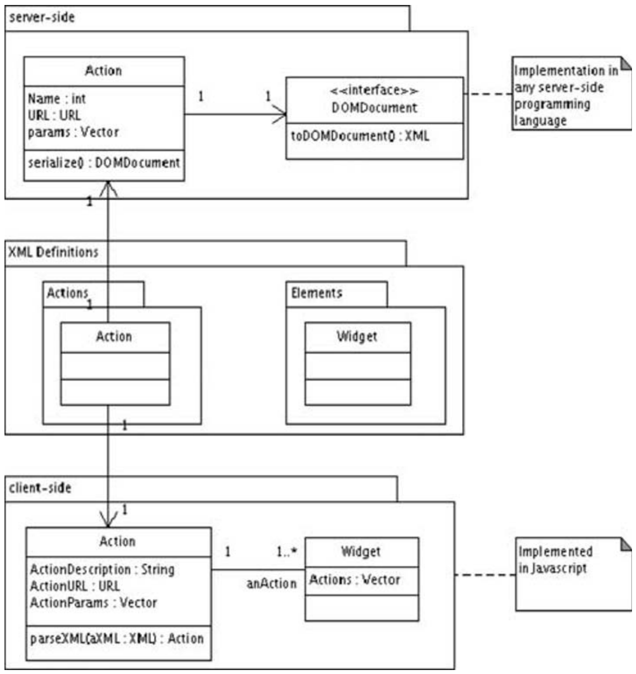

Fig. 2 Subsystems design

The development of objects for both sides was based on such XML interfaces; in fact, we designed and developed class definitions that are able to cope with specified exchange messages in XML format, if they can parse XML interfaces to produce accordingly XML requests. In this way, we are not bound to any programming language or software package.

Two special entities are responsible for communications between client and server: Status Manager and Status Monitor. Both of them embed the concept of “status”; as we aim at realizing a tool focused on live social interactions, we require that all the users share the same experience. Every user is cooperating with others, for instance by moving an avatar in response to another movement. As a consequence all users also have to share the very same status; we implemented this requirement keeping the state of the whole application server-side, and exchanging periodic updates with clients. Once an action has been performed by one of the clients, all the others are being notified within 1-2 seconds. This mechanism is based on the mentioned Status Monitor and Status Manager. Status Monitor is an object managing all the client side tasks and operations. In fact, it is responsible for creating the initial environment (avatars, rooms, widgets and so on), and to track any modification to its internal status (i.e., movement of the user's avatar, insertion of a new message in the real-time chat and so on). Once a modification happens, Status Monitor sends an update message to its server-side counterpart, Status Manager.

There are as many instances of Status Monitor as there are clients, while the system has one and only Status Manager, as we need a central coordination point for the whole system. Then, Status Manager sends an update message at predefined intervals; this message is created considering the past history of all the Status Monitors since the last update message.

Figure 3 provides a sequence diagram showing interactions among subsystems.

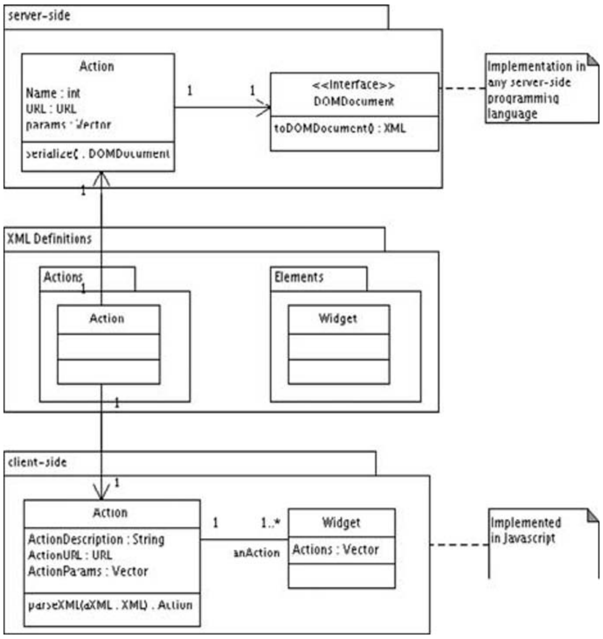

Fig. 3 Sequence Diagram for main objects: Status_Monitor and Status_Manager

XML Interfaces

In our system, interfaces are defined as XML Schema documents; they can be used to verify if a document is well-formed. We adopted XML Schema in order to enable verification of messages in our system, and to keep interfaces independent from actual language used in system implementation. Table 1 shows a sample action request.

Table 1: An example of XML Action message

```
<?xml ... ?>
<action name="aName">
 <param name="1"/ value=2>
 <param name="2"/ value=2>
  <url>

 </url>
</action>
```

Client-side implementation

Required objects and graphical widgets have been implemented in Javascript, the programming language we chose at client-side. Our main issue was delivering browser-independent code. In fact, different browsers often yield different behaviors for Javascript code, both in the graphical part, and in the invocations to asynchronous methods, that are the base of AJAX paradigm. For these reasons, we adopted a Javascript framework to encapsulate and provide us portable solutions for all the issues mentioned: the Dojo toolkit

Dojo (available from www.dojotoolkit.net) is an Open Source DHTML toolkit written in JavaScript. It supports asynchronous calls (i.e., all AJAX interactions), as well as unit testing and layering, with a very small core which provides the package system. The Dojo toolkit enables our software to be browser-independent, as it detects the type of browser where it is executed, and automatically adapts the actions it is required to perform to the actual environment.

Server-side implementation

We implemented our server-side subsystem using PHP, due to the choice of Moodle as our base LMS. PHP version 5 allowed us to exploit a number of services, including reflection and full support for Object-Oriented programming, that we needed. Moreover, as Moodle uses PHP, this simplifies the deployment process, without requiring to set up an heterogeneous system to use our software.

4 Conclusions

We have described the pedagogical approach, tools chosen, and implementation technologies that we have been employing in order to support social networking inside a new generation LMS. The system extends the well-known Moodle Free/Open Source software with a “dynamical learning space” concept, providing a novel interaction model, especially suited to support blended learning. At present, the system has been deployed on Moodle 1.8.3, and we are starting to extensively experience it with users (university teachers and students).

To evaluate our solution, we have started to use automatic tools for collecting process metrics ([15]) already available and developed at the Free University of Bolzano Bozen ([16],[17]). This will allow us to discover usability patterns inside users' experience in an objective way. It is possible, for instance, to understand what services are the most used, for how much time, and the “productivity”, in terms of outcomes deriving from user's experience with that specific tool. This would provide a base of evidence useful on two different scenarios: a technical one, to analyze introduced improvements and eventually refine them, and the pedagogical one, to bring quantitative confirmations for qualitative considerations.

Using both feedbacks coming from educational specialists and quantitative measures, it shall be possible to formulate a complete report of our experience, that in conjunction with a new Learning Management System platform would result in a complete educational proposal, a framework able to coordinate and to combine both in-presence and virtual lessons.

Bibliography

[1] Ward, J.D. and Lee, C.L., A Review of Problem-based Learning, Journal of Family and Consumer Sciences Education, Vol. 20 , n. 1, 2002,
[2] Jonassen, D.H., Hung, W., Learning to Troubleshoot: A New Theory-Based Design Architecture, Educational Psychology Review, Vol. 18, n. 1 , 2006, DOI: 10.1007/s10648-006-9001-8
[3] Jonassen, D.H., Hernandez-Serrano, J., Case-based reasoning and instructional design: Using stories to support problem solving, Educational Technology Research and Development, Vol. 50 , n. 2 , 2002, DOI: 10.1007/BF02504994
[4] Stefanova E., Sendova E., Nikolova I., Nikolova N., When I*Teach means I*Learn: developing and implementing an innovative methodology for building ICT-enhanced skills., 2007, Proceedings of the IFIP Conference Informatics, Mathematics, and ICT: a ‘golden triangle’ IMICT 2007, pg., Northeastern University, Boston, MA,
[5] Dodero G., Ratcheva D., Stefanova E., Miranowicz M., Vertan C., Musankoviene V., The virtual training center: a support tool for teachers community, 2007, Proceedings of Balkan Conference in Informatics (BCI 2007), pg. , Demetra Ltd,

[6] Di Cerbo, F., Succi, G., A proposal for interactive-constructivistic teaching methods supported by Web 2.0 technologies and environments, 2007, Proc. of 18th International Conference on Database and Expert Systems Applications, 2007 (DEXA '07), pg. 648-652, ,
[7]O'Reilly, T., What Is Web 2.0, 2005, http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
[8] Gräther, W. and Prinz, W., Supporting the Social Side of Large Scale Software Development, 2006, Supporting the Social Side of Large ScaleSoftware Development, pg. , Microsoft Research,
[9] Erickson, T. and Kellogg, W.A., Social translucence: An approach to designing systemsthat mesh with social processes., Trans. Computer-Human Interaction, Vol., n. 7, 1 , 2002,
[10]: Stallman, R.M., Free Software, Free Society: Selected Essays of Richard M. Stallman, 2002
[11] Raymond, E., Cathedral and Bazaar, 1999
[12] Dougiamas, M., Taylor, P.C., Moodle: Using Learning Communities to Create an Open Source Course Management System, 2003, roceedings of the EDMEDIA 2003 Conference.
[13] Pfister, H.-R., Wessner, M., Beck-Wilson, J., Miao, Y., & Steinmetz, R. (1998). Rooms, protocols, and nets: Metaphors for computer supported cooperative learning of distributed groups. In: Proceedings of the ICLS'98 - International Conference of the Learning Sciences (pp. 242-248). Charlottesville, VA: AACE Association for the Advancement of Computing in Education.
[14] Garrett, J., Ajax: A New Approach to Web Applications, , www.adaptivepath.com/publications/essays
[15] Humprey, W., Introduction to the Personal Software Process, 1997
[16] Sillitti A., Janes A., Succi G., Vernazza T., Collecting, Integrating and Analyzing Software Metrics and Personal Software Process Data, 2003, Proceedings of EUROMICRO 2003, pg. 336-342,
[17] Scotto M.,Vernazza T.,Sillitti A., Succi G., Managing Web-Based Information, 2004, Proceedings of ICEIS Conference, pg. 575-578.