An Application Framework for Collaborative Learning

Aiman Turani[1], Rafael A. Calvo[1], and Peter Goodyear[2]

[1] Web Engineering Group, School of Electrical and Information Engineering,
The University of Sydney, Australia
{aimant,rafa}@ee.usyd.edu.au
http://www.weg.ee.usyd.edu.au
[2] CoCo, Faculty of Education, The University of Sydney, Australia
p.goodyear@edfac.usyd.edu.au

Abstract. We present the design of a new web application framework for collaborative learning. The framework guides users (i.e. teachers) in implementing online activities based on well-known pedagogical techniques, and simplifies the development of collaboration tools needed to carry out those techniques. There are common tasks across various techniques and our framework organizes them in a layer of abstraction. The framework model has four abstraction layers: Pedagogical Models, Pedagogical Techniques, Collaboration Tasks Patterns, and CSCL Tools. By using this framework, developers will place the control of designing and implementing new functionalities in the teacher's hand rather than in the software designer's.

1 Introduction

Learning usually happens when students are active and collaborate in solving a problem in a social environment [1]. In fact, recent pedagogical research shows that learning [2] is not simply knowledge assimilated with the help of a more knowledgeable person or mediated by a computer system, but also jointly constructed through solving problems with peers by a process of building shared understanding [3]. Collaboration software can be used to support this process, but generic collaboration software is not always appropriate or sufficient to build a meaningful learning experience. Collaborative learning software, as the one described here, brings into the software design the good practices of the established educational design methodology.

Figure 1 shows a model for a collaborative learning process. This process is described as an interactive flow. First the process starts with course objectives specified by the instructor or department. Then the educational context [4] is used to help teachers select a pedagogical technique. Boyley, defines a pedagogical technique as a manner of accomplishing teaching objectives according to how the technique prescribes student interaction with other students and resources [5].

In order to improve the quality and reduce the cost of online teaching, researchers are studying how to increase the reuse of learning content and instructional design. Today, a lot of the research by learning technologists emphasizes 'learning objects' (a way of packaging content modules) reuse. Regrettably, this work might reinforce the idea of learning as information 'transfer' (from the teacher or content to the student) which is a cause of low learner motivation, low engagement, and isolation [6]. Meanwhile, there has been a recent paradigm shift among university teachers, in which activities and collaboration take priority over content delivery. This shift has

D. Lowe and M. Gaedke (Eds.): ICWE 2005, LNCS 3579, pp. 243–251, 2005.

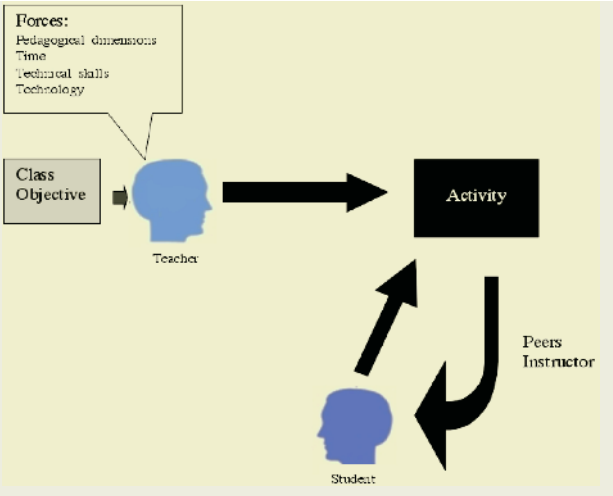

Fig. 1. Collaboration learning process

significant implications for online teaching, where software systems must be built to support these new pedagogical designs. Companies and universities have addressed this new paradigm with a variety of approaches, but in most cases these institutions lack software systems that embody these teaching strategies and support teachers in the design of courses with a strong collaborative component. Where these systems exist, they tend to focus on managing content rather than collaboration.

Engineers design software with abstractions and patterns as part of their standard software engineering practice. The essence of this practice is constructing representations that can communicate the commonalities in a number of problem/solution scenarios. Meanwhile, educational researchers have found that there are commonalities between many of the learning tasks and have defined and designed several 'Pedagogical Techniques' currently used by many teachers [7, 8]. We use these in the process of producing the abstractions we describe later.

Currently, there are several approaches for building software that supports those pedagogical techniques in an online environment. The first, a bottom-up approach, is the most commonly used and involves providing generic collaboration tools such as e-mail, bulletin boards, text chat, or computer conferencing [8] that the teacher then 'bundles' and sequences to create a pedagogical technique. For most pedagogical techniques, those individual tools are not enough [9]. For example, a teacher, who wants his students to participate in a brainstorming technique, might restrict his selection to a bulletin board and a text chat but he will probably need more than that. He might, for example, need an idea chart to hold the posted ideas, a timing tool to keep up the time, a voting tool to select the best idea, etc. Therefore, more than one tool might be required to carry out the brainstorming session. The *technical* solution is to provide more tools, other than those four, for teachers to select from, and enable them to sequence those tools according to the technique structure. If the tools were available, we could build new techniques, but would also put more pressure on the teacher to learn what are the best tools and how to configure them to carry out the activities. Finally this approach would also introduce more difficulties to students.

A second approach would be to provide teachers with specific bundles of tools for each pedagogical technique. This would be hard to implement due to three main reasons: first, the large number of possible pedagogical techniques [9], second, the same pedagogical technique might be performed with different scenarios since no single pedagogical technique structure is agreed upon among all teachers, third, teachers should be able to design new pedagogical techniques.

Our web engineering approach uses the domain specific knowledge from educational designers who have noticed that there are common components embedded in most pedagogical techniques, such as forming groups, provisioning topic information, monitoring, text-based discussion, etc. These components are at a different level from the tools used in the first approach. In this approach we also use the concept of application frameworks [10] that benefits the development of reusable, flexible, and customizable components in designing CSCL applications [11]. However, application frameworks consist not only of software components but also of design patterns. The component represents code reuse while a pattern represents design reuse [12]. Pedagogical techniques can provide the design reuse. In other words, a structure of a certain pedagogical technique, for example a debate in a political science class, could be reused in other learning domains. There is a big potential benefit from applying this new paradigm but the problem lies in the identification and dimensioning of components due to the large variety of abstraction methods used by pedagogical researchers and software engineers. Our approach to this problem is the four layered framework model introduced in this paper. This framework design will provide the basis of an implementation that helps teachers choose a pedagogical technique, and if necessary design the most appropriate tool for that technique. Section 2 describes each of the abstraction levels and Section 3 concludes the discussion.

2 Four Layers of Abstraction

A key design requirement in our application framework is that the teacher should be able to manage, customize and reuse ideas in the whole instructional design process. First, teachers would select which pedagogical model to use according to certain environmental forces. Second, a list of suggested pedagogical techniques associated with the selected model would be presented. Teachers would select the pedagogical technique according to what kind of problem students were asked to solve (learning objectives) and its context. Third, based on design reuse, the system would automatically present a suggested list of tasks to be preformed by the teacher and learners. At this point, a teachers' participation in the design process would terminate. Fourth, the system would map each task to a certain collaboration tool and then assemble all mapped tools to a single tool. Figure 2 shows how the four functional layers are related. This is followed by a detailed description of each layer.

2.1 The First Layer (Collaborative Pedagogical Models)

This is the most general layer as it describes the different collaboration pedagogical models. Morten [8] pioneered work in defining a framework for pedagogical CMC models. His contribution was to divide the existing models into four groups according to four communication paradigms used in computer-mediated communication. The

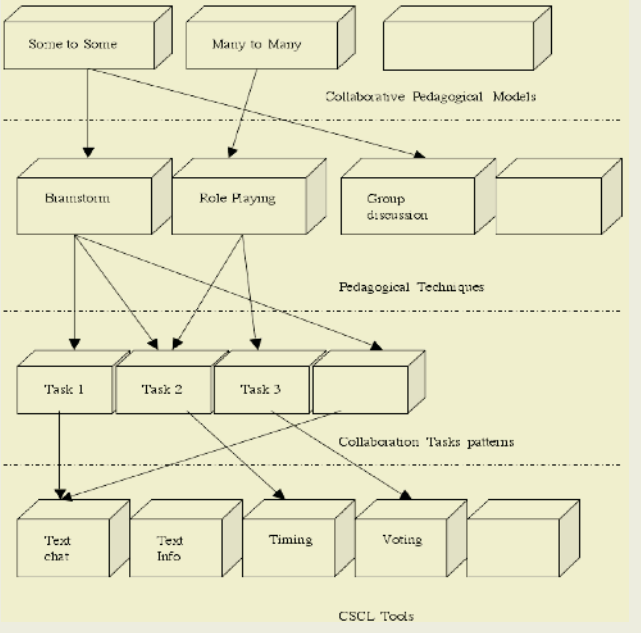

Fig. 2. The four layers of abstraction

first model is classified as *one-alone*, which can be preformed by retrieving information from online resources without communication with the teacher or other students. The second model is classified as *one-to-one*, which can be conducted via e-mail applications. The third model is classified as *one-to-many*, which typically is conducted via bulletin boards. Finally, *many-to-many* techniques, which can be organized within computer conferencing systems or bulletin board systems. As discussed before, defining the models around fixed communication tools makes them hard to adapt or used in other contexts. Pedagogical models should be divided according to their learning objectives, context and forces, not upon tools. Forces could be divided into three types: class size, time and technology. The class size is an important factor in choosing which model to follow. For example, a *one-to-one* model is difficult to implement in a class with 100 students. The second type of force is time. Time plays a major role in design considerations, and often it is the most important factor for a teacher designing a learning activity. A teacher needs to know how much time might be required in designing and facilitating the activity, and how much time students would be required to spend on it. The third type of force is technological which has two aspects, 'tools' particularly which CSCL tools are available and 'technical difficulty' that indicates how much time the teacher would need to learn the tool.

2.2 The Second Layer (Pedagogical Techniques)

Instructional design researchers have documented many different pedagogical techniques. We will build on work by Morten [8] who listed some of the pedagogical techniques used in adult education. Some differences are worth noting: his *one-alone*

model was not used because our focus is on collaboration. Second, we have subdivided the *many-to-many* model into: *some-to-many* and *some-to-some* model. The *some-to-many* model is used to categorize some techniques that imply a small group acting in front of a larger group. The *some-to-some* model is used to categorize small groups interacts within. Figure 3 shows how pedagogical techniques are categorized.

In order to help teachers select the appropriate pedagogical technique, each technique is represented by a pattern. A pattern describes a problem repeatedly in the environment, and then describes the solution [13]. The solution is an essential part of a pattern since it provides the basic knowledge to identify and form the pattern's list of tasks. Table 1 shows a Group Discussion technique pattern.

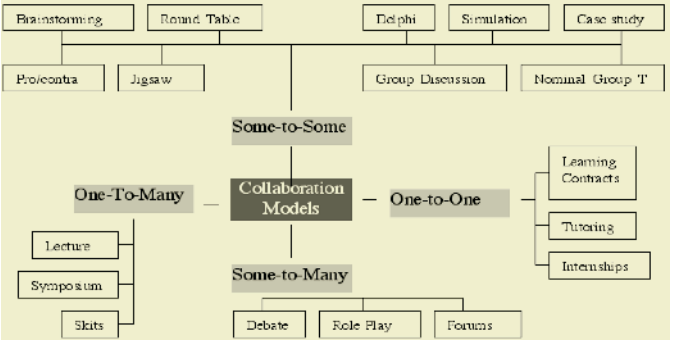

Fig. 3. Pedagogical techniques categorization

Table 1. Group discussion technique pattern

Pedagogical Technique	Group Discussion
Problem	How to establish and encourage group knowledge sharing among students
Example	Discussing how to improve writing and speaking skills for a foreign student
Context	Small groups with different skills and backgrounds interact to develop more knowledge among them
Solution	▪ Teacher specifies the discussion topic, related material and the time ▪ Student will start a Free discussion according to their experience and may build upon others' knowledge ▪ Teacher will guide the discussion when needed
Actors	▪ Teacher ▪ Learner ▪ System

2.3 The Third Layer (Collaboration Tasks Patterns CTP)

Pedagogical techniques are composed of a set of tasks that need to be performed by teachers and students. Therefore pedagogical techniques are considered to be task-oriented. Most of the pedagogical techniques (brainstorming, debate, group discussion, role-play, etc) have many tasks commonalities between them, such as a session creation task, a group forming task, a guiding task, a text interaction task, etc. Tasks

are mainly subdivided between three roles: Teacher, Learner, and System. Some of these tasks are mandatory and some are optional. Teacher tasks can be subdivided into four main components: management tasks, information provision tasks, guiding tasks and assessments tasks. Learner tasks can also be subdivided into four components: group level tasks, individual level tasks, management tasks (for some roles) and additional tasks.

A primary list of tasks is identified after careful analysis of 10 well known pedagogical techniques (Brainstorming, Group Nomination, Group discussion, Round Table discussion, Debate, Role playing, Jigsaw, Pro/Contra, Lecture, One to One Tutoring), which cover all activity models (one-to-one, one-to-many, some-to-some, many-to many) [8]. There are currently 39 common tasks that could form any of those pedagogical techniques. The list of common tasks for the teacher is shown in Table 2 followed by the list of common tasks for students in Table 3.

Table 2. List of common tasks for teacher

Managements Tasks	1	Creating a collaboration session based on pedagogical technique
	2	Group formation
	3	Controlling the session (Activating, Terminating)
	4	Controlling the floor during session process
	5	Setting and controlling the timing
Guiding Tasks	6	Monitoring groups through the process
	7	Tracking participants interactions levels
	8	System Automatic Tracking and Supporting
	9	Providing Guidance
	10	Asking and Answering questions
Provision Tasks	11	Defining the objective of the session (title)
	12	Providing subject related info
	13	Providing sub-subject related info to a specific Role/Group
	14	Providing the session rules and instructions
	15	Preparing presentation text slides
	16	Preparing presentation Flash animations slides
	17	Video presentation
	18	White board drawing
Assessments	19	Results Evaluation
	20	Viewing session details

These tasks will enable the teacher to play a major role in the development of the collaboration tools in three ways: First, the teacher can select one of the supported pedagogical techniques to form a list. Each pedagogical technique will have it's own default tasks list based on a detailed study on that technique. Second, for flexibility, the teacher can: agree with the defaults tasks, delete some of those tasks and/or add new tasks. This will help the teacher to reshape the technique without losing the original pedagogical structure embedded inside. Third, a tasks list can provide a suitable mean for teacher and developer to generate a new pedagogical technique by simply dragging tasks from the list. This will make the designing of a new pedagogical technique's tool simple enough to place the control of the design in the teacher's hand rather than the software designer's hand.

Table 3. List of common tasks for learner

Group level Tasks	21	Group Joining
	22	Small group discussion (Free/Round)
	23	Large group discussion (Controlled by Facilitator/System)
	24	Debating (Controlled by Facilitator/Script)
	25	Role Playing (Controlled by Facilitator/Script)
	26	Ideas posting
	27	Application sharing
Individual Tasks	28	Private writing
	29	Asking/answering/needing teacher's help
	30	Summarizing the result (resolution/conclusion)
	31	*one-to-one* interaction
	32	Individual assigned reading
Management Tasks	33	Controlling the floor
	34	Controlling the timing
Additional Tasks	35	Participating in ideas voting to select the best idea
	36	Participating in yes/no questionnaires
	37	Participating in multiple choice questionnaires
	38	Participating with emoticons
	39	Joining multiple groups

Table 4 shows a default list of tasks needed to conduct Group Discussion technique. There are some tasks mandatory to all techniques such as 1,2,10 while the rest of the 13 tasks are driven from the solution part in Table 2

Table 4. Group discussion technique list of tasks

Teacher tasks:	Task No.
✓ Creating a Group Discussion session	1
✓ Group formation	2
✓ Defining the Group Discussion session title	10
✓ Providing subject related information	11
✓ Controlling the session (Activating, Terminating)	3
✓ Controlling the floor	4
✓ Setting and controlling timing	5
✓ Monitoring groups through the process	6
✓ Providing Guidance	8
✓ Results evaluation	19
Learner tasks:	
✓ Group joining	21
✓ Small group discussion (Free)	22

The Task list associated with any technique can lead to a description of the system architecture. To bridge the gab between teachers and software engineering, first each task is represented by a pattern object. For example, task 1 (Creating a collaboration session) is directly related to the *Session Creation* object. Second, we apply the UML (Unified Modeling Language)[14]. This holds the basic information to drive both the Use Case diagram and the Class diagram.

2.4 The Fourth Layer (CSCL Tools)

In the last layer, the system will map each task in the list to one of the CSCL tools along with any configurations. The final result, what learners see, is the bundle of CSCL tools, tasks and configurations that the designer has included in his pedagogical technique. For example, task number 10 (Defining The Title) will be mapped to a static info box tool and the same will be done to all selected tasks. CSCL tools consist mainly of traditional tools, such as chat, audio, video, whiteboard, forum, but with some additional attributes. First, some tools (Text Chat, Audio,) should have different levels of control. Second, the communication direction should be specified. Third, the text chat should have two additional attributes: Numbering and Authentication. Besides that, some additional tools are needed for some pedagogical techniques, such as Info text box, Voting tool, Timing tool and Gestures tool. Info text can be divided into two parts: static and dynamic. The static component is needed to hold information written by the teacher during the activity design time, such as Title, Problem Related info, etc, while the dynamic component is needed to hold temporary information, such as guiding info, is broadcast during the runtime, such as guiding info. A teacher can show/hide some of these tools embedded in the main tool to synchronize the activities sequence during the run time. Figure 4 shows a screen shot of the prototype for a brainstorming tool.

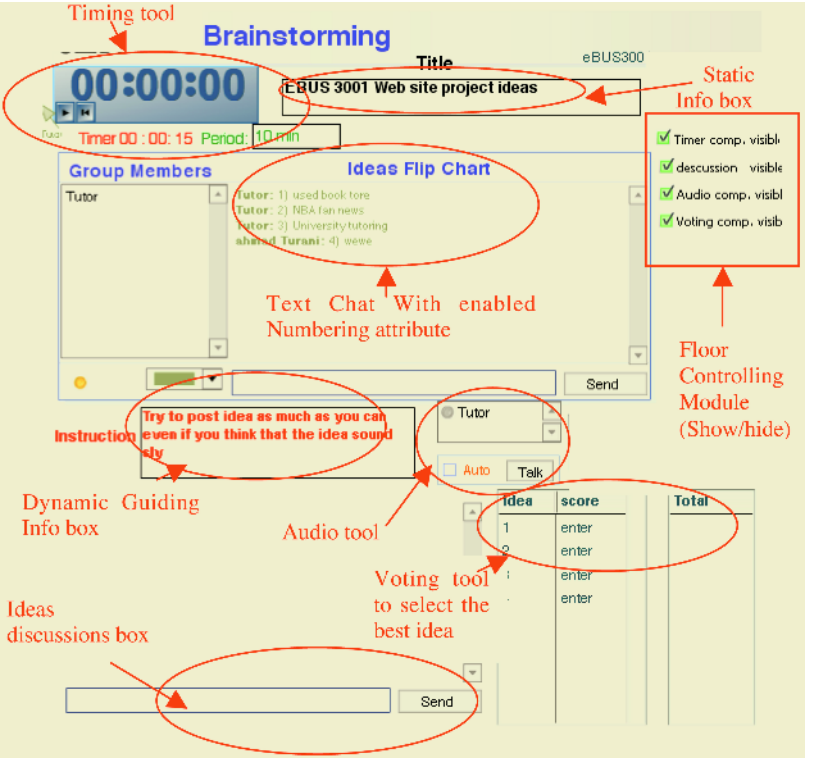

Fig. 4. Brainstorming tool

3 Conclusion

We have described the functional design of a framework to support teachers in selecting a suitable pedagogical technique and to support the construction of new ones. The theoretical framework has been successfully applied on ten well-known pedagogical techniques. The use of the application framework will also enable teachers to develop new pedagogical technique tools with minimum effort. A list of tasks that the teacher and learners should take before, during and after the collaboration session will drive the construction of the desired tool. For customization, it is possible for the teacher to change some of these tasks according to his needs. This approach will prevent losing the pedagogic structure embodied in the original learning activity while providing different kinds of tools for different techniques. We have only reported here the functional requirements and design of the framework. A prototype system is being built using the dotLRN Learning Management System and Macromedia's Collaboration Server. The two systems interact exchanging XML messages according to the IMS Learning Design specification.

References

1. Laurillard, D., *Rethinking University Teaching*. 2nd ed. 2002: Routledge Falmer.
2. Scardamalia, M. and C. Bereiter, *Computer Support for Knowledge-Building Communities.* The Journal of the Learning Sciences, 1994. **3**(3): p. 265-283.
3. Wasson, B. and S. Ludvigsen, *Designing for Knowledge Building*, in *ITU Report*. 2003, University of Oslo Press: Oslo, Norway.
4. Goodyear, P. *Patterns, pattern languages and educational design*. in *ASCILITE*. 2004. Perth, Australia: Australasian Society for Computers in Learning in Tertiary Education.
5. Boyle, P.G., *Planning Better Programs*. 1981, New York: McGraw-Hill Book Company.
6. Stacey, P., *People to People, not just people to content*. 2003, Vancouver.
7. Dimitriadis, Y.A., et al., *Component-Based Software Engineering and CSCL in the Field of e-Learning.* UPGRADE, 2003. **IV**(5): p. 21-27.
8. Paulsen, M.F., *The Online Report on Pedagogical Techniques for Computer-Mediated Communication*. 1995, nki.
9. Baumgartne, P. and I. Bergner, *Categorization of Virtual Learning Activities*. 2004: Hagen.
10. Fayad, M., D. Schmidt, and R. Johnson, *Building Application Frameworks Object-oriented foundations of framework design*. 1999: Wiley.
11. Asensio, J.I., et al., *Collaborative Learning Patterns: Assisting the Development of Component-Based CSCL Applications.* 12th Euromicro Conference on Parallel, Distributed and Network-based, 2004: p. 218-224.
12. Goodyear, P., et al. *Towards a Pattern Language for Networked Learning*. in *Networked Learning*. 2004.
13. Leo, D.H., J.I.A. Perez, and Y.A. Dimitriadis, *IMS Learning Design Support for the Formalization of Collaborative Learning Patterns*. 2004: Spain.
14. Fowler, M., *UML Distilled*. Second ed. 2000: Addison Wesley Longman.