A Document-Centered Architecture for Classroom Collaboration

Abstract: This paper presents a framework system which enables teachers in virtual and face-to-face classrooms to improve their information management during a classroom session. It also allows teachers and students to handle replica of course-related data in both the classroom and home environments. The approach is centered around a generic type of structured documents which supports cooperative concept mapping as well as more domain-specific domain representations and activities (e.g. collaborative model building or programming).

Keywords: collaborative learning, ubiquitous computing, classroom document management

Jens Hardings, Nelson Baloian
Departamento de Ciencias de la Computación
Universidad de Chile
Blanco Encalada 2120
Santiago, Santiago
Chile
Phone: +49-203 3791400

H. Ulrich Hoppe
Lotharstr. 63/65
47048 Duisburg, Germany
Phone: +49 203 379-3553

Mail: jens@hardings.cl
A Document-Centered Architecture for Classroom Collaboration

Jens HARDINGS(*), Nelson BALOIAN(*), H. Ulrich HOPPE(**)

(*) DCC - Universidad de Chile
{jharding,nbaloian}@dcc.uchile.cl

**) Gerhard-Mercator-Universität Duisburg
hoppe@informatik.uni-duisburg.de

Abstract. This paper presents a framework system which enables teachers in virtual and face-to-face classrooms to improve their information management during class- room session. It also allows teachers and students to handle replica of course-related data in both the classroom and home environments. The approach is centered around a generic type of structured documents which supports cooperative concept mapping as well as more domain-specific domain representations and activities (e.g. collaborative model building or programming).

1 Introduction

Although the main focus of most Computer Supported Collaborative Learning (CSCL) has been the development of environments supporting distance learning, still most of the learning activities are held in face-to-face sessions in traditional classrooms, where there are also enough challenges for designing computer based systems for enhancing and enriching this situation. The aim of a Computer-integrated Classroom (CiC) is to take the best aspects out of the traditional teaching using non-intrusive technology that mimics many of the attributes of a normal classroom, while providing the ability to enhance the classroom experience by adding tools that aid teachers and students to overcome their limitations and improve the learning outcomes. Computer technology in the classroom as a non-intrusive technology enriching the situation is not aimed to define new settings but promoting a change based on small steps: new roles, activities, material complement the traditional ones.

In this paper, we present a framework system that is built to provide basic capabilities to teachers in order to improve their ability to manage information within and across classroom sessions which complement human abilities such as decision-making with the powerful potential of information management systems, being able to move issues from the background to the foreground and back again as described by Weiser in [1]. The fields of use can vary, as well as the pedagogical methodologies used, because we do not constrain our system to a particular view of how the sessions have to take place. On the contrary, we encourage diversity and try to enable all forms of learning in our CiC, for example by enabling more interactivity, allowing to eliminate the one-way flow of information and most of all the one-way initiative that has been generally criticised within the educational system. Thus the CiC does not force a modification on the curricula or the way teaching is done within a classroom, but opens up new possibilities that can be exploited.
A situation in which collaboration among peers and between students and teacher are expedite forms of interaction, a CiC can help to improve the perceived and real benefit of educational efforts. Within a CiC it is possible to work with problems that have an ever increasing complexity, including real-time modelling, simulations, collaborative discussions and other issues.

Figure 1: Classroom scenario: teacher sending a file to students

2 The System Requirements

A CiC should define both a hardware environment and a specially designed software system, trying to integrate computers into the everyday classroom rather than to using laboratories. This purpose is evident in the implementation of the NIMIS classroom [2], in which the hardware, software, and even furniture design solutions are strongly tied one to another in order to achieve the desired propose. The NIMIS classroom was aimed to support teaching/learning in the first year of school. In this work we tried to design a CiC for supporting a more general situation, for learners being user of a computer in a more traditional way but without any special skills or computer knowledge.

This means, the system will support classroom settings varying from primary school classrooms to training situations of working people. For this scenario we can define a more standard hardware equipment in order to fulfil the needs for a more general propose CiC system. We do not want to impose exact nor minimum hardware requirements to allow the usage in a wide variety of scenarios. The CiC software system has been designed considering that the room will generally be equipped with an electronic blackboard [3] being operated by the teacher to present and manipulate learning material as well as document management operations like distributing or collecting learning material, homework or assignments. Additionally, the teacher and some or all of the students have personal computers or PDAs which are connected to a (probably wireless) Local Area Network. The network is either connected to the outside in order to have access to a server providing multimedia learning material and administrative information (like student enrolled in courses) or the server has to be located on the local network.
The system should provide tools to enable teacher and students to retrieve, save, exchange, distribute, and share multimedia documents in a classroom setting. These actions should be carried out swiftly in order not to interrupt the normal flow of the lecture. For achieving this goal, we defined a set of general propose semantic actions for teachers and students who are likely to be performed recurrently, in order to implement them in a way it would require from students and especially the teacher the minimum amount of time and action. These are shown in the use-case diagram of figure [2] The actions can be classified into two types:

**Document exchange:** Several of the common tasks within a classroom require the exchange of documents. This is the case when assigning tasks to work on, delivering homeworks or distributing complementary learning material. These actions can be performed for documents of any format and an example can be seen on figure [1].

**Document Sharing:** Synchronizing the use of a certain document by two or more users is achieved by coupling its content. This is only possible while working with special documents adapted for this environment.

The documents used within the CiC can be of any type, but we have also integrated the FreeStyler [4] into the architecture as a primary tool to manage a special kind of documents since FreeStyler allows the manipulation of documents in a way more according to what we think should be the structure of the learning material in a Computer enhanced learning environment.

The FreeStyler tool can be used during multiple working phases as preparation, creative meetings, presentation, postprocessing or wrapping up information Therefore, is serves also as an interface between face-to-face discussions and the documentation process. This is enabled by a cooperative visual language which offers a set of content objects to structure information. A content object combines a symbolic view with predefined interactions. It can be characterized as a template for a special type of information as ideas, concepts, decisions, addresses or internal and external links. Whereas several content objects rather define

![Figure 2: Use cases](image-url)
the category of information, others, i.e. the links, provide additional structural information and interaction features. Together with these content objects users can add handwritten input flexibly. Methods as concept mapping, mind mapping or MetaPlan are easy to perform with the FreeStyler. The external representations both enrich and influence the communication. The content type and functionalities of a FreeStyler document can be extended by defining "palettes" with new node and link types for a specific learning domain (see 4).

The documents may include a workflow description about how and when they should be distributed to the students, collected, where they should be stored, etc. This allows the system to offer useful default alternatives to the user, minimizing the input required to perform each task and keeping the focus on the session.

FreeStyler documents are stored in XML format. Beside the advantages of a standard format in general, XML makes it easy to retrieve information. In order to exchange the maps an integrated mail functions converts the maps to a picture format and send the content as attachments or the original XML files. FreeStyler-internal filters allow for separating types of information in order to focus on special aspects as decision or ideas.

The system will provide an interface to query the XML documents, both in their current state while being edited and off-line in the repository. This way, the teacher can monitor the work of the students over the document as well as grade the work or analyze the result of past sessions. The structure of the documents will help the teacher to look whether certain part of an assignment has been finished by all students, or if some keywords appear in the answers.

![Figure 3: a FreeStyler document showing a Java example](image)

Another aspect of the system is that it should provide feedback information about the classroom sessions in order to allow reflection and keep track of the state of the course. The information gathered should include the file used (opened, modified, created), coupling sessions, file transfers among others.
An important design criteria of the implementation is to develop the system in well defined modules and keep them independent defining a well structured interface between them, so that individual parts can be replaced in the future to enable more flexibility and keep up to date without affecting other parts.

3 The System Architecture

The system consists of several modules, which interact in different ways. The modules and their relations are shown in figure 4.

**Document Manager**: this is the central server module of the CiC, which provides authentication, directory and repository services for all other modules. The Document Manager can potentially access other external systems providing services like the ones described here, in a way that is transparent to the other modules. This is useful if the CiC should for example be integrated with existing systems or should interact with systems that offer more features for some services. The different services provided are the following:

- **Authentication**: before logging into a classroom session, every participant is authenticated and gets a certificate that can be verified by each of the other clients. The authentication mechanism allows a teacher to be able to trust the deliveries of critical information, like homework or other assessment elements.

- **Directory**: the Document Manager maintains a list of current classroom sessions and the participants, along with their current IP address. When signing on for a session, the classroom modules also receive this information and can continue to interact locally within the classroom, so the connection to the Document Manager does not get congested, since it is possible that it is a remote connection.
• Repository: the Document Manager is the interface for the clients to access the documents that are stored in the repository. It is possible to use any repository or courseware administration system as described earlier, by changing only the Document Manager.

Home modules: The home modules are a restricted version of the classroom modules (see next point) in the sense that they do not need several functions of the session management. They interact exclusively with the Document Manager to access the repository. These tools enable students to do homework or prepare examinations accessing the learning material and sending modifications back either as a delivery to the teacher or some group, or as personal annotations that may be accessed later.

Classroom modules: Within the classroom, two modules are used: one for the teacher and another is used by the students. The modules can access all services available through the Document Manager, and also communicate locally between themselves. This way, all authentication as well as retrieval and storage of documents is done against the Document Manager, but local communication and synchronization is done only locally.

Administration Tool: All administrative tasks like adding new users, defining roles and assigning belongings to courses are done using the Administration tool. This tool is basically a 1 to 1 interface to the capabilities of the Document Manager, so it also allows all interactions that could take place from any of the tools, but in an explicit manner. Within the other tools, in many cases the user does not notice that she is using the Document Manager services.

4 Adding new functionalities through palettes: an example with Java

A very powerful feature of Freestyler is the usage of ad-hoc pluggable modules called palettes to define new functionalities. Each palette contains elements called nodes that can be placed in the documents, and also can contain different types of edges that connect different nodes. In this way, different models can be used for different learning subjects or complexity levels. As an example, a palette for supporting the teaching and learning of programming in Java was developed. The java palettes contains three types of different nodes and one link type. The node types are:

Java Code Node: a node which contains and shows a Java program.
Program Input Node: a node which contains (or receives) and shows the standard input for a java program.
Program Output Node: a node which shows the standard output of a program.

In order to associate an Output Input Node to a Program node it is necessary to draw a link from the first to the second one and in order to associate an Output Node to a Program Node the link must be drawn from the second to the first one. Figure 3 shows a freestyler document with the java palette. The palette is shown at the right hand side of the screen and contains the 3 kind of nodes which can be added to the document by drag-and-drop operations. There is an icon for creating links and another one for deleting them. The document of the example shows a Program node connected to both an Input and an output node.

The palette also defines some functionalities, which on the figure they appear on a pull down menu. These are the following:
Create Java Code: this is an alternative way to create a java node, which may be more comfortable for some electronic boards, where drag-and-drop operation may be not so easy to perform (like on a SmartBoard). Nodes are created without any content.
Add Java Code: creates a Java Program Node with content taken from a java program file. After selecting this operation a list of local and remote files containing java code is displayed for selection. The remote files are taken from the course container on the server.

Edit: change the font for displaying the java code, the input and output

Compile: Compiles the code contained in a selected Java Program Node

Execute: executes the code of an already compiled Java Program Node

Using the CiC environment and the Java palette, teachers can show how to write a Java program, run it and show its output in a swiftly way. Since freestyler document can include handwriting and elements from other palettes (like for example, a discussion palette) a pedagogically meaningful documents can be created during the lessons and stored as learning material for the course. With the ”distribute” functionality of the CiC environment documents containing program fragments or programs with errors can be distributed to the students for completing or correcting them as homework or assignment. By establishing a coupling session and synchronising the documents of a certain student with the one on the electronic board the teacher can enable a student to show her solution to a certain problem proposed earlier.

5 Related work

Most efforts done by now to introduce computer technology in the classroom were focused in the hardware configuration. In fact, many ”electronic classrooms” arrangements have been set up in schools and universities. Many of these are designed to support remote lecturing by videoconferencing, like the one shown at [6] and the electronic classroom used in ITM [7]. Other efforts have been directed to capture a lecture and storing it for later review like the one presented in [8]. Norman [9] developed a system for a CiC-like architecture to manage the syllabus of a course and its multimedia-based learning material but it does not support the sharing of this material nor the dynamic changing of the course content that may occur during the lectures.

The authors of this papers exemplified the principal ideas of a CiC in the COSOFT environment [10]. Recently was adapted to ”early learning” in a primary school classroom [2] as well to academic lecturing [11]. Some issues about awareness in the CiC where tackled by [12].

6 Conclusions

The framework presented here is designed to aid teachers in managing the documents to be used in classroom and related activities in a non-intrusive way, with a clear focus on face to face classroom work as opposed to distant learning scenarios. At the same time it enables processing of digital media to improve the ability and quality of the data management that a teacher has to overcome nowadays.

The focus is centered on internal course issues rather than a broader view over several courses, and the system is flexible enough so that it does not force any teaching/learning style but rather accommodates to the teacher’s needs. It can therefore be used within an existing broader administrative system without much change, but as a complement that operates within the classroom, even accessing external resources as needed.
References


