T. Koschmann, D. Suthers, & T.W. Chan (Eds.), *Computer Supported Collaborative Learning 2005: The Next 10 Years!*
Mahwah, NJ: Lawrence Erlbaum Associates. 

# A Lightweight Approach for Flexible Group Management in the Classroom

**Markus Kuhn, Marc Jansen, Andreas Harrer, Ulrich Hoppe**
Institute for Computer Science and Interactive Systems
University Duisburg-Essen
{ kuhn, jansen, harrer, hoppe }@collide.info

**Abstract.** In this paper we describe a session management system for setting up various collaborative classroom scenarios. The approach is addressing the additional workload of administrating classroom networks on the teacher, which is an important aspect for teachers' willingness to implement technology enhanced learning in schools. The system facilitates preparation of classroom scenarios and the adhoc installation of networked collaborative sessions. We provided a graphical interface, which is usable for administration, monitoring, and for specification of a wide variety of different classroom situations with group work. The resulting graphical specifications are well suited to be re-used in the more formal learning design format IMS/LD; this is achieved by a automatable transformation of the scenarios to LD documents.

**Keywords:** Collaborative classroom scenarios, lightweight classroom orchestration, learning design, shared workspaces

## INTRODUCTION

In the area of technology enhanced learning, the term “learning environment” (LE) is usually associated with a virtual or computational system that supports learning in a specific coherent way. There are domain orientated environments, sometimes called microworlds, which support specific semantic representations and processing mechanisms, but also general “learning platforms” that aim at organisational, communication and archiving support for learning communities. In spite of the differences between these concepts, neither one challenges the conventional assumption of the LE residing on one or more computers. In the European project NIMIS (1998-2000), we have tried to break with this understanding by identifying the LE with the classroom. In such a computer-integrated classroom (Baloian et al., 2002), a mixture of traditional (or natural) forms of communication and media may co-exist with digital media serving different functions which may be partly identical to traditional media use and in other parts actually qualitatively new. We have used the term “digital mimicry” to characterise interactive digital media functions which mimic traditional forms such as the use of a pen based big electronic display instead of a chalkboard (cf. Hoppe, 2004). Interactive simulations are a typical example of a genuine new media function which is bound to the digital modality. However, there is a general added value that we expect already experience from combining digitised traditional media (e.g. scanned-in paper notes) with digital mimicry applications and real new media into a new form of digital information flow with specific forms of recording, re-use, re-enactment and extension/modification.

In the more recent EU project SEED (2001-2004), we have tried to create classroom innovation using interactive media together with a group of teachers. These teachers were introduced to the new types of hardware and software (mainly annotation and modelling tools) and were invited and supported in appropriating these for their own teaching. This has led to interesting software extensions and blueprints for teaching in areas so diverse as biology, mathematics and language studies. The focus of these activities was clearly on representational tools, not so much on a general communication infrastructure. Indeed, we found that existing school intranets are still too poorly developed in terms of availability, maintenance and coherence to get the full added value out of the digital enrichment of the classroom in terms of organisational memory functions. Due to this lack of infrastructure, we have not been able to fully implement our concept of a computer-integrated classroom in the various schools we worked with. Yet, we have explored the general feasibility of using new devices such as low cost graphics tablets for hand writing as well as big interactive displays, tablet PCs as well as PDAs in domain specific applications with a special focus on collaborative use. Importantly, the specific applications were not predefined but designed and put into practice by the teachers. In almost all of these experiences, there was a general demand for setting up classroom networks with flexible grouping and archiving/recording mechanisms with an additional time effort being small enough to be justified for a 45 or 90 minute lesson. This gave rise to the implementation of an “ad hoc session manager” which has recently been finished and just undergone the first functional test.

There is an obvious similarity between this session manager and the idea “ad hoc networking” in learning scenarios with wireless and mobile devices. Chang, Sheu & Chan (2003) describe such an approach to setting up “ad hoc classrooms” with mobile devices, e.g in outdoor activities. This approach provides a standard set of communication functions (downloads, uploads, broadcasting etc.). Our goal is to facilitate more explicit and flexible structuring of the learning group based on the precondition of having a surrounding minimal school infrastructure with at least a pre-installed network in the classroom and potentially some stationary computers. Here, teachers are able to set up different collaborative scenarios with minimal effort. It turns out that the concrete scenario specifications can even be understood as a kind of explicit “educational modelling”.

## REQUIREMENTS FOR A FLEXIBLE CLASSROOM MANAGEMENT SYSTEM

Administrating computer based group work of a whole class means additional effort for the teacher. He has to subdivide the class into groups, each using several computers. He has to initiate the work with prepared material and distribute it to the students. In the following phase students work collaboratively on given topics, aware of the work progress of their group members. They share, merge, save, deliver and present their results. The teacher is interested not only in archiving the “final products” but in observing, supporting and protocolling the learning process of each group. Therefore he wants to “visit” the working sessions of his students.

From a requirements engineering perspective (Pohl 1993) we concentrate on the following functional requirements: Concerning the collaboration, the software has to facilitate a) grouping and rearranging of cooperation, b) synchronizing of common workspaces/learning places, c) monitoring, protocolling and archiving. Additionally d) presentation and e) (re-)use of results, i.e. the specified learning scenario as well as the results of the student work, are of high importance for the use in school. The diagram in figure 1 which illustrates the usage of our classroom management system includes use cases representing these functional requirements.

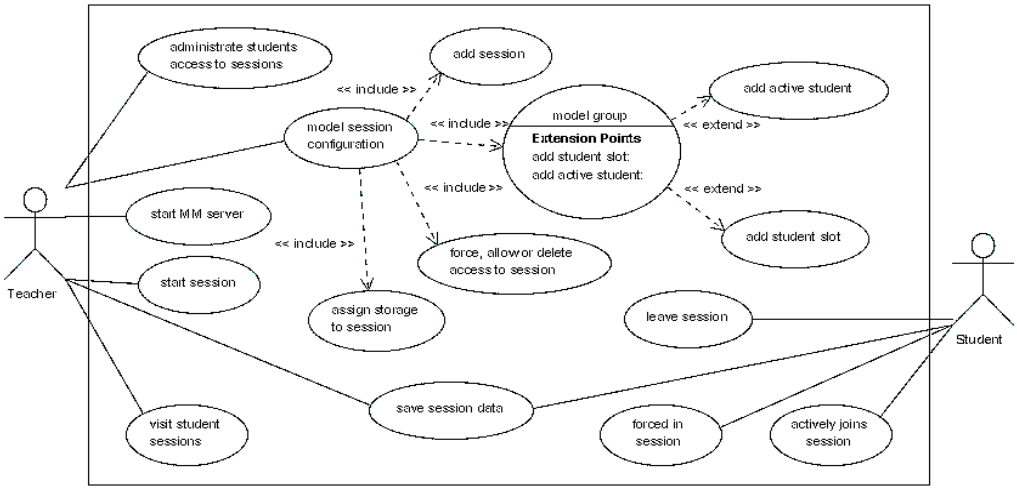


Fig. 1: Use case diagram: session managament for group work

Besides these functional features teachers will only accept a software tool if it fulfills other (non-functional) requirements concerning handling, performance, reliability and added value. Our software is designed to support the teacher in setting up computer based group work *ad hoc* and in preparing sophisticated forms of collaboration *in advance* and it *simplifies* to support, control, manage and rearrange the cooperation between the students and to access the produced data. This allows the teacher to use ICT in a new, different way increasing the benefit of group work, opening up possibilities to arrange cooperation among his students, to start and accompany learning processes, to integrate the results of the students in the current and following courses.

## SYSTEM DESIGN AND IMPLEMENTATION

We implemented the proposed lightweight approach based on the mentioned requirements. Since we already use the collaborative modelling system Cool Modes (Pinkwart 2003) in school for mathematical modelling, computer science lessons and graphical argumentation (Harrer et al 2003) it was an obvious option for us to implement our system on this platform basis. In order to allow for synchronous collaboration among participants of a learning or modelling task, the Cool Modes environment is based on a framework that easily supports the extension of standalone Java applications to collaborative applications. This framework, called MatchMaker (Jansen, 2003) provides the possibility to share the inner data structures of Java applications and logging and replay functionalities to protocol additional information for the evaluation of collaborative work. It is important to stress that this func-

tionality is not application dependent but provided by the framework that supports the collaboration. Therefore, these features are available with all applications that use this framework.

The Cool Modes framework was used with two different intentions in our scenario. On the one hand, it is used by the students to perform their collaborative modelling task, and on the other hand, the teacher used it in order to orchestrate the group work. Basically, a teacher could also model the groups with Cool Modes but use another, MatchMaker based tool, for the collaborative task.

On top of the Cool Modes platform we have developed a graph based visual language for the representation of the classroom networks. This visual language represents graphs with the following nodes and edges:


Figure 2: Elements of the visual language for modelling group work

The user-node represents a *client*. Visible information on this node are the name of the user or partners if students share one computer, the login name, the host he is working on and the IP address of the host. With the help of a *session node*, the teacher has the possibility to model and control (start/stop) sessions, see how many clients currently are connected with the session and the unique name of the session. A *database node* allows the teacher to store the data of a connected session at a previously defined place. This might e.g. be helpful for the teacher to collect the learning objects created by the students. Last but not least, the visual languages provides a fourth node, a *slot node* which acts as a wildcard for clients. Therefore, this node allows the teacher to orchestrate the different groups before the students enter the classroom and log into the system. Furthermore, this so-called *slot-node* could be filled with unique parameters of a certain user, like his login name. If such a unique parameter is provided with this node and the student who fits to this unique parameter logs into the system, he is automatically identified with this node and the node is replaced by the client node representing the student. If no unique parameter is provided, the teacher has to drag the client nodes to the slot nodes in order to make this matching. Basically, the slot-nodes are used by the teacher to model a collaborative learning design and those nodes do also provide the possibility of reusability.

Furthermore, the visual language provides three different edges in order to connect the different kinds of nodes. Basically, we have two edges that allow to connect a slot node or a client node to a session. On the one hand there is the *force-edge* which results in an automatic join of the client in the session and on the other hand we provide an *is-allowed-edge* in order to show a certain client is allowed to join a certain session. Another edge connects the database node with a session node in order to store the session data at the previously defined place.

One of the major goals for this approach was to ease teachers effort. He needs information about the computers that are available inside his classroom. Since the automatic detection is a non trivial task in the absence of a central server that can provide a naming and lookup service, we decided to use a multicast architecture to allow for flexible communication patterns. By this approach, each client only needs to provide two basic features on the multicast layer. On the one hand, the client must be able to receive multicast messages sent by the server to inform the client on which host the MatchMaker server is running, and on the other hand the client itself has to be able to send a multicast message around telling the server that there is a MatchMaker client available at a certain host. With the help of these multicast messages, the server can set up the topology of available MatchMaker clients, and the clients are informed where the server is located. Once the topology and the location of the server is communicated among the classroom network, the communication between the clients and the server is switched from multicast to Java RMI (remote method invocation). All of the higher level task, e.g. forcing a client into a session or getting the available sessions for a client, are then implemented on the RMI level.

## DETAILED EXAMPLE AND FIRST EXPERIENCES OF USE

The group management tool was applied in a German secondary school in a computer science course of twenty 12 graders. The computer lab consisted of 15 + 1 networked computers. Our tool was not only used to administrate the group work but also to evaluate the work process and the results of different forms of group work. Each group work scenario can be classified by the degrees of freedom it offers students to structure their collaboration and their work on the task. Our ongoing research work will have a closer look on the influences of these two dimensions on the learning process of students and it outcomes.

In the described setting two different groups worked on a UML class diagram for a computer based monopoly game. Whereas the second group had the task in common the first one was divided into two subgroups with predefined division of labour. Additionally in this group one student acts as a coach to combine and present the merged parts. Figure 3 shows settings organization prepared in advance through the group management tool. The

figure shows the two sessions for group 1 with four respectively two students. Another student acting as a coach *is allowed* to join both sessions of group 1 and the final presentation session. The other students are *forced* to join the session of group 2. These different modes are indicated by differently coloured edges.

In the first minutes the teacher controlled the group management while the students logged themselves on. Students not automatically assigned to their session by (their correct) log-in name where placed and linked by hand. After the teacher had initiated the group work a short introducing phase of 15 minutes started in which basic classes were defined collaboratively in both groups. These basic models the students elaborated in shared workspaces using a toolbox for UML class diagrams. The modelling work was done by the two subgroups of Group 1 (n = 9) following their distinct orders in a concentrated and goal-oriented style. Out of technical problems the coach was not able to do his mediating work and could not merge the partial models. Group 2 ( n = 11) began in a more "anarchistic" way. After several minutes they used a new start to structure and distribute their work. Now they decided to use separate areas of the shared workspace to work on subtopics in informal subgroups. The teacher observed their work joining the sessions, saving versions of the growing models. Finally students presented the results by joining the presentation session used for visualisation with a data projector.

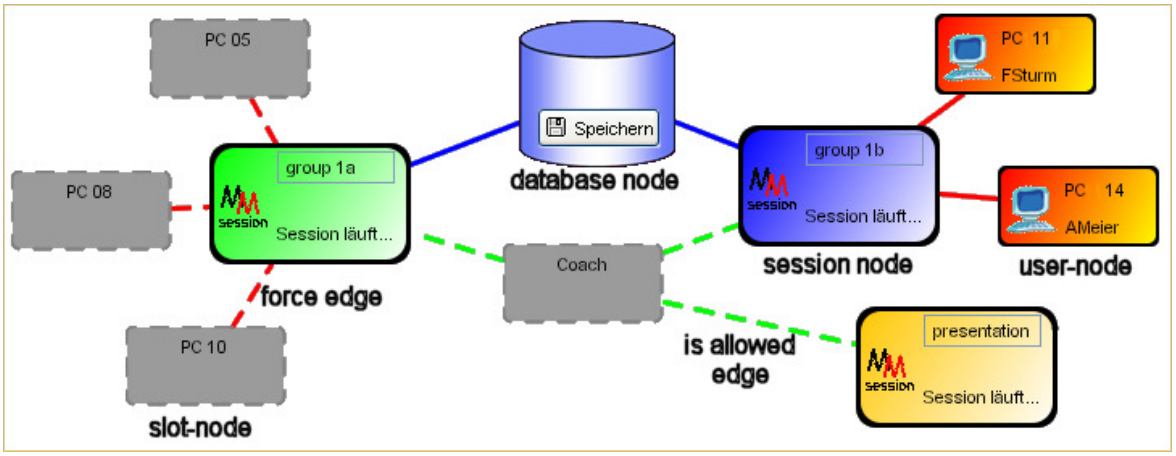


Fig. 3: Modell representing the arrangement of computer based group work

We observed that students of both groups were able to model relatively complex class diagrams in a short amount of time. They could cope with both arrangements very well, working effectively and competently. Using a questionnaire we found out, that the students were widely satisfied with their results (in a scale of 1 up to 5 both groups had an average value higher than 4) and with the collaboration (group 1: 90 %, group 2: 82 %). They enjoyed to work together directly and in a distributed manner on a common task, subsuming their own ideas without restricting each other. They were aware of the ongoing group work. Group 2 enjoyed the freedom to structure and distribute their work, communicating directly before modelling on the computers. In comparison to group 1, a qualitative difference is to be found concerning satisfaction and intensity of work. Whereas in group 1 all students had the feeling of having worked homogenously, in group 2 four students (36 %) pointed out that some had worked less than others, whereas 45 % where satisfied with the collaboration – these may be indications for unresolved conflicts during the negotiation.

## LEARNING DESIGN BY EXAMPLE

The concrete collaborative scenarios created by the teacher have a high potential for further usage: they can be considered as templates of learning design that may be used for different forms of modelling. This can be done by abstracting concrete assignments of learners to role definitions. This is already prepared to a great extent in our system by providing the "slot nodes" as placeholders for concrete group participants.

The definition of learning designs attracted a lot of interest in the last years and resulted in proposals of educational modelling languages, such as ClassSync ML, EML and its successor IMS Learning Design (IMS/LD, IMS 2003) specified by the IMS Global Learning Consortium. The definition of learning designs for collaborative scenarios tends to be much more complex than the designs for individual learning, because different group situations and roles therein have to be specified. Related work, such as (Hernandez et al. 2004) showed that some aspects of complex collaborative designs (also called Collaborative Learning Patterns) are not represented properly in IMS/LD and extensions are necessary. In the light of this we think that the direct use of the XML-based IMS/LD format for specification by the teacher, who usually is not an expert in computer science specification, is a major challenge for the learning designer. This brought us to the idea to use the described tool also as an editor for collaborative learning designs using IMS/LD as output format. The teacher can then specify his learning de-

sign "*by example*" creating a concrete visual model, instead of using the machine-level textual format of IMS/LD. We defined a mapping from our scenarios to IMS/LD document :

1. Each session within the classroom scenario is mapped to an IMS/LD "learning-activity", in our example "1a", "1b", "presentation", using the session's textual description for title and description of the activity
2. Each client node or slot node is mapped to an IMS/LD role of type "imsld:learner", in our example "coach", "Fsturm". Since roles of a learning design are instantiated at runtime and thus instances cannot be specified in the document, the client nodes of CoolModes are also abstracted to roles.
3. The teacher who is implicitly present in the scenario (but not in the model) is represented in the learning design as a role of type "imsld:staff"
4. The whole classroom scenario graph of our visual language format is mapped to an IMS/LD "act"
5. For each learner (client or slot node) an IMS/LD "role-part" is created within the act with the respective "role-ref"; this role-part includes a "learning-activity-ref" to the respective learning activity (in our format a session node, see above) for each edge connecting the learner with the session node. In case of a "force edge" there is one session available as learning-activity-ref. The "role-part" for the teacher includes every learning-activity to show his potential participation in every session, in our example "coach" with role-part learning activity "1a", "1b", "presentation".

At the moment we only can define one-act scenarios directly using our MatchMaker/CoolModes environment. More complex learning designs with a sequence of acts are obviously a desirable target to enable richer classroom scenarios to be defined and conducted. We plan to extend our IMS/LD export so that we can combine multiple models in our system to a temporally ordered sequence of acts and thus a full-fledge learning design "play". The teacher just specifies the configurations separately and connects them with a specific sequencing. Even more convenient is "specifying by example" the whole learning process model by letting the system record her specification process over time. Multiple workspaces and a logging mechanism of the modelling process (Jansen 2003) are already available for our system and will be put to use for this "multiple act specification by example" in our next steps. The exported IMS/LD-format can easily be attributed with e.g. "learning objective" in a simple editor we have developed to enrich the design with more pedagogically oriented information.

## CONCLUSION

Utilising technology enhanced classroom scenarios in school practice usually burdens the teacher with technical administrative effort in setting up the scenario. We addressed this challenge through a lightweight approach for flexibly setting up collaborative classroom sessions. Based on the requirements to reduce the teachers' effort we implemented a classroom management system that can be visually administrated without deep technological knowledge. The resulting visual models can be utilised by the teacher for the preparation of scenarios at home as well as for adhoc classroom setup. The visually specified scenarios are a step towards re-usable learning designs, which is supported in our architecture by a mapping of concrete scenarios to IMS/LD documents which could also be used for different classes, courses, or even other software platforms.

## ACKNOWLEDGMENTS

We thank our students Dominik Schweers, Adam Giemza, Björn Hassing, Thomas Ulken for their implementation of the flexible group management system and Tilman Göhnert for the IMS/LD export functionality.

## REFERENCES

Baloian, N, Berges, A., Buschmann, S., Gassner, K., Hardings, J., Hoppe, H.U., Luther, W. (2002). Document management in a computer-integrated classroom. In Proc. of . CRIWG 2002. Berlin et al.: Springer. (35-44).

Chang, C.Y., Sheu, J.P., Chan, T.W. (2003). Concept and design of ad hoc and mobile classrooms. Journal of Computer Assisted Learning, vol. 19, (336-346).

Harrer, A., Pinkwart, N., Lingnau, A. (2003): Of Birthdays and Earthquakes. In "Community Events - Communication and Interactions: Proc. of Computer Support for Collaborative Learning", InterMedia, Bergen (Norway)

Hernandez, L.D., Perez, A.J., Dimitriadis, Y. (2004) IMS Learning Design Support for the Formalization of Collaborative Learning Patterns. Proc. of ICALT 2004, IEEE Computer Society, Los Alamitos, CA. (350-354).

Hoppe, H. U. (2004). Collaborative Mind Tools. In M. Tokoro & L. Steels (Eds.): *A learning zone of one's own – sharing representations and flow in collaborative learning environments* Amsterdam, IOS Press. (223-234).

IMS (2003) IMS Learning Design Specification (V. 1.0), IMS Global Learning Consortium, Burlington , USA.

Jansen, M. (2003) MatchMaker TNG – A Framework to Support Collaborative Java Applications. In U. Hoppe, F. Verdejo & J. Kay (eds.): Proc. Artificial Intelligence in Education. Amsterdam, IOS Press.

Pinkwart, N. (2003). A Plug-In Architecture for Graph Based Collaborative Modeling Systems. In U. Hoppe, F. Verdejo & J. Kay (eds.): Proc. of Artificial Intelligence in Education, Amsterdam, IOS Press.

Pohl, K. (1993) "The three dimenstions of requirements engineering". Proceedings to the Conference on Advanced Information Systems Engineering (275-292).