MODELLINGSPACE: INTERACTION DESIGN AND ARCHITECTURE OF A COLLABORATIVE MODELLING ENVIRONMENT

Nikolaos Avouris, Meletis Margaritis, Vassilis Komis, Angel Saez and Ruth Meléndez

ABSTRACT
This paper describes the architecture of the *ModelllingSpace* open problem-solving environment. Modelling-Space is a new learning environment supporting synchronous and asynchronous collaborative problem solving by students at a distance. We describe here the key design decisions of the ModellingSpace software and in particular issues related with support for students with heterogeneous sets of primitive entities, control of interaction and dialogue, representation of the entities and models in a format that permits exchange of primitive material, as well as architectural considerations of the distributed application relating to network bandwidth limitations. The paper provides also an outline of server-side tools designed for supporting a community of students, users of the ModellingSpace environment.

KEYWORDS
Collaborative learning, computer-supported collaborative problem solving, synchronous collaborative software, groupware

INTRODUCTION

ModellingSpace is an open learning environment that supports real-time and asynchronous collaboration of small groups of students engaged in problem solving. This environment has been designed and built, based on experience with existing previous tools, like *ModelsCreator 2.0* (Komis et al., 2001), which have been used in the past for teaching multi-disciplinary science subjects in various educational settings, see Komis et al. (2002), Fidas et al. (2002b), Margaritis et al. (2003). The architecture of the ModellingSpace distributed environment is presented in this paper. In particular we discuss issues related to interaction design, support for students with heterogeneous sets of primitive entities, control of interaction and coordination mechanisms built, as well as architectural considerations of the distributed computing limitations. The paper provides also an outline of the server-side tools designed for supporting a community of students, users of the ModellingSpace environment. A number of evaluation studies of the early prototypes have taken place recently in which pupils and teachers of Greek High Schools and undergraduate University students have participated, while more experimental use of software prototypes is in progress; see Margaritis et al. (2003). The main concept of ModellingSpace development has been based on experience with existing previous tools, developed during recent years and tested in the field. The functionality of these original tools has been enhanced and re-implemented. In addition, new tools have been developed and integrated in the new ModellingSpace environment, related to analysis of collaboration and problem solving, discussed in Avouris et al. (2003b).

MODELLINGSPACE DESIGN

This section presents the main aspects of the architecture of the ModellingSpace (MS) system together with the main technological decisions of the system that has been developed.

MS is a software environment that supports individual and collaborative building of various kinds of models. It includes tools that permit building and editing of primitive entities, building and exploring models that are made of primitive entities, synchronous and asynchronous interaction of students, collocated or at a distance, who collaborate in building models out of primitive entities and tools that support analysis of modelling activities. The open character of MS means that students have access to an open set of primitive entities that can be used for building these models.

Key design decisions

The main decisions concerning the architecture are related to the development of the synchronous and asynchronous collaboration functionality, as well as the integration of the meta-cognitive analysis tools in the architecture. The decisions related to the architecture of the stand-alone modelling tools (*Models editor* and *Entities editor*) are based in some extent on existing *ModelsCreator* functionality and design.

Synchronous and asynchronous collaboration for modelling is a case of computer supported collaboration based on the concept of shared artefact represented in a work surface (Dix et al., 1998). In contrary to other collaboration applications in which emphasis is in communication (meeting support, argumentation tools, decision making etc.) in this case the distant partners collaborate mainly by sharing the model in the asynchronous collaboration mode and act on a shared work surface in the case of the synchronous collaboration mode. Our case is similar to collaboration support environments involving development of artefacts, like shared text editors, collaborative design environments etc, in which the partners share the view over the artefact to be developed, which thus becomes a *cognitive space*. A key requirement is therefore to create infrastructure for sharing a view of the model in synchronous modelling activities and additionally support direct communication among the participants. In figure 1 the notion of feed-through the artefact is shown, where one participant's manipulation of shared objects can be observed by the other participants. This communication through the artefact can be as important as direct communication between participants, as observed in Avouris et al. (2003a) and Fidas et al. (2002). Finally the size of the group of collaborating partners and the setting of collaboration in terms of technical specifications of equipment to be used (e.g. network bandwidth) and location of participants are essential characteristics of the problem to determine the architecture.

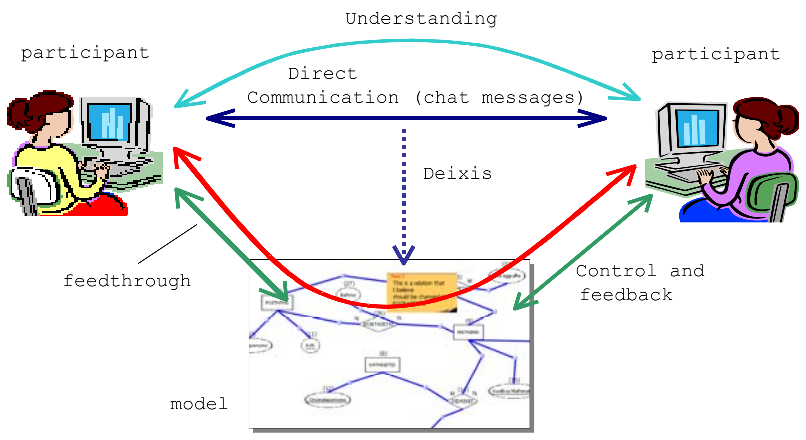


Figure 1: Collaborative-modelling framework

Direct Communication

Various architectural decisions are related to this framework. Considering that the collaborative activity is done mainly between partners at a distance the direct communication mechanism has to be defined. The alternative options have been (Preece et al., 2002):

· *Voice communication* (video phones, video conferencing, media spaces)

· *Text-based communication* (instant messaging, collaborative virtual environments (CVEs), chat rooms)

From these two alternatives the second one has been selected for a number of reasons. Video does not seem to bring any benefit in this context, taking in consideration the current serious limitations of videoconferencing systems (Preece et al., 2002). Additional problems with audio are: logging of voice and transferring it in text form, necessary for meta-analysis and classification of events, is a technically difficult task, there is lack of adequate bandwidth for voice and video communication in most school environments. Also voice or video necessitate use of special equipment, often not available in school lab workstations. In addition, difficulty with distinguishing the identity of the speaker from a group through his/her voice has been reported in various studies.

On the other hand, use of typed messages through instant messaging technology seems to have certain advantages. Transmission of text messages can be done through low bandwidth connections. Students of typical age group of ModellingSpace users (10-16) seem to have developed strong typing skills and instant messaging use habits, since they are frequent users of this technology through various media (SMSs, chatrooms etc.). Finally, the implementation of structured dialogue techniques, through use of dialogue opening options in a chat tool is easy in this case. In addition, if voice communication needs to be used, this can be done using tools external to the MS environment (e.g. voice over IP or telephone connection), especially since such services are made gradually available to schools.

Shared activity space design

One important decision is related to the design of the shared activity space. According to Suthers (2001) the degree of coupling between the activities of different users and the state of applications used by those users can vary. The alternatives according to Suthers are:

· *Strict WYSIWIS* (what you see is what I see). of the activity in the workspace of coordination, provides all users with exactly the same view and controller states. Strict WYSIWIS can support effectively the collaboration of two to three users whose activities are tightly coupled. An example of such environment is NetMeeting.

· *Relaxed WYSIWIS* does not insist that the state of the view be exactly the same, so different users can scroll to different viewpoints and perform their own operations, such as moving objects, until a model change forces an update in the view

· *Model level coupling*, guarantees that the partners share the same model but the view might be entirely different, for example one can view the model as a graph, or run a simulation of the model independently of the others.

From the requirements of ModellingSpace a mixture of alternatives is provided. A strict WYSIWIS is allowed in the main model-editing window. We believe that activity in this area should be faithfully reproduced in all participants' workstations. This is because most of communication and reasoning is based on this shared viewpoint, which becomes the main grounding mechanism of dialogue and through which eventually common understanding can occur. Deviation from this results in confusion of partners since misunderstandings can be generated due to different views when partners are allowed to scroll to different viewpoints, while no strong coupling of the shared view and the direct communication can be achieved. However all additional operations outside this shared workspace, e.g. relating to browsing of themes of study, saving of the model and running graph tools with alternative representations

of the built model, are performed independently by partners involved (a *model level coupling* approach according to Suthers(2001).
A consequence of this design decision can be that high volume of information may be transmitted to participating peers due to the strict WYSIWIS of the shared workspace requirement. A possible solution to this problem is to use replication of the environment in all workstations and synchronization of the workstations states through control messages. This approach has also been suggested by MatchMaker (Tewissen, 2000), Belvedere (Suthers, et al., 1997) Habanero (Chabert et al., 1998), E-slate (Koutlis et al., 1998) etc.
Even this solution however is not satisfactory for an open environment, like ModellingSpace. In our case the models building blocks, i.e. primitive entities (containing often large collections of image files) can differ in peers' workstations. This is due to changes that can occur even during modelling activity, as new primitive entities may be imported from the common repository or received through asynchronous interaction. So in case that a primitive entity is used by one of the partners during modelling, a need arises to transmit possibly large multimedia files to collaborating peers in order to synchronize the peer applications. This can create disruption in smooth collaboration to all collaborating partners, see Fidas et al. (2002b).
A solution proposed for this problem is to send only light control messages to the peers (chat and change of state), including the structure of new primitive entities, while the heavy multimedia files associated to these entities, if required, are sent through the server directly to the requesting peers, without creating disruption to the rest of the group. This hybrid protocol is discussed in more detail in the next section.

Coordination mechanism design

One other important decision is related to the design of a coordination mechanism for the activity in the shared workspace. In computer-supported collaborative environments, like in face-to-face group interaction, a mechanism is needed to control the floor in terms of communication and action in the common activity space. Various alternative coordination mechanisms have been proposed; see Dix et al. (1998) for a survey and a discussion for alternative approaches. Some of them impose no particular control, i.e. any member has his/her own pointing device and can manipulate objects in the activity space or write on the whiteboard. This can create coordination problems with the participants ending up in writing one on top of the other and cancelling each other's actions. Other architectures propose floor control mechanisms, involving the existence of a coordinator, various floor control protocols, like round-robin etc, or protocols of explicit request and concession of the floor. For instance inactivity of the floor owner for more than a certain time can release the floor.
In the case of ModellingSpace we propose a coordination mechanism which involves the notion of the *Action Enabling Key*, owned by one of the participants at any given time. This key owner can then act in the shared workspace, while the rest just observe this activity. This mechanism is supported by key request, key accept, key reject functions. Experiments with this floor control mechanism, see (Fidas et al., 2000) and (Komis et al., 2002), demonstrate that it improves reasoning about action, as partners need to reason and negotiate during key requests.
This coordination mechanism in absence of a coordinator is based on a *pass-the-key protocol*, or in presence of a coordinator can take the form of any protocol imposed by the coordinator who exercises authority through this mechanism. This flexibility is suitable for educational environments like ModellingSpace, where in various settings, educators or researchers wish to use different coordination procedures.

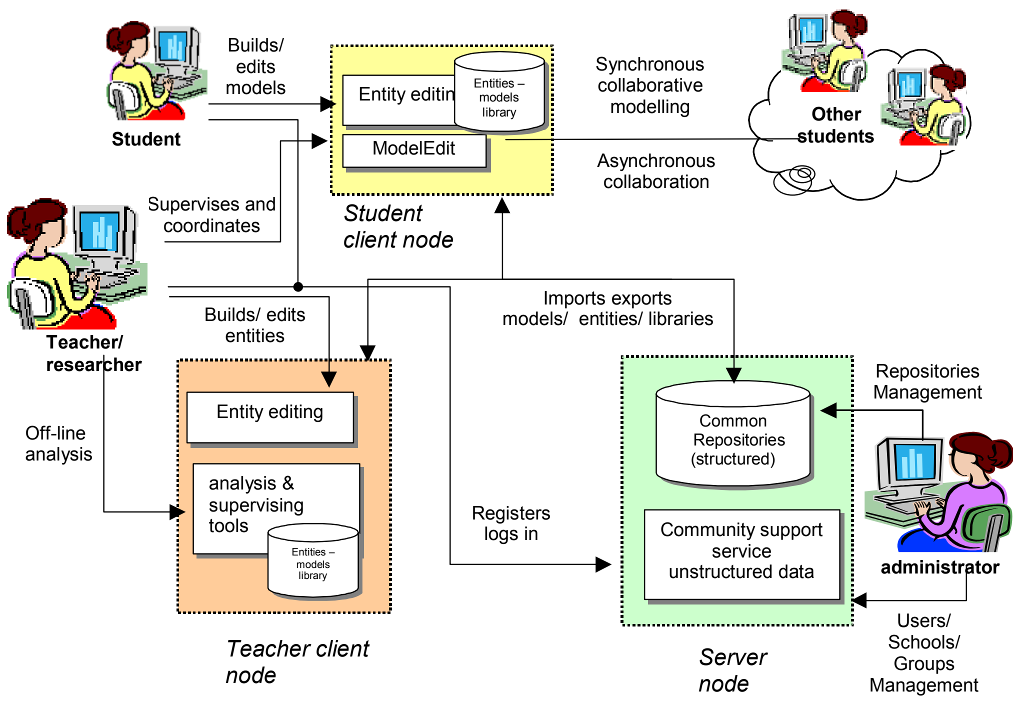

Student
Builds/ edits models
Entity editin
Entities – models library
ModelEdit
Student client node
Synchronous collaborative modelling
Asynchronous collaboration
Other students
Supervises and coordinates
Teacher/ researcher
Builds/ edits entities
Imports exports models/ entities/ libraries
Off-line analysis
Entity editing
analysis & supervising tools
Entities – models library
Teacher client node
Registers logs in
Repositories Management
Common Repositories (structured)
Community support service unstructured data
administrator
Users/ Schools/ Groups Management
Server node

Figure 2: Overview of the architecture: actors and nodes

OVERVIEW OF MS SYSTEM ARCHITECTURE

Based on the design ratio described above, the ModellingSpace (MS) software is defined. This takes the form of a suite of interconnected tools to support collaborative modelling learning activities. The main actors of MS, according to this are the student and the teacher (called learning actors). The latter incorporates many roles: The coordinator/ facilitator of collaborative modelling, who can remotely or locally co-ordinate, coach and supervise modelling activities through the relevant supervision tools. The analyst/researcher who uses the analysis tools in order to study and identify patterns of modelling learning during modelling activities (in on line or off line mode). The creator of primitive modelling entities who uses the editor for building new modelling entities. This last role can be played by advanced students according to the specifications of pedagogical scenarios of use. Additional actor is the administrator of the community and of the common repository.

There are five main components in the MS distributed environment, which reside in three types of nodes, the student node, the teacher node and the server node, as shown in figure 2. The main components are: The *Model Editor*, the *Entity Editor*, the *Analysis & supervision environment* (see Avouris et al., 2003b), the *Common Repository* and the *Community support* environment.

These are briefly presented in the following. There are going to be two different installations of the MS software, the client that can be used either by teachers (teacher client node in figure 2) or students (student client node) with different capabilities and the server that is administered by the administrator and used remotely also by the other actors through their client components. Since the most typical use of MS is in a school laboratory, and in this case the same workstation could be used by many students of different classes, the client supports multi-user access, identification and authentication of the user and user private space. The MS environment is presented in the following as client and server side tools.

CLIENT SIDE TOOLS

Model Editor

The main tool is the ModelEditor (ME), which is accessible by both the teacher and the student. This is a direct manipulation space, which is expected to be used mostly by students for building models out of primitive modelling entities. ME supports building of different kinds of models mostly for students of 11-16 years. The ME needs to support building of dynamic models, i.e. models that simulate a behaviour to the user. These can be either semi-quantitative models, i.e. models in which the entities are related by semi-quantitative relations or quantitative models, where the relations can be mathematical expressions. Also static qualitative models (concept maps), can be built using this environment. Emphasis has been given so far on semi-quantitative modelling and reasoning, as this has been the main innovation of the ModelsCreator environment, (see Komis et al., 2001).

The ME puts great emphasis on visualisation of the modelling entities, their properties and their relations, supporting the reasoning development of young students (NCTM 2000). This feature is extended also to the simulation of executable models allowing their validation through representation of the phenomenon itself in a visual way.

The activity space of the ME modelling environment needs to be shared by multiple actors, permitting collaborative modelling activities of learning actors at a distance. The size of the groups engaged in synchronous collaboration is expected to be small, so point-to-point connection is feasible. The messages exchanged are of small size, as due to replication the only information exchanged relates to control of modelling activities (e.g. add entity E_x to the (x,y)), while the entity E_x itself is not usually transferred between the distant nodes, as discussed in more detail below. Alternative views of a model are supported. A model can be seen as a network of entities and relations, which is the normal view as build in the activity space, or as a table of values, a graph or a bar chart, presenting specific relations and properties of the model in new windows.

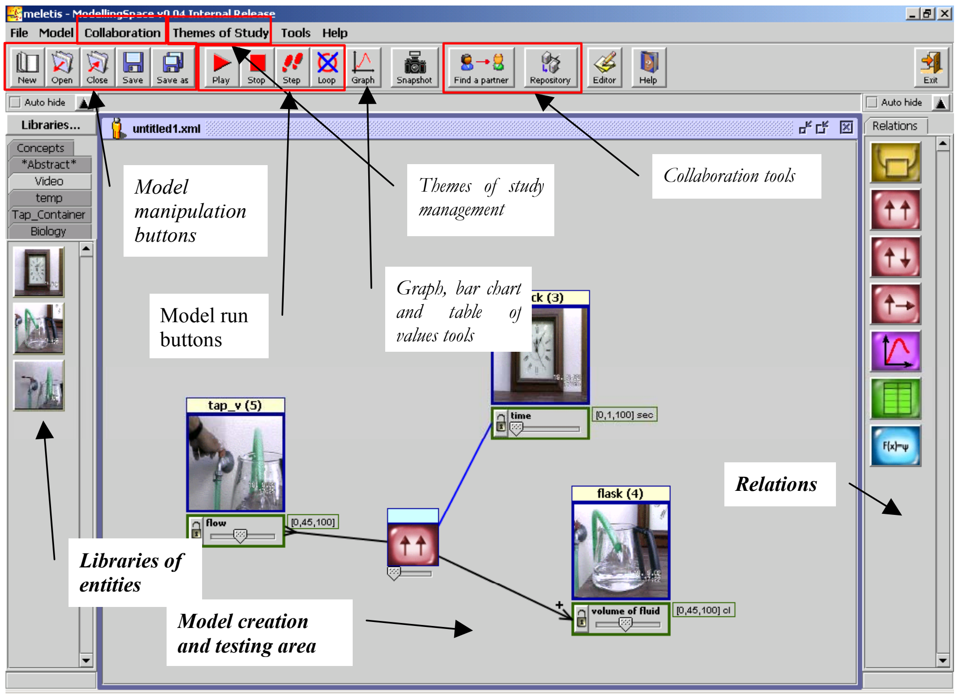


Figure 3: The model editor (ME) environment

The ME is designed to be a user-sensitive environment, providing different functionality to different actors. So the teachers can use the tool for supervising simultaneously many groups of students, and share many collaboration windows, while special permissions are allocated to them in relation to coordination of collaboration, access to libraries of entities and management of student accounts, as discussed in more detail in the following.

Entities Editor

A second tool of the client node is the Entities editor (EE). This tool is used typically by the teacher or advanced student in order to create primitive entities, which can be stored in the local *Entities Libraries* or send to the server *Common Repository*. The entities are the building blocks of the models. Each entity is defined as an object, representing an object or a concept of the real world that has a name, a text description and a graphical representation. A number of properties can be associated to an entity through this tool. For instance the Entity *Plant* can have the properties *Growth*, *Energy*, *Food_intake* in the context of a photosynthesis model.
There are entities that can have more abstract meaning (variables) which have no properties associated. The properties in general have a range of values that they can take; while for each property the min, max and default value is defined. The entity is associated to a number of states. Each state corresponds to a distinct range of values of the entity's properties. An iconic representation of the entity is associate to each one of these states, see figure 4.

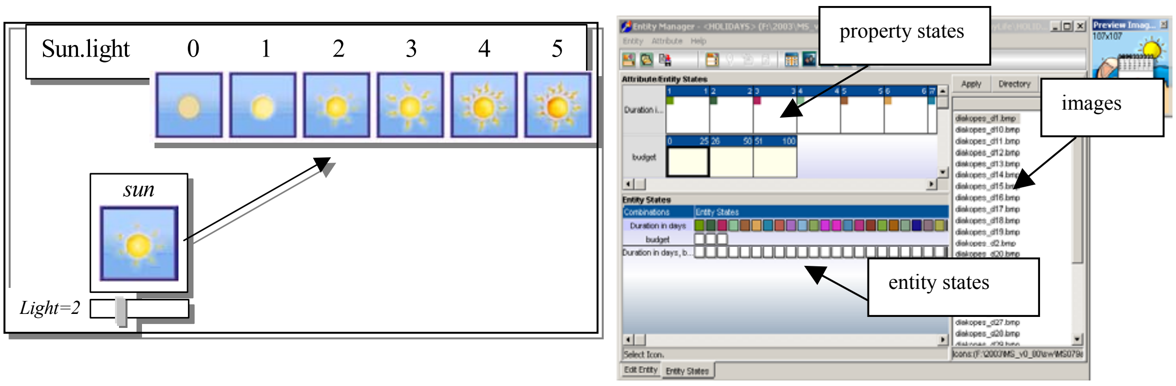


Figure 4: An example of an entity definition, the property light of the entity sun is associated to 6 states and corresponding iconic representations. On the right the state image assignment tool is shown. Two properties have been defined; their states and images have been associated to the produced entity states.

Various image formats can be used as entity representations. A generated entity by the tool is represented by a data structure defining the entity properties, states, etc. and a number of associated image files. An XML representation of the entity can be produced, along with binary compressed representations for storing locally. The user can define as many entity properties and states as he/she wishes, however special attention should be paid on the size of the final entity, which in case of complex entities can be quite large, depending on the image format and number of distinct associated images.
MS is an open environment. The importance of this open character on collaborative modelling and the implications on the architecture should be briefly discussed. In a typical closed collaborative problem-solving environment, the students have at their disposal a common set of basic constitutive abstract primary entities, out of which they construct their representations. These primitives can be rectangles, ellipses, squares, different statement types, etc., as it is the

case in Belvedere (Suthers and Jones 1997), COLER (Constantino and Suthers, 2001), C-CHENE (Baker and Lund, 1997), Modeller Tool (Koch et al., 2001), etc. So common understanding is based on the existence of these common basic primitives. On the contrary, in an open system like MS, one user before entering in a specific collaborative session may possess a different set of primitive elements to this of her peer. As a result diverse sets of primitive objects can be found in the client local libraries and the server repositories. These objects are represented through XML a structured data interchange protocol approach, which permits association of semantic meaning and syntactic validation. In this a GUID is used representing the unique identity of an entity, which is generated by an algorithm as a combination of creation time, unique workstation and user identity at entity creation time.

Communication protocol

Synchronization of collaborating partners is achieved using a peer-to-peer protocol, without intervention of a server. The mechanism is based on a set of reactive agents, which try to achieve synchronization with the corresponding agents of the peer host based on a stimulus–response model. So in a joint problem solving activity each object and each relation introduced, act as reactive agents. The behaviour of each agent depends on whether it is on the active user's side or on the passive user's side. If it is on the active user's side it monitors user events that are related to the particular object (movement, changing of properties, deleting etc.), and sends these events to the equivalent agent on the passive user's side. This is achieved through the Mediators, shown in figure 5. When the Mediator of the passive user's side receives the message, it decodes it and informs the equivalent agent who acts accordingly.

This necessitates that the objects present in the Activity Spaces of two collaborating partners are identical. However, as discussed earlier, there is a possibility that two users are in possession of different primitive library objects, due to the open architecture of the environment. So there can be a case when the active user A adds an object into the shared activity space, which does not exist in the library of user B. In this case it is necessary to update the library of user B at run time with the missing object before proceeding any further. This is done transparently from the users as follows: When user A inserts the new object O_i in the Activity Space, Mediator A informs Mediator B about the addition of the new object, sending the appropriate message with the object's GUID. Mediator B searches the local Entity Library for O_i If this object does not exist on host B then Mediator B asks A to send a copy of object O_i before proceeding any further. Mediator A sends the object, and waits. During this activity the user actions in the shared Activity Space are suspended and a message is displayed that the peer library is updated. After the sending is complete Mediator B informs Mediator A that it has received the object and the activity can proceed. The object icons can be sent either directly as shown in figure 5 or through the server if the size of the multimedia files are too large and can disrupt activity for both partners for too long. In the latter case the message is sent to the server with the GUID of the object, and the partners download the object from the corresponding repository in the server (as it is described in the following section the common repository is organised in many different ones, and not all users have access to all repositories). A process has been designed to look for the entity in the repositories to which the user has access. In case that the material is not found in the public repository, but in a restricted one to which the first user has access but not the rest, a copy of the entity is made in the user's exchange tray and it is from there, where the other users are allowed to pick it up). If the object does not exist in the server, it is uploaded, transparent to the two users from the library of user A.

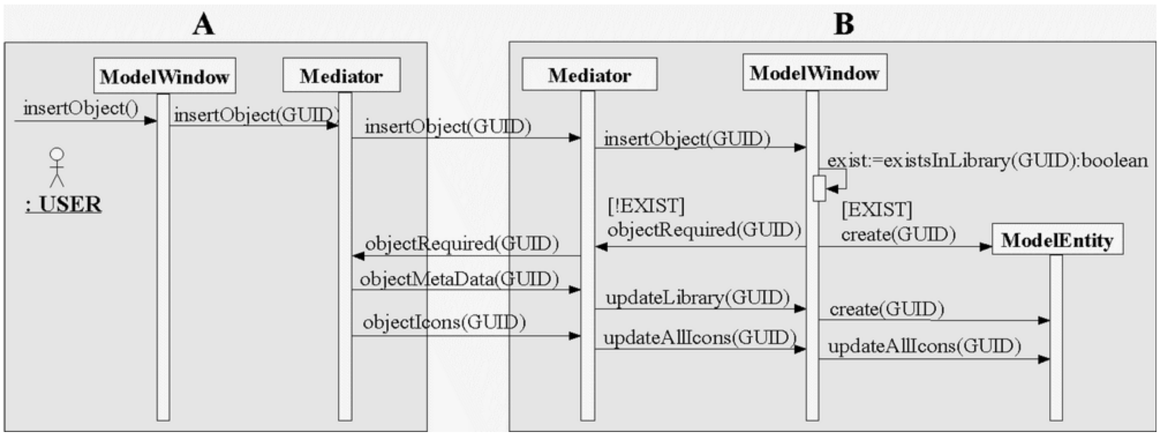


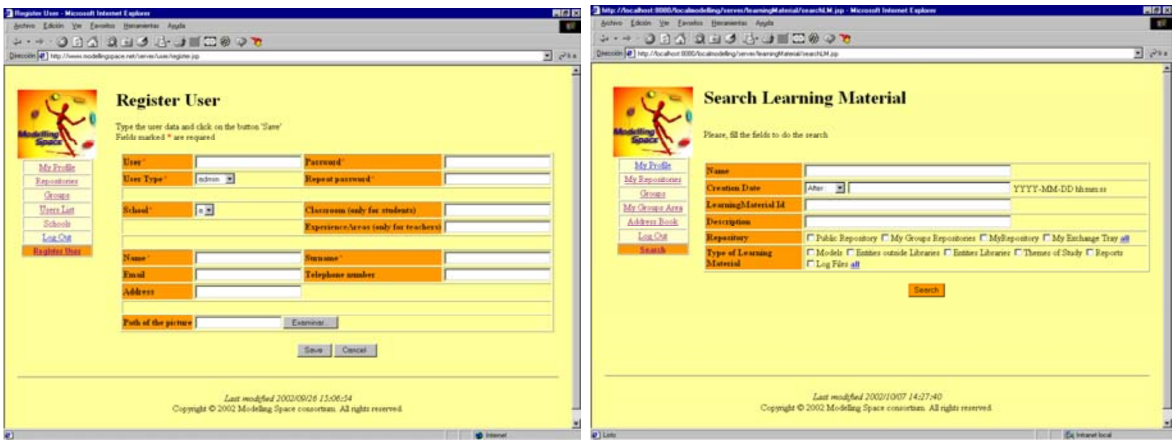
Figure 5: The communication protocol interaction diagram

SERVER SIDE TOOLS

As discussed in the previous sections, the MS architecture is based on a thick client component, which contains a number of interoperable tools. Even synchronous collaboration is effected through peer-to-peer interaction. However the proposed architecture contains also a server node which offers the following services: (a) management of the repositories; (b) management of users and schools; (c) management of collaboration groups; and (d) support of peer-to-peer collaboration. Many issues related to security and asynchronous interaction can be solved through this server, as proposed by many collaboration support systems, e.g. see Heibinger (2001) and Constantini et al. (2001).

(a) *Management of the repositories.* The management of the repositories is deeply linked with the management of the users and the management of groups. The different kinds of repositories that exist in the server are the following: the *public repository*; the *personal repositories*; the *exchange trays*; and the *group repositories*.
The *Public Repository* is the main repository of the ModellingSpace server. Material stored there is available for all users, but only teachers have permission to upload entities, models, themes of study, since only correct models, and useful material should be stored in this repository. Therefore when a student wants to upload material to this repository, the material needs to be validated by a teacher.
Each user has a *Personal Repository,* which no other users can access, and an *Exchange Tray*, accessible also to other users, which is used as a secure way to exchange documents. These two kinds of repositories are automatically created in the server when the administrator enrols a new user, and they disappear when the user is deleted from the system.
To the groups repositories only members of the group have access.
Thus with the term *common repository*, we mean a set of repositories that exist in the server.
(b) *Management of users and schools*: Only the administrator can add new schools or new users to the server, and when a new user is added, two new repositories are automatically created: a personal repository and an exchange tray.
(c) *Management of collaboration groups*: If the concept of a group is understood as a set of users who are collaborating in the construction of a new model, two kind of groups can be distinguished: when users are collaborating on-line and off-line. Collaboration means the shearing of knowledge, work and material, so groups need special repositories to which only their members can access. Therefore at the same time that a group is created a group repository is also created, and the management of these two kinds of groups is not done in the same way.

Permanent groups need to be created by an administrator indicating whether the group is moderated or not; restricted or not (that is if there is a maximum number of members allowed); etc., whereas *collaboration groups* are automatically created when two users start on-line collaboration.
In both cases the life of the group repository depends on the life of the group: it appears when the group is created and once the group is deleted (in the case of the permanent ones) or the on-line collaboration ends (in the case of the collaboration groups), the group repository is also deleted from the server.


Register User

Type the user data and click on the button 'Save'
Fields marked * are required

User*		Password*	
User Type*	admin	Repeat password*	
School*	a	Classroom (only for students)	
		ExperienceAreas (only for teachers)	
Name*		Surname*	
Email		Telephone number	
Address			
Path of the picture	Examinar...		

Last modified 2002/09/26 15:06:54
Copyright © 2002 Modelling Space consortium. All rights reserved.

Search Learning Material

Please, fill the fields to do the search

Name	
Creation Date	After YYYY-MM-DD hh:mm:ss
LearningMaterial Id	
Description	
Repository	☐ Public Repository ☐ My Groups Repositories ☐ MyRepository ☐ My Exchange Tray all
Type of Learning Material	☐ Models ☐ Entities outside Libraries ☐ Entities Libraries ☐ Themes of Study ☐ Reports ☐ Log Files all

Last modified 2002/10/07 14:27:40
Copyright © 2002 Modelling Space consortium. All rights reserved.

Register User - Microsoft Internet Explorer
http://www.modellingspace.net/server/user/register.jsp
My Profile
Repositories
Groups
Users List
Schools
Log Out
Register User
Save
Cancel
http://localhost:8080/localmodelling/server/learningMaterial/searchLM.jsp - Microsoft Internet Explorer
My Profile
My Repositories
Groups
My Groups Area
Address Book
Log Out
Search
Search

Figure 6: The user registration and the search learning material server interfaces

(c) *Support of peer-to-peer collaboration.* The role of the server in the peer-to-peer collaboration has already been described in the Communication Protocol section. Additional functionality of the server involves tracking of physical address of users, who might not have a permanent IP address, and information on presence support, i.e. inform users on availability of their peers for synchronous interaction. Finally, these Community Support Tools provide also other services like session management, login of users, etc.
An interface to the server repository has been built through which one can download material in the *Common Repository* (CR) or any of the other private repositories to which the user has access to, as shown in figure 6.

CONCLUSIONS

The main functionalities of the ModellingSpace architecture are:
(a) MS is an environment in which models of various kinds can be built and explored, made out of primitive entities, making it an environment particularly suitable for science education.
(b) The users, students or teachers, are able to create, store in and retrieve from local or common repositories primitive entities and models
(c) Services are provided for supporting creation and maintenance of the activities of virtual communities of students of different schools who use ModellingSpace through the server.
(d) The teachers who use MS are able to supervise single students or groups of students engaged in modelling activities in the same place (school lab) or from a distance
(e) Asynchronous collaboration of students engaged in modelling activities are supported through community tools
(f) Synchronous collaboration of small groups of students, engaged in modelling activities, are also supported, through a shared activity space and a text communication tool.

The above functionalities are now tested through a number of field studies, e.g. Margaritis et al. (2003), Avouris et al. (2003b), through which the effectiveness of the presented architecture is evaluated.

ACKNOWLEDGEMENT
The reported work has been performed in the frame of the IST-School of Tomorrow Project IST-2000-25385 "ModellingSpace". In this project participate the University of the Aegean, (GR), the University of Patras (GR), the University of Mons-Hainaut (B), the New University of Lisbon (PT), the University of Angers (F) and SchlumbergerSema (SP).

REFERENCES

Avouris N.M., Dimitracopoulou A., Komis V., (2003a), On analysis of collaborative problem solving: An object-oriented approach, J. of Human Behavior Vol. 19, Issue 2, March 2003, pp. 147-167.

N. Avouris, V. Komis, G. Fiotakis, M. Margaritis, N. Tselios (2003b), Tools for Interaction and Collaboration Analysis of learning activities, Proc. 6th CBLIS 2003, Nicosia.

Baker M.J., de Vries E., Lund K. & Quignard M (2001) Computer Epistemic Interactions for co-constructing scientific notions: Lessons Learned from a five-years research program, Proc. 1st EuroCSCL 2001, pp.89-96.

Baker, M.J. & Lund K. (1997) Promoting reflective interactions in a computer –supported collaborative learning environment. Journal in Computer Assisted Learning, 13, 175-193.

Chabert A., Grossman E., Jackson L., Pietrowicz S. Seguin C., (1998), Java Object-Sharing in Habanero, Com. ACM, 41 (6), pp. 69-76.

Constantini F., Toinard C., (2001), Collaborative Learning with the Distributed Building Site Metaphore, IEEE Multimedia, July-Sept. 2001, pp. 21-29.

Constantino-Conzalez & Suthers D. (2001), Coaching Collaboration by Comparing Solutions and Tracking Participation. 1st EuroCSCL 2001, pp.173-180.

Dix A., Finlay J., Abowd G, Beale R., (1998), Human-Computer Interaction, 2nd Edition, Prentice Hall.

Fidas C., Komis V., Avouris N.M. (2001). Design of collaboration-support tools for group problem solving, Proceedings PC HCI 2001, pp. 263-268, Typorama Pub., December 2001, Patras, Greece.

Fidas C., Komis V., Avouris N.M., Dimitracopoulou A., (2002a), Collaborative Problem solving using an Open Modelling Environment, Proc. CSCL 2002, pp. 654-656, Erlbaum Assooc, Hillsdale NJ, 2002.

Fidas C., Komis V., Tzanavaris S., Avouris N., (2002b), Heterogeneity of learning material in synchronous computer-supported collaborative modelling, Computers and Education (submitted)

Koch J.H., Schlichter J. & Trondle P (2001). Munics: Modeling the flow of Information in Organisation. 1st EuroCSCL 2001, pp.348-355.

Komis V., Avouris N., Fidas C., (2002), Computer-supported collaborative concept mapping: Study of synchronous peer interaction, Education and Information Technologies vol.7, 2, pp.169-188.

Komis V., Dimitracopoulou A., Politis P., Avouris N. (2001). Expérimentations exploratoires sur l'utilisation d'un environnement informatique de modélisation par petits groupes d'élèves, Sciences et Techniques Educatives, Vol. 8, no 1-2, pp.75-86.

Koutlis E. (1998) E slate specification, see www.eslate.cti.gr

Margaritis M., Avouris N., Komis V., (2003) The architecture and evaluation of a collaborative learning environment, 6th CBLIS, Nicosia 2003.

National Council of Teachers of Mathematics. (2000). Principles and standards for school Mathematics. Reston,VA: NCTM.

Preece J, Rogers Y, Sharp H., (2002), Interaction Design beyond human-computer interaction, Willey and Sons.

Suthers D. & Jones D. (1997), An Architecture for Intelligent Collaborative Educational Systems. In B. du Boulay, R. Mizoguchi (Eds) 8th World Conference on Artificial Intelligence in Education (AIED'97), pp. 55-62.

Suthers, D.D., (2001), Architectures for Computer Supported Collaborative Learning, Proc. IEEE int. Conf. On Advanced Learning Technologies, ICALT 2001, Madison, Wisconsin

Tewissen, F., Baloian N., Hopper U., Reimberg E. (2000), Match Maker synchronizing objects in Replicated Software Architectures, Proc., 6th CRIWG, Madeira.

Professor Nikolaos Avouris,
ECE Dept. Human-Computer Interaction Group,
University of Patras,
GR-26500 Rio - Patras,
Greece (www.ee.upatras.gr/hci) Email: N.Avouris@ee.upatras.gr

Meletis Margaritis,
ECE Dept. Human-Computer Interaction Group,
University of Patras,
GR-26500 Rio - Patras,
Greece Email: Margaritis@ee.upatras.gr

Asst. Professor Vassilis Komis,
Early Childhood Education Dept.
University of Patras,
GR-26500 Rio - Patras,
Greece Email: komis@upatras.gr

Angel Saez,
SchlumbergerSema,
Albarracin 25,
E-28037 Madrid,
Spain Email: angel.saez@madrid.sema.slb.com

Dr. Ruth Melendez,
SchlumbergerSema,
Albarracin 25,
E-28037 Madrid,
Spain Email: ruth.Melendez@madrid.sema.slb.com