Collaborative distributed environments for learning design tasks by means of modelling and simulation

Crescencio Bravo\textsuperscript{a,}\textsuperscript{*}, Miguel Á. Redondo\textsuperscript{a},
Manuel Ortega\textsuperscript{a}, M. Felisa Verdejo\textsuperscript{b}

\textsuperscript{a}Departamento de Informática, Universidad de Castilla—La Mancha,
Paseo de la Universidad, 4. ESI, 13071 Ciudad Real, Spain

\textsuperscript{b}Departamento de Lenguajes y Sistemas Informáticos, Universidad Nacional de Educación a Distancia,
Ciudad Universitaria, s/n. 28040 Madrid, Spain

Received 28 July 2004; received in revised form 19 January 2005; accepted 28 January 2005

Abstract

The Simulation discipline has to face new challenges such as the incorporation of Collaborative Technologies for professional use as well as for teaching purposes. This integration permits the creation of new kinds of support for collaborative learning processes. In this paper, we explore the potential of this synergy with DomoSim-TPC, a synchronous distributed collaborative environment for the teaching and learning of Domotics. The system supports an active, simulation-based and problem-based approach for learning house automation design. Using this learning environment, teachers propose and organize problem solving activities and the students carry out, in a collaborative way, the construction of artefacts (designs) using modelling and simulation tools.

© 2005 Published by Elsevier Ltd.

Keywords: CSCL; Modelling and simulation; Networked learning environments; Distance education

* Corresponding author. Tel.: +34 926 295300x3717; fax: +34 926 295354.
E-mail addresses: crescencio.bravo@uclm.es (C. Bravo), miguel.redondo@uclm.es (M.Á. Redondo),
manuel.ortega@uclm.es (M. Ortega), felisa@lsi.uned.es (M.F. Verdejo).

1084-8045/$ - see front matter © 2005 Published by Elsevier Ltd.
doi:10.1016/j.jnca.2005.01.003
1. Introduction

The Simulation discipline, now fully matured, is currently being successfully applied in all social and human sciences and in engineering, including the teaching–learning processes. However, these tools have to face new challenges, incorporating new human–computer interaction paradigms such as Virtual Reality and Collaborative Work/Learning (CSCW\textsuperscript{1}/CSCL\textsuperscript{2}), and they need to reconsider their traditional techniques in order to adapt them to the new Web technologies. With respect to Web-based Simulation, Page (1998) points to Distributed Modelling as one of its areas. It researches the development of tools and environments supporting the collaborative and distributed design of simulation models through Internet. This interest in group work is also being shown by businesses that will increasingly use their geographically disperse personnel through multiple application domains using modelling and simulation and other resources to solve complex problems (McQuay, 2004).

According to Hansen (1990), students retain 25% of what they listen to, 45% of what they listen to and see, and 70% when they manipulate, control and modify experiments, putting into practice what they are learning. These experiments can be carried out by means of simulation. From the interaction between Collaborative Technologies and Simulation new kinds of support to learning can be created.

The integration between Simulation and CSCL has been explored in several systems. In most of them although communication tools, such as chats or discussion lists, and other collaborative tools, such as whiteboards or shared views, are included, they are not linked or are very loosely linked with the modelling and simulation task. Moreover, the challenges of synchronous distributed collaboration, made possible thanks to network technology, should be confronted. These challenges and limitations have been approached with the DomoSim-TPC project (Bravo, 2002; Bravo et al., 2002; Redondo, 2002). The objective of this project has been to create a powerful and attractive environment to improve the process of teaching Domotics. This environment proposes active, problem-based learning, providing modelling and simulation tools. The students, organized in groups, solve problems (selected from a collection) using modelling and simulation techniques. The environment itself allows the teachers to manage the problem collection and to define activities. Furthermore, tools to monitor and analyse the students learning processes are available.

The Domotics domain is taught as a subject matter in Secondary Education centres and in Engineering university colleges. Domotics means home automation or intelligent building design. The objective for students is to learn how to design home automation solutions that include the management of thermal and luminosity comfort, energy control and security control (accidents and intrusion).

The remainder of this paper is organized as follows. In Section 2 we review some simulation environments that include collaborative support, in order to characterize the state of the art and to propose relevant features to describe this kind of systems.

\textsuperscript{1} Computer-Supported Cooperative Work.
\textsuperscript{2} Computer-Supported Collaborative Learning.
The DomoSim-TPC environment is described in Section 3: firstly, the background and learning approach is discussed; secondly, we focus on the simulation task detailing the functionality of this component; and lastly, the global system architecture is presented. The underlying general pedagogical approach is further elaborated in Section 4, including the organization of the learning process and the support provided by the system. Section 5 includes some comments on evaluation aspects of the environment as well as a comparison, in terms of functionality, to other collaborative simulation systems. Finally, in Section 6, a summary and conclusions are drawn and future work is outlined.

2. Collaborative simulation and related systems

Computer-Supported Collaborative Learning (CSCL; Koschmann, 1996) is a paradigm that studies the use of computer technology to support collaborative approaches for learning. The combination of collaboration with modelling and simulation tasks allows the definition of learning scenarios where peers learn together to build models and to check their validity. We will refer to this combination as Collaborative Modelling and Simulation for Learning (CMSL). The works of some authors justify this approach. For example, van Joolingen (2000) uses the term Collaborative Discovery Learning to refer to the new educational applications that can be offered from the interaction that takes place between the collaboration support and the specific learning processes relative to discovery in simulation-based environments, whilst Salomon and Globerson (1989) state that collaboration can have a positive influence on explanation and on the discovery learning process.

In order to propose a classification of CMSL systems in which to frame the environment we have developed, we have studied the following simulation-based learning environments that incorporate collaborative support: ERCIS (Berglund and Eriksson, 1998), SESAM (Software Engineering by Simulation of Animated Models; Schneider and Nakakoji, 1995), SIMPLE (Simulated Processes in a Learning Environment; Plaisant et al., 1999), WebNet (Stahl, 2000), LESP\(^3\) (Learning Environment for Simulation of Particulate models of matter), TurboTurtle (Cockburn and Greenberg, 1998), SIMPLEZ (Llamas et al., 2001) and CROS (Giannetti, 1998). These systems can be studied from two criteria: the type of modelling they allow and the collaboration strategies they incorporate (see Table 1).

With regard to the type of modelling, we identify two classes of systems. In the first one, the model is embedded in the system and cannot be substantially altered nor created by the students. In this case, the only possible user intervention consists of changing the value of some parameters to influence the simulation. In the second class, the model to simulate is built by the students. This building process can be carried out without a previously established goal or in order to reach a particular aim. In the first case the students test and experiment freely on the models they build, without aiming for a specific target. In the second case, a learning goal is usually expressed in terms of a problem formulation, which can include data, constraints and requirements for a solution.

---

With respect to the collaboration strategies related to the simulation, we distinguish between individual simulation systems and synchronous collaborative simulation systems. Most of the systems provide a simulation in which only one user participates. For that reason the simulation is called individual. However, traces of this simulation can be stored and made available for exchange by the different users (teachers and students). Providing the learners with the recording of their actions (traces) can help them to monitor their behaviour, to reflect on their progress and to experiment with revisions of their experiences. Furthermore, the traces can be exchanged using asynchronous tools such as electronic mail. These traces are an opportunity for collaboration: students can revise the work of their peer students and build contributions on those made by others. Other asynchronous tools, such as group glossaries or discussion forums, can also be used. Some individual simulation systems also include synchronous tools, for instance for communication or revision of the models and of the simulation process.

In the case of collaborative simulation systems, the simulation is typically carried out in a synchronous way, also known as of real time. This simulation can be distributed, when each participant executes a simulator that is part of a global simulation, or not distributed, when the whole simulation is shared by all the participants, who have the same view of the model and can interact simultaneously, with it.

The shared simulation systems, such as DomoSim-TPC, are substantially, different from the Shared Window Systems, such as Netmeeting. In the former the users share the simulation, but each one executes an application instance that can be used and

---

Table 1: Criteria to classify the CMSL systems

<table>
<thead>
<tr>
<th>CMSL System</th>
<th>Modelling Type</th>
<th>Collaborative Strategies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The model is fixed</td>
<td>The model is built by the students</td>
</tr>
<tr>
<td>ERCIS</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>SESAM</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>SIMPLE</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>WebNet</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>LESP</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>TurboTurtle</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>SIMPLEZ</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>CROS</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>DomoSim-TPC</td>
<td>×</td>
<td></td>
</tr>
</tbody>
</table>

configured according to his/her preferences. In the latter a single-user application is replicated in the users’ computers; that is to say, there is only one application instance.

Table 1 organizes the mentioned systems, taking into account the above relevant features for learning purposes. Analysing the type of modelling we can see that most of the systems do not allow modelling. ERCIS, SESAM, SIMPLE, WebNet, TurboTurtle and CROS simulate a fixed model incorporated in the system that can only be parameterised. LESP and SIMPLEZ do allow modelling. In the former the students build particle models by means of direct manipulation that they will be able to simulate later on. In the latter the students edit the assembler code of computer programs whose execution they will be able to simulate. In DomoSim-TPC the built models are the solution to a design problem.

In relation to the collaborative strategies, the majority of systems provide an individual simulation. SIMPLE, WebNet and LESP permit these simulations to be exchanged among students and teachers. SESAM allows teachers and students to share arguments and experiences by means of the discussion on the registration of the actions taken during the simulation. In this case the collaboration is synchronous and asynchronous. CROS and SIMPLEZ allow synchronous interaction by means of a chat, but in a separate workspace from that of simulation. ERCIS is a system that is utilized for military training and uses distributed simulators to recreate large scale simulation. The most similar system to DomoSim-TPC is TurboTurtle: both offer synchronous simulation so that all the participants can share it and interact on the model in real time.

3. DomoSim-TPC: a CMSL environment for learning Domotics

The DomoSim-TPC environment is used for collaborative learning of domotical design. It is the successor of DomoSim-TP (Bravo et al., 2000a), a system for distance learning of Domotics by means of planning and design. The ‘C’ in DomoSim-TPC stands for collaboration. This new environment has been designed and developed between 1999 and 2002 by the CHICO5 Group from the University of Castilla—La Mancha. The situation in which it is applied, the teaching method, its architecture, how the problem solving process has been structured, and the kind of tasks it allows to be carried out are described in the following paragraphs.

3.1. Background

The realization of practical works is a key issue in the learning of Domotics. However, the material needed to carry out these practices is usually expensive and in many cases is not available. These problems make it difficult to bring the students close to real settings, to reproduce accidents and to outline chaotic situations like those that can occur in the real world and at which their design should be aimed. With the use of DomoSim-TPC to study this subject, especially by taking advantage of its simulation features, these limitations can be overcome and the learning experience can be improved.

---

5 http://chico.inf-cr.uclm.es.
Design activities involve the creation of artefacts fulfilling a set of requirements. Our approach for teaching domotical design is to provide learner-centred scenarios, promoting learning by doing (Schank et al., 1999), in a problem-based approach. This perspective develops the aforementioned ‘put into practice’ Hansen’s idea, which is frequently carried out with simulations. In modern theories of learning, the view of the learner as an active agent is becoming increasingly important (van Joolingen, 2000). A basic principle of Constructivism is to see learning as an active process, in which the student elaborates interpretations based on his/her practice to build a meaningful personal knowledge. The Learner-Centred approach postulates that the control is on the learner side, while the role of the teacher is to be seen as mentor or coach. Another important aspect is the kind of learning activities proposed to students. Current problem-based approaches claim that authentic problems (Norman and Spohrer, 1996) provide the context for meaningful learning. Furthermore, with computer-based modelling and simulation tools, we are providing the students with the same kind of tools that experts use to solve real problems (Guzdial et al., 1996).

The DomoSim-TPC environment is based on the instructional method of Problem-Based Learning (PBL; Barrows and Tamblyn, 1980). In PBL the students work actively in groups on problems extracted from real practice. According to Guzdial et al. (1996), collaboration in problem solving activities provides not only an appropriate performance but also the groups of students can solve more interesting and complex problems than individual students. When students work in groups, they need to propose and discuss designs, make arguments and give opinions to other group members, encouraging the kind of reflection that leads to learning. However, collaboration rarely emerges spontaneously. The role of the teacher is to design the learning activities in such a way as to create occasions for collaborative behaviour. There are a number of collaborative strategies proposed in the literature.

Our collaboration procedure for modelling and simulation is based on the structuring approach of Wessner et al. (1999). According to these authors, learning can be improved by defining certain constraints and by structuring the learning process using Learning Protocols. These protocols can be useful at the different collaboration levels, for instance to support group activities requiring some coordination control or common navigation. In line with these ideas, the procedure we propose—fully described in Section 4.1 and shown in Fig. 4—involves two stages that correspond with the main domain tasks: to create a design model and to validate the model using simulation processes.

The modelling activity includes three tasks (Bravo et al., in press):

- **Design.** To build the model that solves the problem, using a design tool allowing objects to be inserted in a shared whiteboard, relating them and giving value to their properties.
- **Work distribution.** Students have to decide by themselves how to divide up and share the work. The system allows them to assign either specific design actions or sub-problems to each group member.
- **Parameterisation.** To give a value to the general variables of the problem. In order to do so, the students follow different strategies to decide what value should be assigned to each variable.
The validation/experimentation activity consists of two tasks (Bravo et al., 2002):

- **Cases and Hypotheses.** To decide what simulation cases should be used for experimentation and the set of hypotheses to validate, in order to justify that the model is a solution fulfilling the requirements.
- **Simulation.** To run the simulation cases with the designed model.

In the following paragraphs we focus on this experimentation activity.

### 3.2. Learning approach for the experimentation phase

A design problem is proposed to students. Solving a problem means to build a model according to some specifications. To support this process, collaborative workspaces are provided, which allow different solutions to be created and then checked. The experimentation stage is articulated in such a way as to stimulate ‘what would happen if...?’ reasoning in order to investigate whether or not the model has the behaviour expected, covering the range of potential different situations. The system support for this task is presented below.

#### 3.2.1. Cases and hypotheses workspace

There is a Cases and Hypotheses workspace where the students develop skills for hypotheses generation and testing. In Discovery Learning theories (van Joolingen and de Jong, 1997) hypotheses are described as statements about the relations between variables and experiments and about the variable changes together with the results or effects of these changes. Despite the advantages of reasoning with hypotheses, it has been reported that students often find it difficult to establish them (de Jong and van Joolingen, 1998): they do not know what a hypothesis is or the form it should take, and they find it difficult to state or to adapt a hypothesis on the basis of the acquired data. To overcome in part these difficulties, problems in the DomoSim-TPC library incorporate example hypotheses.

Parallel to the hypotheses management, in the workspace supporting this task new simulation cases can be proposed and the one to simulate can be selected. A case is a particular simulation execution. Each problem includes a small list of cases to simulate, although the students can define new cases. Thus, in this workspace the students can carry out any of these four subtasks: propose a new simulation case, select a case to simulate, propose a new simulation hypothesis and annotate whether a hypothesis has been validated or not. In Fig. 1 the Cases and Hypotheses panels of the Cases and Hypotheses workspace of DomoSim-TPC are shown. Once a case has been selected, and the simulation button pressed, then the system switches to the Simulation workspace. When the simulation finishes users return to this Cases and Hypotheses workspace.

In Fig. 1, two simulation cases related to a problem can be seen. The problem formulation requires students to run simulations for two different situations: with daylight and by night. In the figure, the student *hperia* has proposed to the group the simulation of the second case (1). In DomoSim-TPC cases and hypotheses are handled independently. The cases help to test whether the model covers different behaviours, but simulation results should be studied and compared with the hypotheses, in order to check whether...
the hypotheses are confirmed or invalidated. As stated before, a problem incorporates examples of hypotheses, which act as clues for the students. However, the most interesting aspect is that students can add new hypotheses related to significant events or situations in the model and they can check their validity in a new experimentation cycle. In the figure three hypotheses can be seen: two incorporated in the problem and one proposed by the students. There are two unresolved hypotheses and another one that has been marked by the students as being true (it is validated) (2). The student dmartin has proposed the true value ($V$) for that hypothesis and the student hperea has agreed on that (3). This hypothesis states the idea that ‘it is possible to reach a temperature of 20 °C’ as the problem requires. The students have demonstrated this hypothesis to be true in the simulation. The false value ($F$) represents a hypothesis that is invalidated.

3.2.2. Simulation workspace

Simulation is a mechanism that allows us to validate designs and discover the laws that govern the behaviour of the models. In order to carry out this task in collaboration, besides the traditional components of a simulator, other additional components are required. Thus, we distinguish between domain-dependent components and generic tools to support collaboration. The former are the direct manipulation mechanisms that allow interaction on the models and the elements that show simulation information. In order to simulate domotical models, we have represented this domain computationally by means of a set of objects. These domain objects, called operators, have properties, called parameters, and are related to each other by means of binary associations. Generic tools consist of the communication and coordination support and of awareness functions. The collaborative simulator in Domosim-TPC includes the following elements (a snapshot of the interface is shown in Fig. 2):

- Direct manipulation mechanisms for interaction on the models:
  - Collaborative electronic whiteboard (a): It is the work surface that contains the domotical model and the house plan.
  - Simulation actions (b): The set of buttons that permit actions to be carried out on the model objects, such as to switch on/off, to open/close, etc.

---

6 Awareness and knowledge of the rest of the group members and of their work.
Simulation information elements:
- Environmental information (c): It contains information in relation to temperature, illumination and electrical consumption, as well as the simulation clock.
- Panel of Properties (d): It allows the values of operator properties to be shown.
- Configuration options: To change the simulation step and other parameters.

Communication and coordination support:
- Structured Chat (e): It is used for communication in real time. It offers a set of predefined communication acts suitable for simulation, such as ‘Why…?’ , ‘There is a mistake in…’, ‘I think that…’, etc.
- Decision-Making Tool (f): This button opens a tool supporting decision-making in order for students to decide whether or not to carry out certain actions.
- Panel of end of simulation (g): It allows the students to decide to finish the simulation.

Awareness functions:
- Session Panel (h): It shows the list of participants and the name of the user who currently takes the floor to act on the model.
- List of Interactions (i): It is a trace of the events generated by simulation actions as well as actions from the users, represented in a textual way.
- Tele-pointers (j): They are the users’ remote pointers.

Fig. 2 shows a simulation example of a domotical model made up of four subsystems: luminosity (SI), temperature (ST), accidents (SA) and intrusion (SS). In this problem
the house plan is of only one room (lounge). An activated light bulb (1), a closed blind (2), a person in the room (3), etc. can be seen. In the panel corresponding to the environment (c), the temperature (19.65 °C) and luminosity (26.4 lux) are displayed, as well as the current consumptions of the two load lines (80 and 220 W) (4). The simulation step is fixed at 1200 s and the clock shows the current time (5). The List of Interactions (i) contains messages giving information about the automatic behaviour of the regulator systems. The Panel of Properties (d) shows the state of a radiator. In general terms, the problem outlines two requirements: to keep the temperature between 17 and 23 °C and the illumination between 100 and 200 lux; and has a constraint: to limit the electrical consumption to 1500 W. The users cbravo and mredondo have exchanged messages during the simulation by means of the Structured Chat (e). To be specific, a student is expressing that the consumption remains under the maximum.

The student who has proposed the case for simulation is the one who takes the floor for that run, i.e. he operates the simulation conditions. This alternative allows a turn change mechanism in the different simulation executions that are carried out, due to the fact that the turn assignment is established by the case proposal procedure. Nevertheless, the system can also be configured so that all the participants could interact at the same time, although this option introduces more conflicts than the first one. This student can take part with simulation actions that alter the automatic behaviour of the model (automated house). The actions we have considered are to switch on/off, to open/close, to break a link, to set automatic/manual operation, to introduce human presence in a room or to cause an accident. Accident simulation allows the students to observe the model behaviour when problematic situations arise in order to check if the model reacts in a suitable way. Changes in the model state are reflected immediately on the whiteboards of all the participants in a graphic way and in the lists of interactions in a textual way, making the experimentation space available to all the participants in the same view, something considered essential according to van Joolingen (2000).

The teacher-type users have the pause button to stop the simulation for a moment, for instance, to propose a question, to cause a reflection, etc. and to continue with the simulation afterwards. This, together with the possibility of the teacher taking part in the simulation with any simulation action, offers interesting ways of mediating in the students’ learning.

In DomoSim-TPC the teachers define the problem collection with the help of domain experts, which guarantees the maximum fidelity of the representation of the problem and of the simulation model with reality. This aids the students learning how to face situations like professional designers do.

For each problem the requirements of the model the students have to build are described. A problem includes the following elements: identification, formulation, requirements, constraints, internal and external environment, simulation cases and hypotheses. The internal and external environment consists of a set of parameters that characterize the house and the environmental aspects. Some of these parameters will have an initially assigned value and others will have to be defined by the students as part of the problem solving task. Some parameters will be modifiable and others will not.

This problem library is organized in three complexity levels (lower, medium and high). Thus, the teacher can propose problems with a specific complexity level depending on
the knowledge level of the group. An activity consists of the proposal of a specific problem to a group of students for its solution with a specific help level (lower, medium and high). Problem solving can be considered with different help levels (reinforcement), depending on the previous knowledge of the participants and their process of knowledge construction (Bravo et al., 2002). The help level, together with the problem complexity level, allows the teacher to propose series of problems applying a scaffolding strategy (Rosson and Carroll, 1996).

3.3. Architecture: components and functionality

The different components of the system are illustrated in Fig. 3. There are four subsystems and two types of actors that play a different role: teacher and student. These components offer a functionality depending on the type of agent. For teachers, there is a manager of activities component and a monitoring and analysis component. The first includes three modules: (i) the design problem library, handled with an authoring tool; (ii) the activity manager, in order to configure learning experiences in terms of problem

![Fig. 3. Physical and functional architecture of DomoSim-TPC.](image)
solving activities to be carried out either individually or in a group; and (iii) a group management tool, to organize the users into groups and assign them an activity coordinator teacher. The teachers define the work sessions making use of the Agenda of Sessions, indicating the period of time in which the activities can be carried out. A teacher also has tools for the analysis, evaluation and monitoring of activities (Bravo, 2002) taking into account the models (solutions) built by the students and the actions carried out in this building process, i.e. the traces stored in the DBMS.

The students solve the problems organized in small groups. They interact mainly with two system components: communication and coordination, and design and simulation. The Agenda of Sessions indicates the timetable in which they can work. They can plan and coordinate themselves before approaching the problems using a Chat and/or an Electronic Mail tool. They have a specialized tool of Modelling and Experimentation (M&E), as well as specifically designed tools for communication and coordination support: a Structured Chat and a Decision-Making Tool. Using the same tools the teacher will be able to collaborate with them before the sessions as well as during and after them.

In summary, the developed subsystems (Bravo, 2002) are as follows:

- **Manager of Activities Subsystem.** It manages information in relation to the problem solving activities and to the participants.
- **Monitoring and Analysis Subsystem.** It allows the information registered to be synthesised and analysed while activities are being carried out.
- **Communication and Coordination Subsystem.** It contains synchronous and asynchronous tools for communication and coordination.
- **Design and Simulation Subsystem.** It is used to solve the problems by designing and simulating models.

At a technological level, the architecture follows a client/server approach. The DBMS stores all the data elements necessary to configure the experiences and the results of the processes (traces, solutions, …). The DBMS is accessed by means of JDBC. The Synchronization Server (SS) is a process that supports synchronous collaboration among the users. For this purpose, it uses other processes such as the Interaction Distributor and the Voting and Simulation Servers. This infrastructure is based on the processing and distribution of compressed data packages through sockets on TCP/IP networks, according to a centralized architecture (Bravo et al. 2000b). Other collaborative applications can use this synchronization infrastructure whenever they fulfil the communication protocol that the SS implements, as described by Bravo (2002).

These two centralized services of storage and synchronization can reside in the same server or can be separate. With this configuration the performance and efficiency requirements for the use of the system in the teaching centres are fulfilled. However, this architecture can be extended to support situations with a higher number of users in which this centralization can produce a bottleneck. In that case, distributed DBMS can be used and more instances of SS would be distributed in different computers, assigning different groups of users to each one of them. This procedure can be dynamic, so that the system decides what server to assign to a new group of users depending on its work load. This solution increases the scalability of the proposed approach.
The system has been implemented using Java technology, which facilitates its execution from the Internet, making its use at distance possible. The problem solutions and the simulation traces can be expressed by means of XML-based descriptions, which would facilitate their manipulation with other software tools and the integration of DomoSim-TPC with others systems.

4. The collaboration procedure for modelling and simulation

The M&E Tool is used to build the solution to a domotical problem. As mentioned previously, we propose that the students practice an explicit coordination and collaboration protocol to carry out the learning activities. These protocols can be described by means of a state diagram where each state represents a shared workspace. The workspaces share information, but each one allows a different kind of task to be carried out. In this section we first present the collaboration protocol and the tasks involved, and we then look in more detail at the interaction aspects so as to explain our choices for the user interface, followed by a summary of the DomoSim-TPC support for collaboration.

4.1. Defining the tasks

The first workspace to be accessed when the group starts to solve a problem is the Design one (Fig. 4). In this workspace the students build the model without there initially being any work organization. To organize their work, they have to move to the Work Distribution workspace. This distribution is reflected in a data structure (Current distribution) that the Design workspace supplies and that the Work Distribution workspace returns updated (New distribution). The students also have to select the Parameterisation workspace to define the problem variables there. This workspace receives a data structure containing the current values of the variables (Current variables) and returns the updated values (Updated variables). These two workspaces have access to the model in the design workspace, which
allows them to be adapted to its elements and to carry out checks. When the students consider that they have finished, i.e. their current model is a potential solution, they move to the Cases and Hypotheses workspace. This space has access to the data structures that represent the designed model and the general variables, so that students find these objects when they open the workspace. From this space, once a case is selected, the group can move to the Simulation workspace, described in Section 3.

As illustrated in Fig. 4, this procedure defines a flow, allowing the combination of modelling and experimentation phases. In order for the users to come to an agreement about moving from one workspace to another, a synchronous voting mechanism is available: first, a user proposes to move, and next, when all the users have shown their agreement (with OK votes) the movement to the new workspace is carried out. The group is not obliged to follow a specific sequence of nodes. Workspaces can be visited in any order according to the graph in Fig. 4.

4.2. The interaction approach

Table 2 contains the structure and view of each workspace as well as the interaction mechanisms that allow the task to be carried out. In these workspaces the students carry out the domain tasks by means of elementary actions of direct manipulation. According to Shneiderman (1998), this interaction paradigm accelerates learning, facilitates

<table>
<thead>
<tr>
<th>General Task</th>
<th>Task/ workspace</th>
<th>Structure</th>
<th>View</th>
<th>Direct manipulation mechanisms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modelling</td>
<td>Design</td>
<td>Electronic whiteboard and toolbars</td>
<td>Graphic representation of the model</td>
<td>Electronic whiteboard Object–action model Objects: domain operators Actions: edit, parameterise and link</td>
</tr>
<tr>
<td></td>
<td>Work Distribution</td>
<td>Criterion and assignments</td>
<td>Lists of design actions and assigned actions</td>
<td>Object–action model Objects: criteria, users and types of design actions Actions: propose, ok, not ok</td>
</tr>
<tr>
<td></td>
<td>Parameterisation</td>
<td>Parameter list</td>
<td>Parameter list</td>
<td>Object–action model Object: parameters and values Actions: propose, ok, not ok</td>
</tr>
<tr>
<td>Experimentation</td>
<td>Cases and Hypotheses</td>
<td>Simulation cases and hypotheses</td>
<td>Graphic representation of the model, and cases and hypotheses lists</td>
<td>Object–action model Objects: cases and hypotheses Actions: propose, ok, not ok</td>
</tr>
<tr>
<td></td>
<td>Simulation</td>
<td>Electronic whiteboard, simulation action toolbar and information panels</td>
<td>Graphic representation of the model and of the environmental information</td>
<td>Electronic whiteboard Object–action model Objects: domain operators Actions: on/off, open/close, manual/automatic, …</td>
</tr>
</tbody>
</table>
the intuitive and effective use of the systems and reduces and prevents errors. In Design and in Simulation tasks, the collaborative electronic whiteboard metaphor based on the object-action model is followed, which provides a structure for increasing the user’s control (Cox and Walker, 1993). The objects in the design activity are the domotical operators (activators, receivers, systems, links and other elements) and the user actions are to insert, select, delete or move operators, operator parameterisation and linking operators.

The other workspaces are also based on the object–action model, but here the objects manipulated by the users are forms and the actions are a set of speech acts, available as a set of buttons. In the case of the Work Distribution workspace the objects are the different criteria and elements (users and types of design actions) for dividing and assigning the work, and the actions (buttons) are for proposing, agreeing with or disagreeing with a particular distribution. For the Parameterisation workspace the objects are the parameters, and the actions are to propose values and to reply to the proposals. For the Cases and Hypotheses workspace, as detailed in Section 3, the objects are the cases and hypotheses and the actions to propose, agree with or disagree with a particular selection or value.

To materialize the negotiation processes in these workspaces we follow the Language/Action Perspective (Winograd, 1988), where language is postulated as a primary dimension of human cooperative activity. Actions are carried out by means of buttons that represent speech acts: Propose (Proponer), Agreement (OK), Disagreement (No OK), Simulate (Simular) and Confirm (Confirmar). The negotiation processes are deployed according to the graph in Fig. 5: a first student (student i) carries out a proposal (the value for a hypothesis, a case simulation, …) and the rest of the students can reply either with an agreement (student j) or disagreement (student k); alternatively they can abstain. If no student explicitly disagrees, the proposal is carried out. The state diagram is enforced by the system, but it keeps one instance for each user’s proposal, so that the users can answer to the proposal they have previously selected.

4.3. Support for collaboration

DomoSim-TPC is a particular case of a CMSL environment from which we can extract a task organization. Two interdependent cognitive tasks are identified: to solve the problem and to collaborate. In order to solve the problem, specific domain actions are
used. Collaboration requires communication and coordination to exchange information in relation to the domain, to coordinate actions and to reach agreements. In Table 3 the collaborative support considered in DomoSim-TPC is shown.

The domain tasks are implemented by means of the shared workspaces of the M&E Tool previously presented, in which the users collaborate synchronously. The collaboration tasks are materialized with synchronous and asynchronous tools. The Generic Chat, the Electronic Mail and the Agenda of Sessions are used before the problem solving sessions so that the students communicate and coordinate among themselves.

The Decision-Making Tool and the Structured Chat together with the awareness support are available in all the workspaces, except for the Cases and Hypotheses workspace, which does not offer communication tools because it is used for a very specific task. Thus, the support for both kinds of tasks (domain and collaboration) is integrated in each workspace.

The Structured Chat limits the conversation to a pre-established set of communication acts that generally encourage learning and guide design, according to Flexible Structuring (Baker and Lund, 1996). This approach is very restricted and, in line with the first system evaluations by teachers and students, a function that allows them to send any text was incorporated. The Decision-Making Tool is a general purpose tool that allows users to reach agreements by means of voting processes.

The awareness techniques, integrated in all the workspaces, are aimed to facilitate real time collaboration, contributing to the perception and knowledge of the interaction that the users carry out in a shared workspace. This is achieved with three tools: (i) tele-pointers, which are representations of the mouse pointers of the remote users, (ii) a Session Panel, which keeps a user list (name and pictures) and the state of each one, in order to know what each user is doing (editing, selecting, communicating, …), and (iii) a list of the interactions carried out by other students, which are visualized in text format. In order to implement the tele-pointers, a process in the user client tool communicates the user mouse pointer coordinates to the Interaction Distributor, which distributes them to the rest of the user processes, so that they can show the mouse pointer of that user. The tele-pointer and the corresponding user’s name in the Session Panel are shown in the same colour to aid identification. All these elements allow users to know what the other students are doing and where they are doing it, so that their possible intentions could be predicted.

<table>
<thead>
<tr>
<th>Support</th>
<th>Tools</th>
<th>Type of interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication</td>
<td>Generic Chat</td>
<td>Synchronous</td>
</tr>
<tr>
<td></td>
<td>Electronic Mail</td>
<td>Asynchronous</td>
</tr>
<tr>
<td></td>
<td>Structured Chat</td>
<td>Synchronous</td>
</tr>
<tr>
<td>Coordination</td>
<td>Decision-Making Tool</td>
<td>Synchronous</td>
</tr>
<tr>
<td></td>
<td>Agenda of Sessions</td>
<td>Asynchronous</td>
</tr>
<tr>
<td>Awareness</td>
<td>Tele-pointers</td>
<td>Synchronous</td>
</tr>
<tr>
<td></td>
<td>List of Interactions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Session Panel</td>
<td></td>
</tr>
</tbody>
</table>

Table 3
Collaborative support in DomoSim-TPC
5. System evaluation

In the development of DomoSim-TPC we have used a software engineering approach inspired by a participative, iterative and prototype-based design. During this development different formative (Tessmer, 1993) and heuristic (Baker et al., 2000) evaluations have been carried out, which have allowed us to find out and fix mismatches in the functionality and the usability of the system.

5.1. Evaluation experiments

In order to evaluate DomoSim-TPC, a total number of 69 test activities have been carried out during the academic year 2001–2002, in which 50 students, 2 teachers, 2 domotical experts and one usability expert have taken part. The results of these tests have been analysed in depth by Bravo (2002).

Some evaluation exercises have been carried out in real learning situations during a four-month period, in which 14 students from two teaching centres organized in pairs participated (one from each centre). Fig. 6(a) shows the average solution time (design and simulation) by room and management area (thermal and luminosity comfort, energy control, etc.) of five problems (p1–p5) of growing complexity solved by seven pairs of students. We observed that the time used decreases, although the improvement does not take place until the second problem, due to the learning curve in relation to using the system: p1 is the first problem the students approach after their process of training with DomoSim-TPC and they still find it difficult to work quickly and efficiently. Fig. 6(b) indicates the average error percentage made in the solving of the same problems. We consider an error as the inclusion in the solution of an incorrect operator. A tendency for this percentage to decrease is observed, with the exception of problem p4.

Fig. 6 shows that the necessary time to solve a problem (Fig. 6(a)) and the number of errors that the students make (Fig. 6(b)) decrease as the users make progress in the problem solving activities. This is an indication that the students solve problems better and faster each time.

Fig. 6. (a) Average time employed in the solution of the problems p1–p5 (left). (b) Average error number in the solution of the problems p1–p5 (right).
5.2. Comparison with other systems

Table 4 summarizes the different features of the studied systems that use collaboration and simulation for learning. The table columns identify the following aspects:

- **Systems/Domain.** This column shows the system name and the application domain.
- **Model/Problem.** They indicate, respectively, if the system is based on the design of a model and on the solution of a problem.
- **Traces.** It indicates if the students’ interactions and the simulation events are registered.
- **Collaboration.** Tools and mechanisms provided to support collaboration.
- **Awareness.** Awareness techniques that are included.
- **Analysis.** Mechanisms available for the analysis of the processes carried out and the result.

In ERCIS, SIMPLE, WebNet, CROS and TurboTurtle a single model incorporated in the system is simulated. In SESAM different models can be generated by means of a description language that has to be compiled. In SIMPLEZ the program to simulate can be defined using an editor, whereas in DomoSim-TPC and LESP the students themselves collaboratively design the model to simulate by means of direct manipulation.

In our proposal a collection of problems to be solved by the students is available. ERCIS, SESAM and TurboTurtle are also based on the setting out of problems to the students, but in DomoSim-TPC the problems are defined and outlined by the teachers with the help of authoring tools. In DomoSim-TPC the teachers also have tools for organizing the groups and defining problem solving activities.

Most of the systems register the interventions that the users carry out and the changes in state of the simulated model, but they only allow the exchange and the revision of these traces. In DomoSim-TPC these events are also stored for their query, and automatic tools can also be used to carry out quantitative and qualitative analysis (Bravo, 2002).

DomoSim-TPC offers a wide range of collaborative tools. It has asynchronous tools, such as Electronic Mail and Agenda of Sessions, as well as synchronous tools, such as Structured Chat, Decision-Making Tool and all the interaction mechanisms of the M&E workspaces. Most of the systems use electronic mail and chat tools. Among the systems studied we highlight the distributed simulation of ERCIS and the shared simulation of TurboTurtle.

With respect to awareness, the majority of the systems studied are based on asynchronous collaboration, so that the awareness characteristics they offer are limited. Only ERCIS and TurboTurtle show a panel with remote information about the participants in the simulation. TurboTurtle incorporates tele-pointers and tele-data as well. As advantages of DomoSim-TPC we point out the user’s picture incorporated in the Session Panel and the use of lists of interactions.
<table>
<thead>
<tr>
<th>Systems/domain</th>
<th>Model/problem</th>
<th>Traces</th>
<th>Collaboration</th>
<th>Awareness</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERCIS/Military Applications</td>
<td>No/Yes</td>
<td>Yes (simulation)</td>
<td>Distributed simulation</td>
<td>Remote information panel</td>
<td>Revision of the simulation trace</td>
</tr>
<tr>
<td>SESAM/Software engineering</td>
<td>Yes/Yes</td>
<td>Yes (simulation)</td>
<td>Rule monitor, Natural language communication</td>
<td>–</td>
<td>Revision of the simulation trace</td>
</tr>
<tr>
<td>SIMPLE/Engineering</td>
<td>No/No</td>
<td>Yes (simulation)</td>
<td>Electronic mail</td>
<td>–</td>
<td>Revision of the simulation trace</td>
</tr>
<tr>
<td>Semiconductors WebNet/LAN Design</td>
<td>No/No</td>
<td>No</td>
<td>Electronic mail, Interactive glossary</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>LESP/Particle Models of Matter (physics)</td>
<td>Yes/No</td>
<td>Yes (simulation)</td>
<td>Electronic mail</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>SIMPLEZ/Computer Architecture</td>
<td>Yes/No</td>
<td>Yes (simulation)</td>
<td>Electronic mail, News board, Chat</td>
<td>–</td>
<td>Revision of the simulation trace</td>
</tr>
<tr>
<td>CROS/Electricity</td>
<td>No/No</td>
<td>No</td>
<td>Chat</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>TurboTurtle/Newtonian Physics</td>
<td>No/Yes</td>
<td>No</td>
<td>Shared simulation</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>DomoSim-TPC/Domotics</td>
<td>Yes/Yes</td>
<td>Yes (modelling, simulation and collaboration)</td>
<td>Electronic Mail, Agenda of Sessions, Recent News, Board Chat, Structured Chat, Synchronous Design and Simulation (Shared Simulation)</td>
<td>Te-data, Tele-pointers, Participants’ names</td>
<td>Traces of design, simulation and collaboration; Qualitative and quantitative analysis of interaction</td>
</tr>
</tbody>
</table>
6. Conclusion

6.1. Summary

In this work we have proposed a set of features that allows network-supported collaborative learning environments based on simulation (CMSL environments) to be explored and categorized. The systems studied prove that this synergy is being applied positively in collaborative learning processes.

DomoSim-TPC is a synchronous distributed CMSL environment for the teaching and learning of Domotics. With its development we have aimed, on the one hand, to overcome a situation of lack of educational tools for this domain and, on the other hand, to contribute to the scientific community with a proposal of a CMSL environment that can inspire the development of systems handling other design domains. In this way:

- We have created an environment based on modelling and simulation that supports students and teachers to develop a problem-based learning method.
- To that end, we have defined a collaboration procedure, the shared workspaces and the tools that allow the students to carry out the domain tasks and the collaboration tasks.
- We have described and justified the direct manipulation, communication, coordination and awareness techniques required for a collaborative modelling and simulation system.
- We have defined an architecture for real time CMSL systems that can be generalized in different dimensions, either to other learning tasks or domains. The management tools and the communication and coordination tools are components directly usable for other systems. The system architecture, the general collaboration procedure, the model of collaborative simulation in real time with design of the scenario as well as the awareness techniques are useful outside of this system and constitute a framework proposal for the development of learning environments by means of modelling and simulation.
- DomoSim-TPC has been subjected to evaluation and has been used in real learning experiences. The different evaluations have allowed the evolution of the product, the debugging of errors and the generation of improvements. During the experiments we have checked that besides learning how to model and to carry out simulations, the students should learn how to collaborate, communicating and coordinating among themselves. When the students collaborate, they obtain better solutions, make fewer errors and work faster.

The opinion of the teachers is that the system is useful and very necessary. They consider that it is a valuable complement to the theoretical teaching of Domotics, since at the moment they do not have well equipped laboratories that allow them to carry out adequate practices. However, they indicate that the configuration of experiences requires a small effort.

We have compared our environment with other CMSL environments, although in fact there are not many that incorporate real time collaboration for modelling and simulation,
and principled support to carry out a PBL approach. Therefore, we believe that our proposal is more complete and innovative from a learning perspective.

6.2. Future work

Our current agenda focuses on reusability aspects of the data and outcomes of the different components as well as system generalization. We will use XML-based specification languages for a generic definition of design case studies (Bravo et al., 2003). This level of abstraction to describe object models and behaviour open the way to find other domains in which to apply the DomoSim-TPC approach. As part of this work, a specification technique for discrete-event simulation models, an authoring tool to facilitate the building of this kind of specifications and a framework that processes this specification to generate a collaborative simulator are being designed.

Acknowledgements

This work has been supported in part by the Ministerio de Educación y Cultura (Spain) in the TIC2002-01387 project and by the Comunidad Autónoma de Castilla-La Mancha (Spain) in the PBI-02-026 project.

References

Bravo C, Redondo MA, Ortega M, Verdejo MF. Collaborative environments for the learning of design: A model and a case study in Domotics. Computers and Education; [in press].


Redondo MA. Planificación Colaborativa del Diseño en Entornos de Simulación para el Aprendizaje a Distancia. Doctoral thesis, Computer science department, University of Castilla—La Mancha. ProQuest information and learning (Current Research); 2002.


Schneider K, Nakakoji K. Collaborative learning as interplay between simulation model builder and player, In: Proceedings of CSCL’95, the first international conference on computer support for collaborative learning; 1995.


Crescencio Bravo received his MSc in Computer Science (1996) from Universidad de Sevilla (Spain) and his PhD (2002) from Universidad de Castilla—La Mancha (Spain). He joined the Computer Science Engineering Faculty of the Universidad de Castilla—La Mancha in 1998. His research interests include computer-support for Collaborative Learning and Cooperative Work and Computer—Human Interaction.

Miguel Á. Redondo received his MSc in Computer Science (1997) from Universidad de Granada (Spain) and his PhD (2002) from Universidad de Castilla—La Mancha (Spain). He joined the Computer Science Engineering Faculty of the Universidad de Castilla—La Mancha in 1999. His research focuses on the fields of New Information Technologies applied to Collaborative Learning and Computer—Human Interaction.

Manuel Ortega received his MSc in Science (1982) and PhD (1990) from the Universidad Autónoma de Barcelona (Spain). He joined the Computer Science Engineering Faculty of Universidad de Castilla—La Mancha (Spain) in 1990. His major research areas are in the fields of New Information Technologies applied to Collaborative Learning and Computer–Human Interaction. He is the head of the CHICO Research Group (http://chico.inf-cr.uclm.es).

M. Felisa Verdejo received her MSc in Science (1972) from the Universidad Complutense de Madrid (Spain), PhD (1975) from the University Pierre et Marie Curie (France) and PhD (1980) from the Universidad Complutense de Madrid (Spain). She joined the Universidad Nacional de Educación a Distancia (Spain) in 1992. Her research interests includes the field of knowledge-based and collaborative technologies to support human activity in distance learning scenarios. She is the head of the LTCS Research Group (http://ltcs.lsi.uned.es/).