A system for the specification and development of an environment for distributed CSCL scenarios

M.F. Verdejo, B. Barros, T. Read, M. Rodriguez-Artacho

Departamento de Lenguajes y Sistemas Informáticos, U.N.E.D
Ciudad Universitaria s/n, 28040 Madrid, Spain
{felisa,bbarros,tread,martacho}@lsi.uned.es

Abstract. In this paper the 'Active Document system', grounded in the Activity Theory, is presented. This system serves both as a representational framework for the description of learning activities and a harness for the mechanisms necessary to support the creation and management of the corresponding computer-based scenarios. This system is made up of three components, firstly, a set of authoring tools for the creation and configuration of the different Active Documents required for specifying the learning activities. Secondly, a (distributed) repository of learning objects that consists of a variety of tools and resources for the described activities. Thirdly, an Active Document architecture that manages the Active Documents and generates the user environment necessary in order to carry out the described activities, together with the appropriate resources and tools. This proposal appears to offer a solution to the problems of producing reusable and customizible computer-based learning environments.

Introduction

A considerable amount of general-purpose (and domain specific) scientific software has proved to be highly suitable for learning scientific and technological principals; software such as visualizers, simulators and modelling tools. Furthermore, an increasing number of interactive and collaborative tools have become technically affordable for a wide spectrum of the educational community, opening up the possibility to support social constructivist learning approaches in computer-based environments. However, the production of a customized learning environment is still very time-consuming where even the complete coverage of a single subject area would require a very large amount of effort. The goal of the research presented in this paper is to provide a computational model and an underlying technological infrastructure that will permit the design and development of collaborative learning activities that involve a variety of resources. The approach adopted here is grounded in the Activity Theory (henceforth, AT) because it captures the social perspective of a learning community, and provides a unified view of how to specify a learning activity at a conceptual level: the different actors and their responsibilities, the context, the

learning goals and the mediating tools. An authoring paradigm is presented that could meet the needs of various actors and which separates technical aspects of software creation from the design of collaborative learning activities. The focus of this work is placed on learning experimental sciences where there is a pressing need for students to improve their learning processes with a better articulation of theory and practice throughout the academic year, especially in the context of distance learning. A way forward is to engage the students in a variety of activities, including the performance of experiments either in real or virtual settings, supported by a distributed collaborative computer environment. The premise here is to offer a persistent, structured, dynamic, active and personal work space to sustain their constructs in a long term learning process. For this purpose, the interoperability of tools and outcomes will be a central issue to be addressed.

Related Work

This work is related to three active research areas: (1) AT (2) sharing and interoperability issues, and (3) cognitive and communication tools to support active learning processes. What follows is a brief summary of ongoing initiatives that address these topics. AT views cognition as being a social activity. Individuals work and learn in groups and communities with organizational structure, interacting with others, using tools and resources, and following rules according to roles in order to perform purposeful actions. Socially oriented constructivism forms the basis of the Computer Supported Collaborative Learning paradigm (henceforth, CSCL). CSCL settings can take a great variety of forms. AT has proved to be a useful framework to describe and analyse collaborative settings [5][1]. However, very little work [2][10] has been undertaken which embodies AT in a computer model. The proposal presented here uses AT as the specification language for an authoring system that defines collaborative learning scenarios. The output of the complete authoring process is a set of XML data structures describing the learning activities, i.e., the community involved, the tasks and their interrelationship, the different roles participants can play with the tools to be used, as well as the objects of the activities. A system, implemented in Java, dynamically creates the learning environment specified by these data structures providing learners with an integrated workspace for the development of activities.

During the past seven years, a number of initiatives have been undertaken in order to define metadata schemas. Starting with Dublin Core (henceforth, DC), DC 1.0 in 1996, different communities have produced a large number of data and resource descriptions such as the educational metadata and content packaging specification proposed by the IMS project to describe learning resources, intended to be used for indexing and retrieval purposes in distributed repositories, to integrate multiple content on computer-based training courses. Extensible Markup Language (henceforth, XML), developed under the auspices of the WWW Consortium, offers amongst other things, the possibility of exchanging structured data between applications. Specifications such as DC and IMS can be expressed using XML. Developing standards for describing and reusing learning objects have crystallized in

groups and committees such as the IEEE LTSC or the more recent initiative of CEN/ISSS. In parallel with the work being undertaken in this area, other research groups [7] have selected a software engineering framework that focuses on the definition of a component-based architecture, in order to support the building of interactive educational applications from software repositories. Their approach is to develop small components, typically developed in Java, which can be combined by powerful connections and sharing mechanisms (e.g., the ESCOT project, [3]). These objects are assembled in order to generate specific applications. The challenges here are to determine the right level of component granularity and connectivity for the application designer, in this case an educator, using an authoring-tool to assemble pieces in order to generate a domain oriented learning environment without the need to create or modify programs.

Other proposals such as [9], deal specifically with mechanisms supporting a tight integration of communication and task-oriented tools, to enhance the cognitive support in collaborative learning settings. Furthermore, the problem of how to combine diverse representational systems in a generic way by adding semantics without assuming a particular knowledge domain still exists. The benefit of multiple ways of viewing and manipulating objects using a variety of representations for learning purposes has been analysed by [8]. These alternative representational forms contribute to the idea of giving an active dimension to content objects, i.e., the meaning of the content would depend upon the domain context giving rise to a variety of perspectives but, most important, by being partially interoperable. Furthermore, the enrichment of not only the content behaviour but also the relations between objects has recently been explored in other tools [4] in a way that integrates highly specialized tools with a general-purpose cooperative visual language-based environment. The potential of visual language-based connective workspaces with intelligent plug-in components should be further pursued in order to reach more generic solutions.

The structure of the rest of the paper is as follows: in the next section the overall approach adopted in this work is described, what will be referred to as the Active Document (henceforth, AD) system. Subsequently, the potential of this system will be illustrated with examples taken from a learning scenario in the domain of chemistry, and finally, technical implementation issues about the underlying system architecture are discussed. We conclude with a summary and an outline of future work.

The Active Document system

The AD system provides both a representational framework for the description of learning activities together with a harness for the mechanisms necessary to support the creation and management of the corresponding computer-based scenarios. Using this approach, a set of learning activities can be formally described. The units of description are based on AT, explicitly considering all the concepts involved in an activity: the division of labour (tasks and subtasks), the mediating tools (learning objects), the norms (partially captured in terms of roles), the object as well as the community. The formalism for the description is inspired by the recent paradigm of

Educative Modelling Languages, and by the experience of the development of an EML in this research group [6].

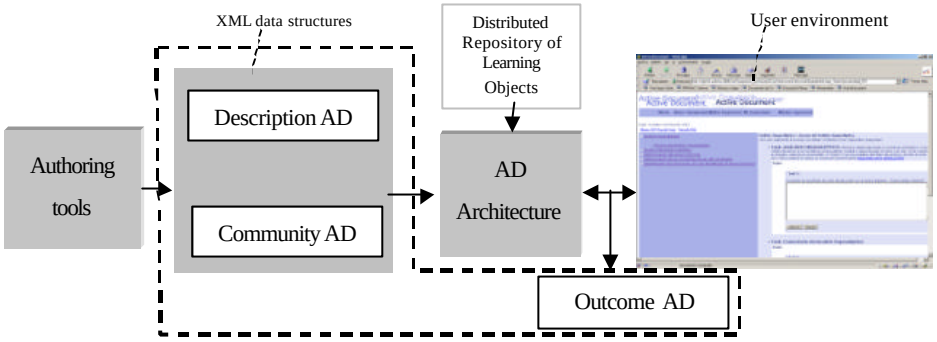

Fig. 1. The Active Document system

The AD system (figure 1) consists of a set of components necessary to create and process the ADs. The main components are: (1) Authoring tools for the creation and configuration of the different ADs required for specifying the learning activities. The authoring tools are currently XML editors which allow a lecturer to compose a set of ADs for specific learning applications. (2) A (distributed) repository of learning objects that consists of a variety of tools and resources for the described activities. It can include generic tools (such as editors) and specific tools like simulators or tele-operational devices, as well as domain-specific content repositories of semantically linked material. (3) An AD architecture that manages the ADs and generates the user environment necessary in order to carry out the described activities, together with the appropriate resources and tools.

The specification of an Active Document

The concept of an AD includes three aspects of the learning process: a description of the activities, a description of the communities and the outcome of the work undertaken within the environment. The ADs are specified in XML and are defined by three pairs of document type definitions (or DTDs) and their corresponding XML document. The ADs are: (1) The description of the division of labour in the tasks and subtasks (referred to as the 'Description AD'). (2) The actors and roles involved in the collaborative tasks (referred to as the 'Community AD'). (3) The outcome of the activity (referred to as the 'Outcome AD'). As can be seen in figure 1, the Description AD, along with the specification of the actors that perform the collaborative activities (specified in the Community AD) are interpreted by the AD architecture that dynamically creates the appropriate user interface, according to the elements defined in the two XML structures. As the learning activity proceeds, the outcome produced by each student is represented in XML in the Outcome AD which stores the results of the learning process and the task structure described in the Description AD. These

three ADs are described next. The 'Description AD' specifies a collection of activities, each of which reflect the components of an activity as described by AT, modelling the division of labour and the mediating tools associated with each task. Activities can be grouped within this AD, to provide (optional) sequencing and prerequisite dependences between *groups* of activities. The definition of an activity includes the following: (a) The description of the object of the activity. (b)The specification of the tasks and subtasks, and for each one (if applicable) the different roles that the participants involved in the task can play. (c) The tools and resources available for each role related to a task.

```
<activity id="Act_2" name="Activity Title>
  Description of the activity. Formatted text, graphics, external
  documents, etc., could be  also inserted here.
<taskbyrole id="Task_1" name="Task  1 Title" roles="student, teacher">
           Description of the task 1
         <mediating_tools>
             <resource_ref id="ref_tool 1" id_ref="Spectrum_Editor"
                display="inside"  label="Spectrum Editor Tool">
             <parameter>
               <param name="text" value="Comment this result" />
               <param name="image" value="Act_1;Task_3" />
             </parameter>
             </resource_ref>
       </mediating_tools>
    </taskbyrole>
    <taskbyrole id="Task_2" name="Task  2 Title" roles="teacher">
           <mediating_tools …>
                      <resource_ref … />  <resource_ref … />
          <mediating_tools …/>
    </taskbyrole>
</activity>
```

Fig. 2. An excerpt of an activity definition

Figure 2 shows an example of the components of the activity definition in the Description AD. The XML fragment would be interpreted by the AD architecture in order to produce a user interface for the learning environment where the activity called "Act_02" can be carried out, whose description can be seen to be divided into a set of tasks (Task_1 and Task_2), each of which to be performed by subjects in one or more roles (two roles are shown in the example: student and teacher). The description of this "task by role" (`taskbyrole` tag) consists of a description of the task to be carried out, followed by the declaration of the available tools, referred to as a type of resource. Each role involved in the activity will have its appropriate task by role definition. The definition of the Community AD will provide user assignment for the tasks. The `resource_ref` tag is a reference to a tool, also providing parameter values. The tool used in `Task_1` is a collaborative graphical editor used here to annotate and interpret a chemical spectrum. Other possible resources include external document repositories or different types of tools. The objects generated as a result of an activity can be considered as input for another activity. For instance, `Task_1` of `Act_02` uses of the previously selected spectrum (the result of `Task_3` in `Act_01`) as the input for the collaborative graphical editor. As well as the activities, the Description AD also can include:

- The definition of the overall structure of the learning scenario in which the activities are embedded. This structure is generic for a set of learning scenarios. In the case of a laboratory setting, this would include: the specification of an experiment with a fixed pattern to describe its aim, a theoretical component and safety guidelines, and one of more activities as described above.
- Content elements: A domain-specific repository that can be accessed to provide a semantically linked material.

These components are also expressed in XML in a similar way as an activity, but are not shown in the previous figure. **The Community AD** represents the activity organization in order to describe the assignment of roles for a specific task to the members of a given community. For each activity, a description of the community involved is provided. As has been previously stated, this description is processed by the AD architecture in combination with the Description AD in order to relate the appropriate tasks and tools to the corresponding members of the community. The use of a separate XML structure for the community gives rise to two interesting mechanisms: firstly, communities can change during the development of the activity, thus allowing dynamic role assignments to be made (amongst other possibilities); and secondly, different Community AD can be combined with the same Description AD, providing a flexible mechanism for the re-use of the same division of labour description for a set of different working groups. The **Outcome AD** specifies the way in which the results of the tasks performed in the environment are stored, thus providing an *active* component, a vision of the current work completed and in-progress. Thus, the Outcome AD is in fact the real *active* component of the AD organization, i.e., it is the result of the work generated by a specific actor involved in the activity described by the Description AD.This representation provides a definition, at the desired level of detail, of the work and the objects generated during the learning process. The Outcome AD, rather than a sequence of plain text can contain complex elements like graphics, tables, structured dialogs, maps, etc., in an XML format embedded into it, with links to non XML objects outside, e.g., a MS Word document. Furthermore, the AD architecture makes this structured collection of heterogeneous objects *persistent* during the life-cycle of the user within the environment, providing tools for their manipulation, storage and retrieval. This mechanism forms the basis for passing objects between tools in a transparent way for the user. Some interesting applications can be considered due to the nature of the Outcome AD representation. In the case of an experiment, it could for example, facilitate the creation of a report by the simple selection and copying of the relevant embedded objects once the experiment has terminated. The organization of the Outcome AD reproduces the structure of the Description AD in terms of the structure of activities and tasks, but differs from it in the sense of having an `outcome` tag for each of the performed tasks. Furthermore, there is an outcome AD XML structure for each actor involved in the activities described above.

Example

The definition of the AD presented above is sufficient to configure many different types of group activities. However, the aim here is to explore its capabilities for collaborative learning in experimental scenarios. What follows is an example of an activity, a part of an experiment, which has been developed to explore the current version of the AD system for a second year degree course in Organic Chemistry. An *experiment* usually involves several *activities* composed of *subtasks*. For each subtask, there are some indications about the particular constraints as well as the different possible resources and tools available to perform them. The experiment in question is the analysis and identification of the characteristics of a given chemical substance. Each group of students has to analyse a different substance, selecting the adequate subtasks and necessary steps. A *subtask* can be defined to be a basic (individual or group) action such as testing the material in the lab, annotating the results, deciding the next step to be done, reflecting on the results, elaborating a conclusion (considering the evidence), simulating the behaviour of a substance in the lab, etc. In the Description AD, one or more tools are associated with each subtask in order to carry out each action. These tools are invoked automatically and transparently in the environment when the students choose to undertake the corresponding subtask. What follows is a description of one of the activities of this experiment.

In this **activity** the student has to identify the compounds of the substance in question and from there infer what the substance actually is. This activity is composed of both individual and collaborative subtasks. It should be noted that each stage of the chemical analysis undertaken in this activity, which should eventually lead to the correct identification of the substance in question, is based upon the previous one. Therefore, depending on the result of a particular subtask, the students have to reflect upon the results obtained, describe the working situation, dismiss possibilities and decide upon the nature of next step to be taken in the analysis process Since these subtasks are essentially reflective in nature it is more appropriate to undertake them as a group. The structure of this activity can be seen in the environment on the left hand side of the figure 3. Students have to the select the most appropriate spectrum for this substance, based upon the data each student has gathered up until now. This subtask is undertaken collectively using the text-based collaborative discussion tool shown in the figure. As can be seen, from this tool the students can access a glossary of images of chemical spectra available for this choice. Once the decision has been made, the image selected from the glossary is automatically loaded into the collaborative visual markup tool to enable the students to perform the next task: the identification and characterization of the distinctive peaks within the spectrum. This is shown at the bottom right hand side of figure 3. Once the analysis of the spectrum of the substance has been undertaken, making use of the results of the previous activities, the students are finally in a position to decide upon the name of the substance. A visualization of the current state of the Outcome AD for this endeavor can be seen at the top right hand side of the figure 3. It should be noted that whilst some parts of this process have been collaborative in nature, the students have personal copies of the Outcome ADs that reflect their overall contributions.

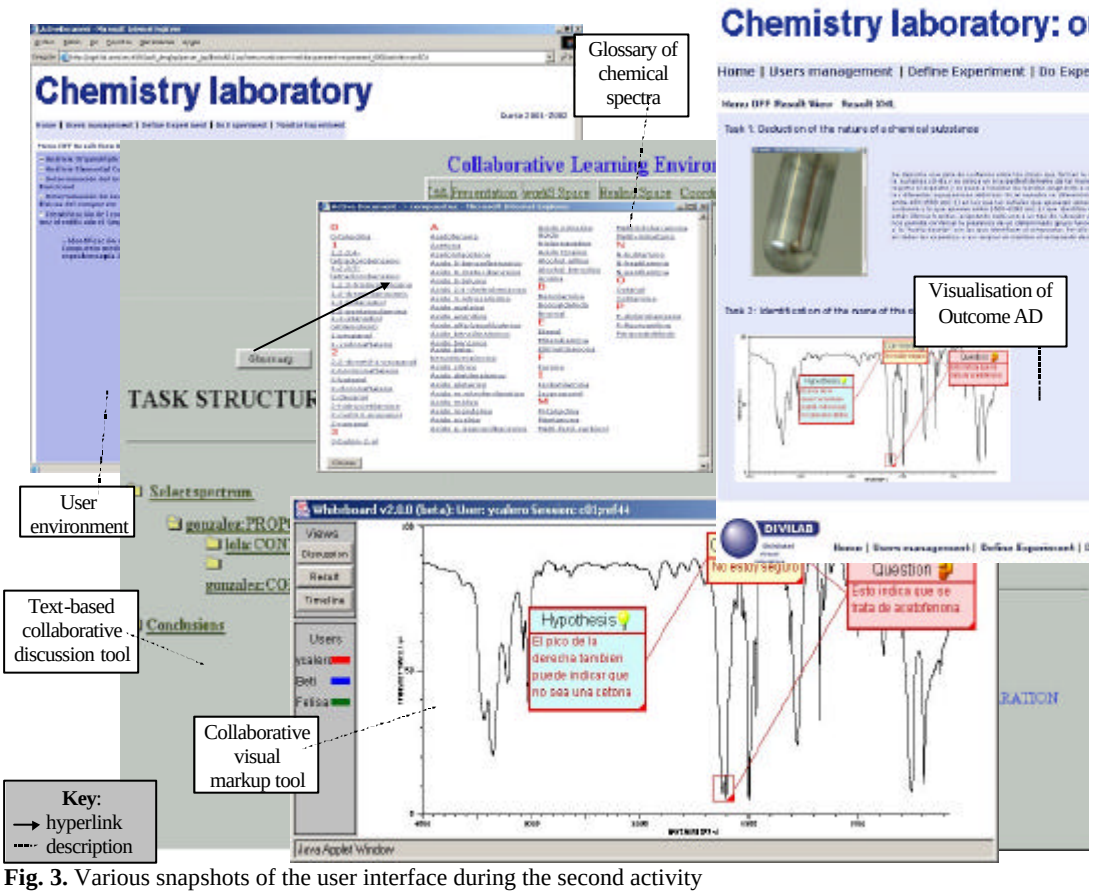

Fig. 3. Various snapshots of the user interface during the second activity

The Architecture

As can be seen in figure 4, the architecture is divided into various levels in order to accentuate operational flexibility and provide content reusability. This stratification has lead to the definition of the levels as follows: firstly, the presentation level completely separates aspects of the generation and management of the interface from other system functionality and also maintains persistent data during user sessions. Secondly, the configuration level undertakes the control of the data structures necessary both to manage the persistent user sessions and control the overall structure of the interface of the system. Thirdly, the application level manages the interchange of data between the external applications and the system during the experiments, which gives rise to the dynamic and active characteristics of the system. Fourthly and finally, the control level handles the low-level data interchange between the system and the underlying database. This architecture has been designed and developed using a combination of Java and XML technologies, where each functional level shown in the figure consists of several underlying components.

There have been three main goals in this design process. Firstly, the specification of each level enables parts of the system to be redefined without affecting overall functionality (for example, providing wider scenario scope by simply adding new presentation layer logic to permit Java-enabled hand-held devices or personal data assistants to connect to the system). Secondly, the declarative nature of the specification of the educational content in XML greatly simplifies its production and enhances its reusability. Thirdly, the production of a system that is portable between different computer hardware and operating systems with minimal configuration changes. The tools use established XML data standards to represent results in order to facilitate the reuse of the data. A couple of examples can be seen in the form of XHTML to represent formatted data such as paragraphs and tables and the use of Scalable Vector Graphics (henceforth, SVG) to represent the results of a graphical tools (such as a visual collaborative discussion tool), enabling the results to be viewed directly in any SVG-enabled tool (such as MS Internet Explorer) or automatically converted into other graphical formats, such as GIF or JPEG, for subsequent reuse in other applications (such as MS Word). As the user advances through an experiment the results of each tool used for a given task are incorporated into the outcome AD which 'grows' to reflect this progress. The control level serves as a transparent and persistent low-level data management device for the AD. The current version of the system can use any relational database that has a JDBC driver. The XML data that make up the various versions of the AD is collapsed into columns within the database. The standard limitation of this approach, that of not being able to search the XML data structures before extracting them from the database, is not a problem because additional columns in each table act as the search keys for data access.

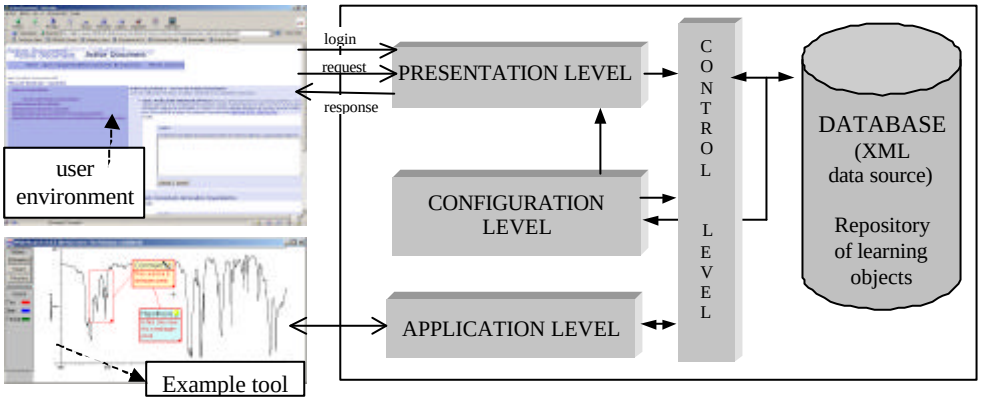

Fig. 4. The architecture of the AD system

Summary and future work

In this paper the 'Active Document system' has been presented for the design and development of resource-based collaborative learning activities. The two aspects most important in this work are: the computational specification of AT, which leads to an educational modelling language, which has been used to specify the ADs, and secondly the Outcome AD, which is the really

active part of the system in the sense that it grows as the student progresses through the learning activities and represents the results of the work as objects that are reused by subsequent tools. Whilst its current domain of application has been that of an experimental science, namely chemistry laboratory sessions, it will also be applied to other non-science evaluation scenarios which should provide valuable insights into the ways in which it can be extended and improved. Finally, the presentation level of the AD architecture and the authoring tools that define the ADs can be identified as candidates for future work. The former needs to be extended to support non-desktop information devices computers, and the latter, needs to evolve from standard XML editors into a fully functional scenario authoring environment.

References

1. B. Barros & M.F. Verdejo. 2000. "Analysing students interactions process for improving collaboration. The DEGREE approach". In *International Journal of Artificial Intelligence in Education*, vol 11, pp. 221-241.
2. G. Bourguin & A. Derycke. 2001. "Integrating the CSCL Activities into Virtual Campuses: Foundations of a new infrastructure for distributed collective activities". In P. Dillenbourg, A. Eurelings, K. Hakkarainen (eds.) *European Perspectives on Computer-supported Collaborative learning*, pp. 123-130.
3. C. DiGiano & J. Roschelle. 2000. "Rapid-assembly componentware for education". *Proceedings of the Int.Workshop on Advanced Learning Technologies*. IEEE Computer Society Press.
4. U. Hoppe, K. Gassner & N. Pinkwart. 2000. "Augmenting cooperative visual languages environments with object contexts and process semantics". In *Proceedings of New Technologies for Collaborative Learning NTLC 2000*, pp. 63-70.
5. K. Issroff & E. Scanlon. 2001. "Case studies revisited: what can activity theory offer?" In P. Dillenbourg, A. Eurelings, K. Hakkarainen (eds.) *European Perspectives on Computer-supported Collaborative learning*, pp. 316-323.
6. M. Rodríguez-Artacho & M. F. Verdejo. 1999. "Using a High Level Language to Describe and Create Web-based learning scenarios". In *Proceedings of the IEEE Frontiers in Education Conference*.
7. J. Roschelle, C. DiGiano, M.Koutlis, A. Repenning, J. Phillips, N. Jackiw & D. Suthers. 1999. "Developing Educational Software Components". *Computer*, September 99, pp 2-10.
8. D. Suthers. 1999. "Representational bias as guidance for learning interactions: a research agenda". In S.Lajoie & M.Vivet (eds.) *Artificial Intelligence in Education 99*, pp. 121-128. Editors. Amsterdam IOS Press. 1999
9. W. R. van Joolingen. 2000. "Designing for discovery collaborative learning". In *ITS 2000*, pp. 201-211. Springer-Verlag.
10. M.F. Verdejo, B. Barros, M. Rodriguez-Artacho. 2001. A proposal to support the design of experimental learning activities. In P. Dillenbourg, A. Eurelings, K. Hakkarainen (eds.) *Perspectives on Computer-supported Collaborative learning*, pp. 633-640.

Acknowledgement. This work has been funded by Divilab IST-1999-12017 and EA_2C_2 CICYT TIC2001-007