Active Document Description

Technical Report n. 2
LTCS Group, UNED
2003.

Active Document Description

Contents

Summary

This report is a paper version of the information that can be found on the Active Document web page, http://sensei.lsi.uned.es/ActiveDocument. As such, the writing style is different from other deliverables in that it is terser, often follows a question-answer structure. The background of the development work for the Active Document is presented, followed by an architectural overview and the actual composition of the Active Documents. Finally, the way in which the pedagogical design of a scenario is presented to illustrate how the system can be used.

What is it? What is it for?

The Active Document (AD) System provides a computational model and an underlying technological infrastructure that permits the design and development of collaborative learning activities that involve a variety of resources and devices. Emphasis is placed on learning experimental sciences where there is a pressing need for students to improve their learning processes with a better understanding 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.

The purpose of the Active Document System is to:

- Provide an editing tool for the creation and configuration of the different ADs required for the specification of the learning activities.
- Supply the authors (creators or definers) of learning activities with a list of distributed learning objects that consists of a set of references to a variety of tools and resources that can be used for the described activities. These tools and resources are distributed in the sense that they can be installed on the same machine as the user, on the AD Server or any other machine connected to the Internet. The tools are not necessary all developed by the UNED group, it is also possible to integrate tools from other developers.
- Set up an architecture that generates the user environment necessary to carry out the described activities together with the appropriate resources and tools.
- Provide a persistence mechanism that will enable students to store the results of their activities and reuse them in subsequent stages of an experiment.

The AD can be seen from three perspectives:

- Student view: a computer-supported environment that offers a structured scenario with functionality for carrying out individual and collaborative activities. A variety of mediation tools are available, throughout the entire experiment such as structured glossaries, others specifically intended for a particular phase, for instance simulation models.
- Teacher view: the teacher has the role of looking after the work of the different groups, guiding them in the development of the tasks, assessing their results and giving access to the following activities. The Active Document System offers an environment for carrying out these actions, offering different tools to comment upon, controlling and monitoring them
- Designer's view: the Active Document System has an authoring environment that allows the edition of the Active Documents. All this data is edited in XML files. The interface of this editor is divided into two parts, on the left, a tree representing the different elements of an Active Document, and on the right, a form which allows to fill in the values for the element related to that node. Each element needs to be defined and added for the different XML files.

Pedagogical Background

The pedagogical background of the Active Document System is founded in three areas, constructivism, social constructivism, and the activity theory.

Constructivism

- Knowledge is created as a function of the individual's experiences and how the person gives meaning to them.
- An individual mind's world is constructed after this person's interpretation of events.

Social Constructivism

- Peer interaction is decisive for the cognitive development of a person.
- Social-cognitive conflict and coordination are key elements for the intellectual development as they provoke communication between peers.

Activity Theory

- It combines objective, ecological and socio-cultural perspectives of human activities.
- The model is based on the activity, a collective event developed by its participants' actions, which exists within and transforms a material environment.
- Elements of an activity: community, norms, division of work, objective, resources, subject, and outcome.

Technological Background

The underlying technological framework for this work is the Java2 platform, XML, and the client-server applications. Firstly, the Java2 platform has been widely accepted in both academia and industry for the development of many different distributed and architecturally neutral projects. As well as the characteristics (simple, architecture neutral, object oriented, portable, distributed, high performance, interpreted, multithreaded, robust, dynamic, secure) that make it ideal for these types of systems, it also has the advantage of having been used to develop many different APIs and code libraries. Hence, it is usually possible to find something related to an application that can be used as part of new project work without having to redesign/recode everything from scratch.

Secondly, XML, in combination with other related standards and processing technologies, makes it possible to define the content of information interchange

separately from its formatting, making it easy to reuse that content in other applications or for other presentation environments. Most important, XML provides a basic syntax that can be used to share information between different kinds of computers, different applications, and different organizations without needing to pass through many layers of conversion.

Thirdly, the client-server model of distributed processing is well established, and although newer and different models have appeared (such as peer-to-peer), it still survives as the most popular form of building a distributed system, and as such, has been adopted for this work.

Architecture

The ActiveDocument System is a **computational architecture** composed of three parts:

- A metadata editor for the creation and configuration of the different ADs required for the specification of the learning activities.
- A distributed set of tools and resources that can be used in the described activities. A certain degree of interoperability exists between the tools and the system in the sense that results from particular tools can be reused. References to the tools and resources are included in the AD when they are defined.
- An architecture that generates the user environment necessary to carry out the described activities, together with the appropriate resources and tools, and manages interaction between users and the system. Hence, tool invocation, resource access, and result generation are all handled transparently by the system so the user experiences a seamless operation within the interface

The general architecture underlying the Active Document (AD) System and its management is shown in Figure1. The figure represents two different views, or abstractions, of the architecture, which are associated, in order to clarify both its structure and functionality. In the top part of the figure the relations between the XML data structures and the AD System are presented, and in the bottom part, the relations between the functional levels of the system and the data. As can be seen by the dotted lines between the two parts of the figure, the system has four functional levels. The four XML data structures that defines a particular AD scenario together with the results produced by students are labelled as ① in each part, and the generated user interface as ②.

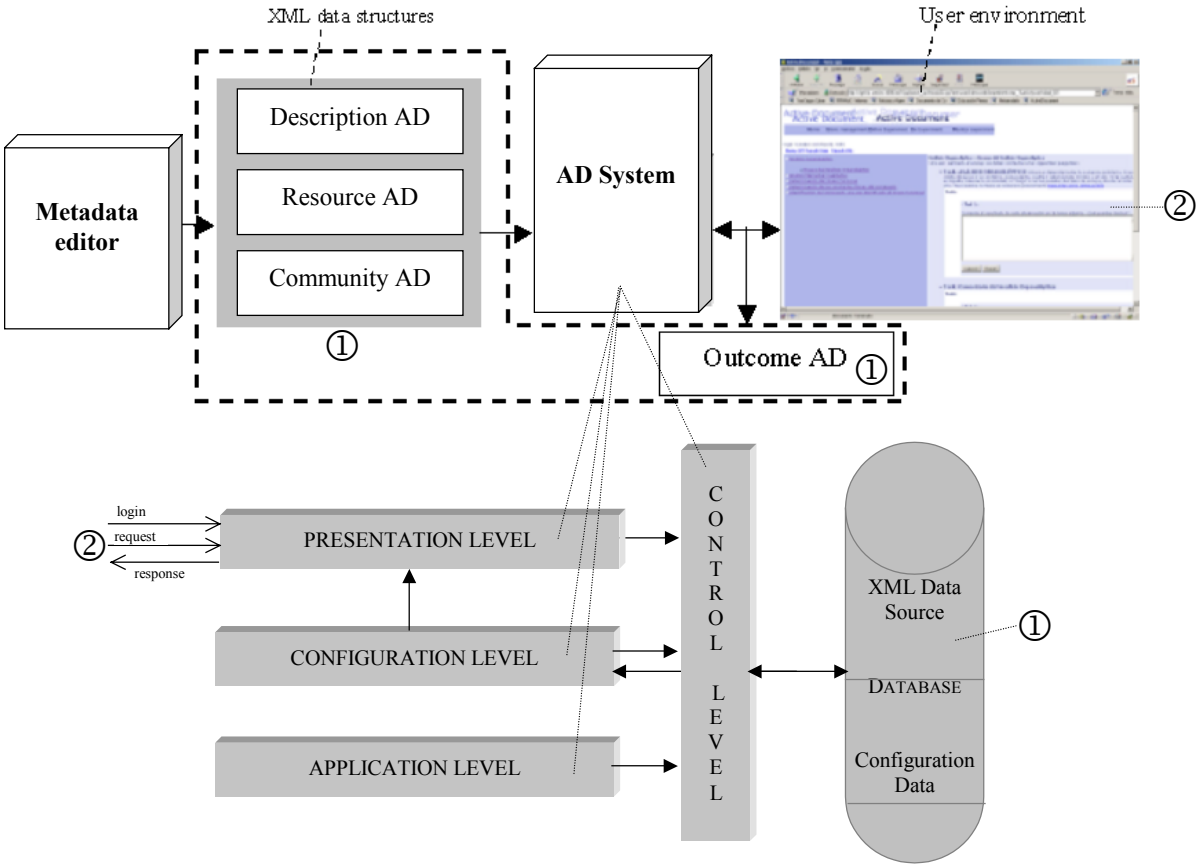

Figure 1. The Architecture of the Active Document (AD) System

The AD system architecture has been developed using **Java-related technology** (specifically components taken from the J2EE platform). Each module shown in figure 1 consists of several components, built upon JSPs, Servlets, JavaBeans, Java classes, and XML data structures. Our goal was to tightly define the functionality of each module, carefully separating presentation glue from business logic and data access. Such an approach provides two fundamental benefits, firstly, the ability to redefine parts of the system without affecting overall functionality (for example, providing wider scenario scope by simply adding some new presentation layer logic to permit Java-enabled hand-held devices or PDAs to connect to the system). Secondly, this approach provides implicit scalability since the components can be re-situated within a fully distributed EJB-based J2EE container. The underlying characteristics that give rise to these two

benefits typify the differences between the first technology prototype of the Active Document architecture and the current one.

The main features of the design are as follows:

- In the context of this architecture, JSP functions well for transformation and presentation as template pages that can easily incorporate dynamic material pulled from session-based JavaBeans and Java classes deeper in the architecture. In the case of the user interface an HTML page is dynamically built in response to the user profile and expressed preferences. Each user that logs into the system has associated with him/her configuration data that instantiates a number of session JavaBeans that handle interactions with the system.
- Users interact with the Active Documents contained in the system via JSPs using Template Method patterns to enforce common design across the entire application and implement common behaviour, such as: page state management, common page processing, common error processing, and the mechanisms for sharing information between pages.
- Tools started from a particular Active Document can take the form of Java applets used within the browser, or stand-alone applications (Java or otherwise) that connect to access Active Documents stored in the database.
- Each user has an XML data structure that can view and edited, which contains his/her preferences. Internally the system will maintain another XML profile with the privileges and data about its interaction with the system.
- Configuration documents will be stored as XML in a database table. Upon login, the user's configuration document is retrieved and passed on to the configuration component for processing. If the user chooses to change their configuration document, the changes take effect immediately.
- The content of the Active Documents come from the XML data stored in the database (any database that supports JDBC). The content is retrieved for presentation using JSPs or for tool manipulation using a combination of SQL and JAXP which generate a stream of SAX events filtered through XSLT style sheets. Style sheets are cached upon first use, which improves performance.
- When accessing Active Document structures and tools, the TM will be consulted, and when the user makes changes to the Outcome Active Document , they will be updated in the database appropriately and the Task Manager will be informed.

This architecture provides a framework for managing the Active Documents and assorted tools. Since all software that makes it up is written in Java, the entire architecture can be mounted on any server where there is a Java2 VM available for the OS (Windows 9x, NT, 2000, Linux, FreeBSD, Solares) and can be used to handle Active Documents and the tools associated with them. Tools that wish to access an Active Document or parts of it or the result data associated with it can do so via simple HTTP/TCP calls.

The AD prototype

The AD system is being developed and used in the context of three projects: Ea2C2, Divilab and Coldex. What follows corresponds to the Version n.1, which has been undertaken in the project EA2C2, available as a demo in the web page of the Active document.

Description of elements that make up an AD scenario for the AD demo version

The concept of an Active Document includes three aspects of the learning process: a description of the activities, a description of the learning communities and the outcome of the work undertaken within the environment. They are specified in XML and are defined by four 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 resources used by in the tasks (referred to as the 'ResourceAD'). (4) 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 Definition

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.

The Community Definition

These components are also expressed in XML in a similar way as an activity. **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 Resource Definition

The **Resources** represent the support for a specific task available to students when they undertake an experiment. For each task, a description of the available resources is given. This description is processed by the AD architecture in combination with the Description AD in order to relate the appropriate tasks and tools. Performing the activities specified in an Active Document may either require or benefit the use of a number of resources, such as external document repositories or different types of modelling tools.

The Outcome Definition

The **Outcome AD** specifies the way in which the results of the tasks performed in the environment are 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.

How to use the AD to create experimental scenarios

Before starting to actually prepare Active Documents, a teacher needs to undertake a pedagogical design that identifies the tasks that make up the experiment that they wish to undertake and the resources that are available for it. In this section, this process will be illustrated by using a chemistry laboratory as a practical example. For a complete case study, see (1)

The experimental context can be broken down into three working phases:

1. Pre-Lab phase:

Students will solve similar problems as those to be proposed in the lab, but in this phase they will get the data from *simulations*. One goal is to improve the students background knowledge, using a problem solving approach. In this phase, students will be asked to solve similar cases as they will do in the physical lab, however physical and chemical manipulations to obtain data will be simulated. In this phase the focus will be on the underlying reasoning process: generate hypothesis, confirm/discard candidates by classification and pattern-matching techniques. In this way students will be aware of the kind of experiments and mental techniques they are going to work with. Students would feel more familiar in the lab; they could pay more attention to the details of the manipulations and will also have more time for thinking and reflection while doing.

Students will work at home, with a *virtual environment.* This environment will offer an integrated space where the student, provided with guidelines, has to solve a set of problems. The guidelines (an active document) will include for each kind of experiment:

- An introduction
- A minimal theoretical background
- Task description

The task description is organized in several subtasks and all the different possible steps to perform them. Not all the tasks should be done for a particular problem. The student has to find his way, selecting the subtasks and the steps to be performed for his case. This document will contain links to *a multimedia glossary* where different kind of domain knowledge (concepts, properties, instruments, procedures...) are defined or explained in different structured ways. In this way we will promote a goal-oriented approach for seeking information. Also some advice would be available about particular choices. This advice would be adaptable, at least using two parameters, one from the teacher to specify whether or not should be available, and another from the previous student behavior, to offer different levels of feedback. For this purpose, the system will build a simple student model.

The work of students will be monitored, with some input and support from the automatic tracking performed by the system. Besides, students would have the possibility of asking

questions to the teacher, related to a particular subtask. These questions and the answers (a kind of FAQ) would be linked dynamically to the glossary. In this way the glossary will evolve with the use of the system, and it is a student contribution to a kind of "organizational memory" for future students.

Lab phase:
A computer would be available in each lab workplace. The support environment would be as before, but the contents and functionality enriched, for instance providing help about how to do a particular manipulation. This kind of advice can be in form of animations. Students will work together in the same place. The *notepad* here will be shared. Collected data about the experiments can have a variety of formats, for instance include pictures of the process and intermediate results.

The role of the assistants will be similar as it is now; hopefully they will not be asked about very basic concepts and could be more available for better supervising.

Post-Lab phase:
Students at home will prepare together their final structured reports about the lab using the *virtual environment*. These reports should include also elaborated explanations, as they have learnt in the Pre-Lab phase, about some questions asked in the guideline document. Assistants will be able to correct (and annotate) these reports as soon as students will decide to publish them. Students will know about the status of this correction process and be able to see the comments as soon as the assistants mark the report as corrected.

The focus of this illustrative example is placed upon the pre-lab session, and the way the pedagogic design is structured. For this, it is necessary to consider the actual objective of the chemistry scenario, which is to identify elements in an organic sample, and highlight the four main stages of the elemental analysis:

1. The production of an alkaline fusion
First, students have to undertake observations (looking at, smelling) about the physical and organoleptic properties of the unknown component. Then they have to write an essay to present a solution for the identification of the sample substance. The kind of procedure to apply is *by reaction* and the result is called *alkaline fusion*. This procedure transforms the original component in an ionic solution of the elements. In this way these elements can be analyzed.

2. The identification of elements
Subsequently, three kind of different experiments are performed on the alkaline fusion to identify the presence of N, S, and halogens (X). In each case there is more than one way to detect an element. The students have to decide which experiments are performed, and the observations for establishing the presence/absence of a particular element.

3. Guessing the functional group
Attending to the relation between solubility and structure there is a well-known classification table including seven different groups. (The table is included in the handout). A first step is to establish to which group the unknown sample belongs and then to identify by further discrimination the functional group. This is structured into a discrimination tree procedure for guiding the students during this phase. Each reasoning step consists on: perform a test, observe the results (clarity of the solution) and establish some conclusions taking into account the data previously obtained. In each group there are different functional groups. Once the group is determined, the functional group should be identified by chemical essays.

4. Confirm the functional group identity
For this task they have to obtain the infrared spectrum of the guessed functional group. With the spectrum, they have to check their hypothesis and give an interpretation for the main bands of the spectrum.

Hence, in the pre-lab phase the students have to undertake the previously detailed four phases using simulations provided by the AD System instead of real laboratory tests. When contemplating the production of a pedagogic model of this phase, it is necessary to know what resources are available. In this case, as well as online glossaries of the chemical substances (which consists of different types of information, for example, photos of each substance, chemical properties, etc.), there is a set of tools that can be used for different purposes (calculation, discussion, reporting, etc.). What follows is a brief summary of these tools that have been put together by the UNED group for the chemistry experiments:

- Collaborative Tools
  - General
    - **Degree**. The Degree tool provides a way to undertake semi-structured textual interaction between students involved in an activity.
    - **Report**. The Report tool permits the students to build a document in a collaborative manner. The resulting document is therefore composition of all the partial contributions of each user along with information about the owner and date of each one.
  - Domain Specific
    - **WBoad2**. A whiteboard that enables students to identify parts of an image and comment on it or on the comments of others. The objective is to provoke the debate between community members.
- Individual Tools
  - General
    - **DrawTool**. Using DrawTool the students can make up simple pictures based on a collection of primitive forms (circle, square, rectangle, etc.).

        - **Notepad**. The user also has access to the standard Windows Notepad (if the user is working on a MS Windows computer) for the composition of individual simple documents.
        - **Calculator**. This the standard Windows Calculator and works with the same restrictions as the Notepad.
        - **LocalWebCamera**. This permits a locally installed Web camera to be used by a user to take photos of his (her) external work.
        - **WebCamera**. The WebCamera tool provides remote access to a Web camera.
        - **ToolTest**. ToolTest is a tool for the construction of classical multiple choice test questions. Instances of this tool are usually included in a task to validate the pedagogical evolution of the student.
        - **Correction**. The Correction tool permits to the teacher correct and comment the results generated by a student.
    - Domain Specific
        - **ChemistryEditor**. The ChemistryEditor is a tool that depicts all type of organic formulas. To this end, it provides some primitive Carbon based constructions that can be linked together to form a larger formula.
        - **VisorSimulations**. The VisorSimulations provide some simulations for different formats (text description, real photos, pictures, video, and so on…)
        - **Thermometer**. The thermometer is another simulation tool whose behavior is as a standard laboratory thermometer.
        - **SelectQuality**. The SelectQuality tool provides a mean of choose the quality characteristics of certain chemical substance. This tool is used by the chemistry student in tasks related to qualitative analysis.

Phase four of the previous organic chemistry analysis will be taken as an example of the production of the AD learning description. Hence, the teacher can see that this phase can be broken down into three tasks:

1. Selection of an appropriate spectrum: the students access an online glossary of spectrum and select a possible one for use in the next step.
2. Collaborative discussion about the spectrum: the spectrum selected in step 1 is presented inside the tool, and as can be seen in the figure 3, the students work together using a collaborative whiteboard in small groups to analyze and label peaks on the spectrum selected in the previous phase, in order to be able to identify the substance based upon the results of the laboratory results.

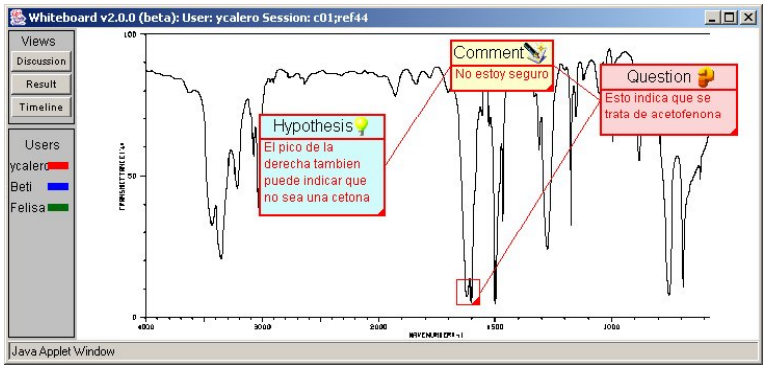

Figure 3. Use of collaborative whiteboard to identify peaks in a spectrum

3. Once the students have reached a "collective conclusion" in their small groups about the substance they are studying, they have to answer some multiple choice questions (individually) to show that they have achieved the conclusion for the right reasons, for example:

1. If an alkaline solution is colored, what it means?
    a) There are colored ions and the product is an ester
    b) There is initial product on dissolution
    c) Sodium are dirty and has the solution has been stained
    d) The alkaline fusion has been done badly

2. Adding ferrous sulphate to the solution appears a dark green precipitate…
    a) Always
    b) Only when sulphur is present in the solution
    c) When the initial product is an amine
    d) At rarely times

Once the design has be detailed, i.e., a set of tasks and roles have been identified with associated resources (tools and multimedia data), the Active Documents need to be specified for them that when interpreted by the Active Document System will give rise to the virtual experimental environment, as detailed in (2). An example of the editor interface for the AD specification process can be seen in figure 4.

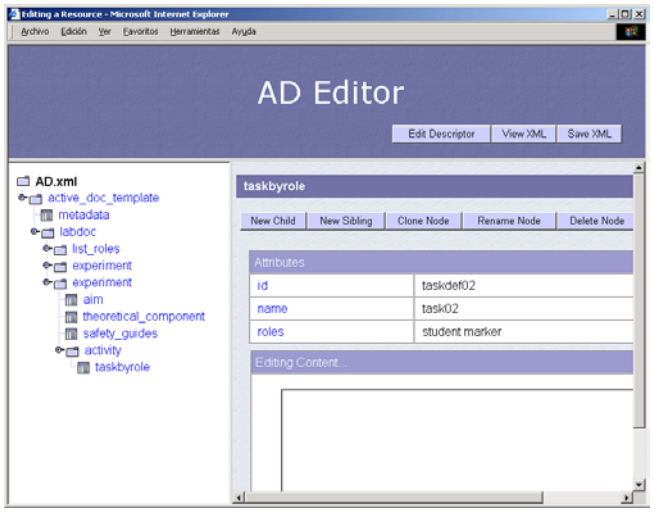

Figure 4. AD Editor

As can be seen in figure 4, the AD Editor assists a teacher in the process of defining a learning environment and its associated resources in terms of ADs. Particularly definitions of activities, experiments, task, roles, and assignments of responsibilities to roles can be undertaken with this tool. As can be seen in the figure, the AD Editor is composed of frames. On the left the global nested structure of a particular Active Document can be inspected, and on the right, different user interfaces are presented depending on the structures that need to be modified. Several buttons help the user create, delete or modify the nodes in the edited document. Details are provided in the Active Document Manual (2).

How the systems works

The interface

The Active Document system permits the definition of roles. Each role configures a group of users and offers a variety of functions depending on the type of available tasks, user types and tools. Three roles can be identified, that of student, teacher, and experiment designer. The latter has been illustrated in the previous section. An example of the student interface for the previously specified organic chemistry scenario can be seen in the figure 5 and the teacher interface in the figure 7.

The structure of the interface in figure 5 can be seen to be the following. Firstly, there is a menu bar at the top of the page that presents the student with general environmental properties that can be changed. Secondly, on the left, there is a hierarchical menu that presents the experimental task structure that the student has available to explore. Thirdly

and finally, on the right, there is an area where the related task appear together with links to the relevant tools and resources.

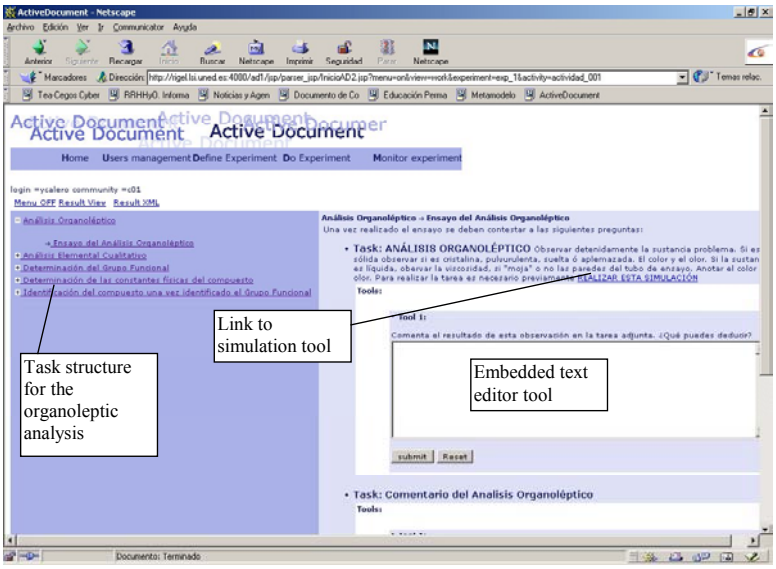

Figure 5. Student Active Document Interface

It should be noted that 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. 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.

In figure 6 the way in which the Outcome AD XML data structure 'grows' as the students generates a result can be seen. At the top of the figure the relation between the state of the Outcome AD can be seen when the student logs in for the first time, subsequently, when the student uses a tool and generates a result, the new Outcome AD structure can be seen. Although it is not possible to see the details of the change, due to the image size, the qualitative size change can be appreciated.

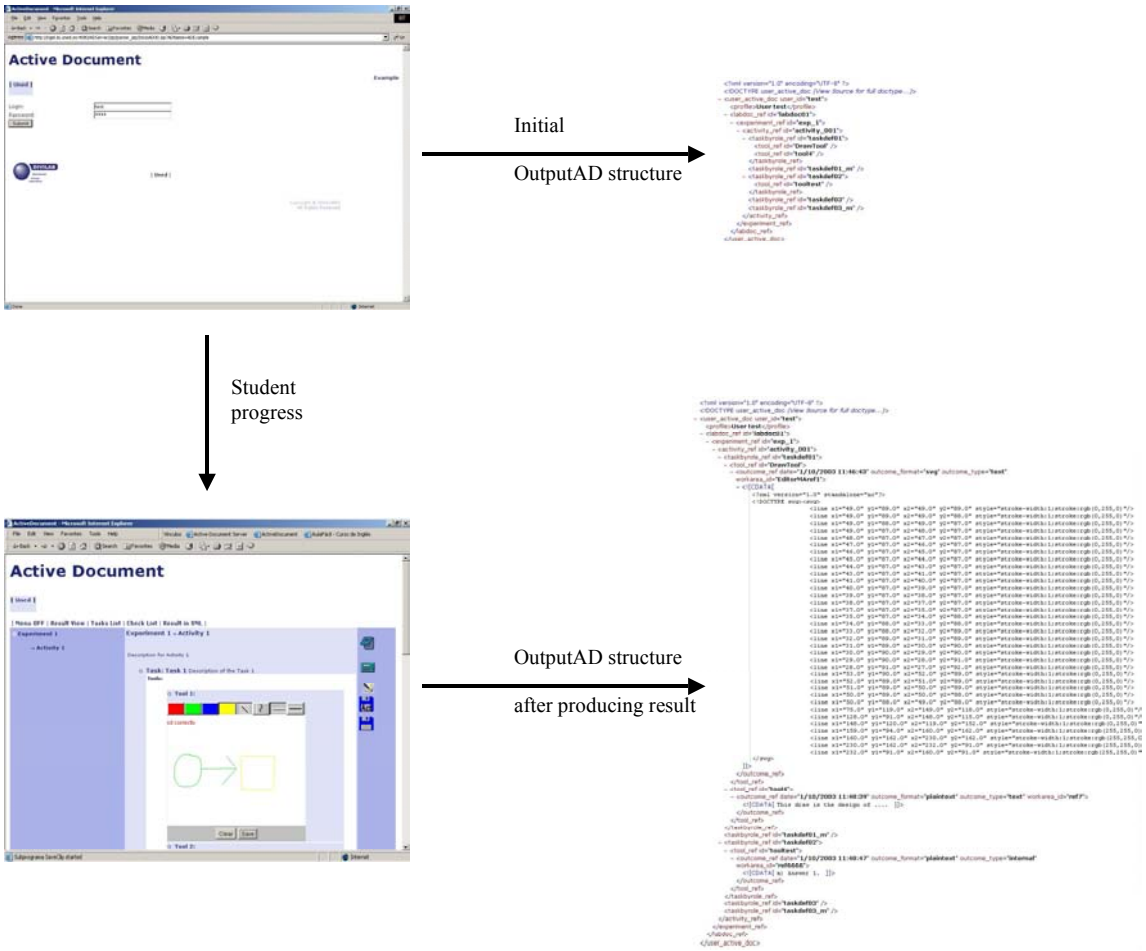

Figure 6. OutputAD growth

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.

In figure 7, the teacher interface has a similar top menu structure together with the options to see the full student list and the other experiments that are currently in progress for a particular student. In the figure the correction tool interface can be seen where the teacher has the option to comment upon the results produced by a particular student using two different tools.

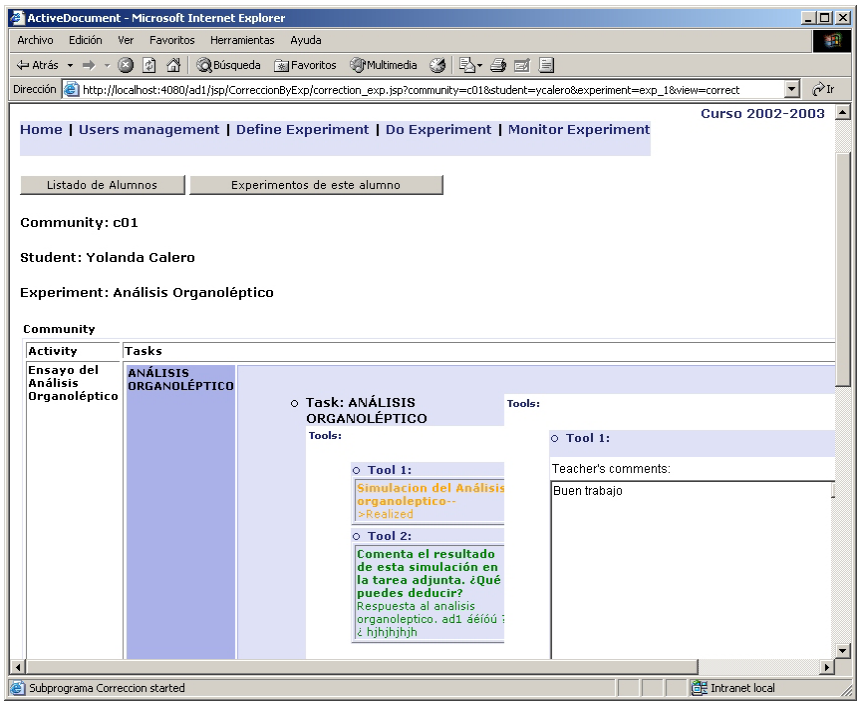

Figure 7. Teacher interface to the chemistry Active Document system example

For a more detailed description, see the user manual (2).

References

(1)- A scenario testbed for a local experiment site in Chemistry, Technical Report n.1, LTCS Group, UNED 2003

(2)- Active Document Manual, Technical Report n. 3, LTCS Group, UNED 2003