AIED 05 WORKSHOP 9

Amsterdam, 18-22 July, 2005

Third International Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia

WORKSHOP

12th International Conference on Artificial Intelligence in Education, Amsterdam, the Netherlands

A[3]EH: The 3rd workshop of Authoring of Adaptive and Adaptable Educational Hypermedia

Alexandra Cristea[1], Rosa Carro[2] and Franca Garzotto[3]
[1]*Eindhoven University of Technology, The Netherlands*
[2]*University Autonoma of Madrid, Spain*
[3]*Politecnico di Milano, Italy*

Abstract. The A[3]EH follows a successful series of workshops on Adaptive and Adaptable Educational Hypermedia. This workshop focuses on models, design and authoring of AEH, on assessment of AEH, conversion between AEH and evaluation of AEH. The workshop has paper presentations, poster session and panel discussions. As you see, we have a full agenda, so don't hesitate to join us!

Introduction

This workshop follows a successful series of workshops on the same topic. The current workshop focuses on the issues of design, implementation and evaluation of general Adaptive and Adaptable (Educational) Hypermedia, with special emphasis on the connection to user modelling and pedagogy. Authoring of Adaptive Hypermedia has been long considered as secondary to adaptive hypermedia delivery. This task is not trivial at all. There exist some approaches to help authors to build adaptive-hypermedia-based systems, yet there is a strong need of high-level approaches, formalisms and tools that support and facilitate the description of reusable adaptive websites. Only recently have we noticed a shift in interest, as it became clearer that the implementation-oriented approach would forever keep adaptive hypermedia away from the 'layman' author. The creator of adaptive hypermedia cannot be expected to know all facets of this process, but can be reasonably trusted to be an expert in one of them. It is therefore necessary to research and establish the components of an adaptive hypermedia system from an authoring perspective, catering for the different author personas that are required. This type of research has proven to lead to a modular view on the adaptive hypermedia. One of these modules, which is most frequently used, is the User Model, also called Learner Model in the Educational field (or Student Model in ITS). Less frequent, but also emerging as an important module is the Pedagogical Model (this model has also different names in different implementations, too various to name here). It becomes more and more clear that for Adaptive Educational Hypermedia it is necessary to consider not only the learner's characteristics, but also the pedagogical knowledge to deal with these characteristics. This workshop will cover all aspects of the authoring process of adaptive educational hypermedia, from design to evaluation, with special attention to Learner and Pedagogical models. Therefore, important issues to discuss are, among others:

* What are the main characteristics (that should be) modelled of learners?
* How can the pedagogical knowledge be formulated in a reusable manner?
* How to support pedagogic scenarios?
* How can we consider user cognitive styles in adaptive hypermedia?
* How can we consider user learning styles in adaptive hypermedia?

* Are there any recurring patterns that can be detected in the authoring process generally speaking, and in the authoring of user or pedagogic model in particular?

The workshop will also lead to a better understanding and cross-dissemination of user-specific patterns extracted from existing design and authoring processes in AH, especially focused around user modelling and pedagogic modelling.

The workshop aims to attract the interest of the related research communities to the important issues of design and authoring, with special focus on user and pedagogic models in adaptive hypermedia; to discuss the current state of the art in this field; and to identify new challenges in the field.

Moreover, the workshop should be seen as a platform that enables the cooperation and exchange of information between European and non-European projects.

Major Themes

* Design patterns for adaptive educational hypermedia
* Authoring user models for adaptive/adaptable educational hypermedia
* Authoring pedagogic models for adaptive/adaptable educational hypermedia
* Generic authoring for adaptive/adaptable educational hypermedia
* Authoring patterns for user models and pedagogic models in adaptive/adaptable educational hypermedia
* Authoring Tools for user models and pedagogic models in adaptive/adaptable educational hypermedia
* Generic authoring tools in adaptive/adaptable educational hypermedia
* Reusable user models and pedagogic models
* Connecting adaptive educational hypermedia with cognitive/learning styles
* Evaluation of authoring tools for adaptive educational hypermedia
* Evaluation of adaptive educational hypermedia design patterns
* Evaluation of adaptive educational hypermedia authoring patterns

The workshop will consist of three major parts:

- full paper paper presentations,
- poster session and
- panel discussions.

The FULL PAPERS papers at the workshop are grouped according to the theme they touch, as follows:

1. THEME: models, design and authoring: [1], [2]
2. THEME: adaptive assessment authoring: [3], [4]
3. THEME: authoring and converting: [5]
4. THEME: authoring evaluation: [6], [7]

The workshop also has a number of POSTER PAPERS, of work-in-progress, projects and idea-work on authoring of AEH [8],[9],[10],[11], so we hope for a lot of discussions.

The PANEL will be constituted of a number of invited researchers and well-known specialists on the field of AEH and A^3EH, which will discuss a number of topics from the above list to be announced at: http://wwwis.win.tue.nl/~acristea/AAAEH05/AIED2005.htm

Last but not least, the workshop chairs wish to gratefully thank the Program Committee, for their graceful willingness to sacrifice some of their busy time to not only select relevant papers and posters for this workshop, but also to give valuable advice for improving the papers and panel themes:

Helen Ashman, University of Nottingham (UK)
Paul De Bra, Technische Universiteit Eindhoven (Netherlands)
Tim Brailsford, University of Nottingham (UK)
Licia Calvi, University of Pavia (Italy)
Hugh Davis, Southampton University (UK)
Serge Garlatti, GET-ENST Bretagne (France)
Nicola Henze, University of Hannover (Germany)
Judy Kay, University of Sydney (Australia)
Kinshuk, Massey University (New Zealand)
Wolfgang Nejdl, University of Hannover (Germany)
Toshio Okamoto, University of Electro-Communications (Japan)
Helen Pain, University of Edinburgh (UK)
Simos Retalis, University of Piraeus (Greece)
Pilar Rodríguez, Universidad Autonoma de Madrid (Spain)
Johann Schlichter, Technical University of Munich (Germany)
Marcus Specht, Fraunhofer Institute for Applied Information Technology (Germany)
Daniel Schwabe, PUC - Rio de Janeiro (Brazil)
Carlo Strapparava, IRST (Italy)
Lorna Uden, Staffordshire University (UK)
Vincent Wade, Trinity College (Ireland)
Gerhard Weber, Freiburg University of Education (Germany)
Wolfgang Woerndl, Technical University of Munich (Germany)

Hope to seeing you all at the workshop!

The Workshop Co-chairs
Dr. Alexandra Cristea - Eindhoven University of Technology, The Netherlands
Dr. Rosa M. Carro - University Autonoma of Madrid, Spain
Prof. Dr. Franca Garzotto - Politecnico di Milano, Italy

References

[1] Manuel Freire and Pilar Rodriguez, Comparing Graphs and Trees for Adaptive Hypermedia Authoring
[2] Harrie Passier & Johan Jeuring, Using Schema Analysis for Feedback in Authoring Tools for Learning Environments
[3] Symeon Retalis, Petros Lalos, Creating personalised quizzes both to the learner and to the access device characteristics: the Case of CosyQTI
[4] Paul Cristea, Rodica Tuduce, Automatic Generation of Exercises for Self-testing in Adaptive E-learning Systems: Exercises on AC Circuits
[5] Alexandra Cristea, David Smits and Paul de Bra, Writing MOT, Reading AHA! - converting between an authoring and a delivery system for adaptive educational hypermedia -
[6] Declan Dagger, Vincent Wade, Evaluation of Adaptive Course Construction Toolkit (ACCT)
[7] Alexandra Cristea, Craig Stewart, Tim Brailsford and Paul Cristea, Evaluation of Interoperability of Adaptive Hypermedia Systems: testing the MOT to WHURLE conversion in a classroom setting
[8] Valentin Cristea, Stefan Trausan-Matu, Authoring and delivering adaptable learning objects in SINTEC
[9] Lichao Li, Judy Kay, The Cost of Authoring with a Knowledge Layer
[10] . Ferrer-Roca, A. Figueredo, K. Franco, A. Diaz-Cardama, Telemedicine ontology for AIED
[11] Vicente Arturo Romero Zaldivar, Jon Ander Eliorriaga Arandia, Mateo Jerónimo Lezcano Brito, New Features of the OPScript Language

Comparing Graphs and Trees for Adaptive Hypermedia Authoring

An experimental setup using the new version of the TANGOW *Adaptive Hypermedia System*

Manuel Freire [a,1] and Pilar Rodríguez [a]

[a] *Escuela Politécnica Superior, Universidad Autónoma de Madrid, Spain*

Abstract. A key decision in designing authoring tools for Adaptive Hypermedia is the type of representation to use when authoring the relationships among contents. This paper describes an experiment that compares two representations, one based on trees, another using clustered graphs for the same task. An effort has been made to isolate all other aspects of authoring from the comparison; the choice of trees vs. graphs should be the only difference between both editors, which have exactly the same capabilities and goals: to author courses for the new version of the adaptive educational hypermedia system TANGOW. Experimental results for a small survey using the described setup are included.

Keywords. Adaptive Hypermedia, Visualization, Focus+Context, TANGOW

1. Introduction

Adaptive Hypermedia (AH) systems are based on the idea of adapting site contents to users, instead of using a one-size-fits all approach. They maintain some sort of information about the user, and the user's goals and previous actions (the *user model*[4]), and try to present content that will be relevant to the current task for that particular user by performing different types of adaptation on a (structured) content repository; the relationships among the elements in the content repository define a structure, which is often termed *content model*, or when a repository refers to a specific domain, *domain model*. In an educational context, a content repository usually consists of multiple individual courses (although this distinction can be blurred if re-usability is inserted into the equation, and several courses share contents).

The structure of an adaptive hypermedia course is therefore invariably more complex than that of static one: adaptation is performed by following certain procedures that relate a given user model with the domain model, and the domain model must include some sort of extra information that guides the adaptation process in this task. This information is usually added as meta-data, labelling content units or subunits with special tags and establishing relationships among them. An adaptation engine later makes use of this information and a user

[1]Correspondence to: Manuel Freire, Av. Tomás y Valiente 11, ES-28049 Madrid, Spain.
Tel.: +34 91 4972267; Fax: +34 91 4972235; E-mail: manuel.freire@uam.es

model to present a seamless course tailored to this user's perceived goals and needs. Some relations among course units usually have much greater priority than others. For instance, the *part-of* relationship typically defines the high-level layout of a course, a layout that can then be altered before presentation by the adaptation engine (maybe *part-of-if* would be a better description), and then further refined at lower levels of locality to provide more fine-grained types of adaptation.

We present an experiment comparing different structural representations in two authoring tools developed for the WOTAN adaptive hypermedia course system. WOTAN is a new version of TANGOW[7,6], an educational AH system developed at the Universidad Autónoma de Madrid by the author's research group. In section 2, the basics of the adaptation mechanism used in WOTAN are described. The editors are described in section 3, with emphasis on the user's interface to the structure rather than the actual details involved with course update, user model expression syntax, etc. The tree-based editor is introduced in 3.1, and the graph-based editor is discussed next. In section 4, we describe an experimental setup designed to compare both approaches in different authoring tasks, and discuss the results gathered in an experiment with a small number of users.

Finally, section 5 provides concluding remarks and lines of future work, and insists on the generality of the representation problem, which we believe to be common to any adaptive hypermedia system that performs global adaptation on a closed, tightly coupled content repository.

2. WOTAN, a new version of TANGOW

The TANGOW adaptive system has been updated in the last year. The new version, WOTAN, features better integration among the administration, presentation and authoring tools, and uses XML files instead of database tables for course data, user model, and system configuration information. A new exercise subsystem has also been introduced. The remainder of this section provides an overview of the system architecture, the course model, and the user model, as a basis for the presentation of the authoring tools.

The renovation of TANGOW has been partly inspired by other adaptive hypermedia course systems, particularly by work on the AHA system [9,15] and adaptive hypermedia reference models such as [3], although in TANGOW there is no proper separation between the "teaching model" and the "content model".

2.1. Architecture

WOTAN is implemented as a set of modules (servlets inside a Java-based web application) that share session information. The `exer` module provides a web-based interface that allows authors to create and test different types of exercises, which can later be included in courses; the `admin` module implements access control and provides user and course management. Authors can upload and download courses from the system through this interface. Finally, the `pres` module is in charge of presenting courses to users, performing adaptation as needed. Currently, it is designed to closely mimic the behavior previous TANGOW interface, but the adaptation module is easily replaceable for new adaptation requirements.

Course authoring occurs client-side, and completely off-line; this allows much greater interactivity within the authoring tool, and richer representations of the course structure.

2.2. User model, course model, and adaptation

Fig. 1 illustrates the contents of a user session. Sessions include a user model with global student data (such as preferred language, learning styles[13], etc.), and a course-specific "overlay" for each course the user is enrolled in. All model information is encoded in attribute-value pairs, where the attributes of course overlays mark a location in the course model where the attribute is of relevance. If the user is currently performing a course, the current course model (as annotated by the corresponding overlay) will also be stored in the session; when the session is finished, it will be serialized back into the user model's course overlay for that particular course.

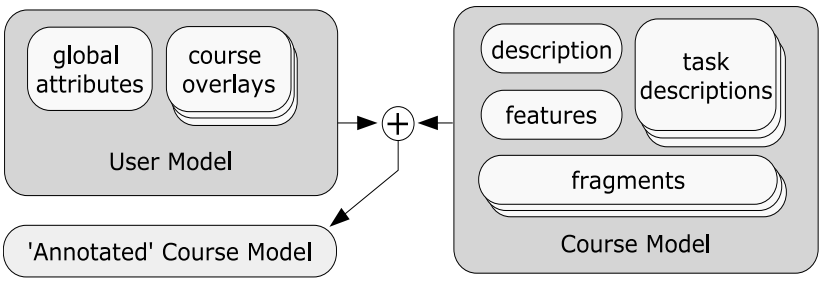


Figure 1. Construction of an annotated course model in WOTAN, the new version of TANGOW.

Adaptation data is contained in the course description, an XML document which describes a course in terms of tasks, rules, fragments, versions, and adaptation features.

Tasks are blocks of content, which may contain both other blocks and/or content fragments. A course always defines a *main task*, which acts as the entry point to the whole course. Containment is not exclusive: several tasks may refer, either directly or indirectly, to the same subtasks or fragments; but no task may include an ancestor as a subtask. Therefore, in terms of inclusion, TANGOW courses are directed, acyclical graphs (that is, graphs where no cycles can be created by following edges).

The decomposition of a task in subtasks is governed by *rules*. Each rule has an activation condition (a condition on the user model that must be met in order for the rule to be considered active), a list of subtasks, and a criteria for sequencing the subtasks (deciding in which order they will be recommended) and evaluating successful task completion. When several rules are simultaneously active for a given task, only one may be triggered (the student has a choice here). In order to simplify navigation, at most one rule for each task may be triggered. This means that, in any given moment, the path that a student has taken throughout a course will be a well-formed tree. The task + rule mechanism implements navigational adaptation in TANGOW.

Fragments are small snippets of html, images, applets, etc. A fragment may have several versions, and each version will have an associated expression that evaluates its suitability to a given user model. The most suitable version of each fragment in a task is selected when building the html pages that will represent that task to a given user, implementing canned-text adaptation.

Features describe additional aspects to be added to the user model for this particular course. For instance, in a course on microprocessor design, it may be relevant to know the user's familiarity with boolean algebra. A feature *boolean-algebra* could be added as a "feature" to be considered in the course. Features are initialized when a course is first visited, using an automatically generated questionnaire, and the values filled in are later

available for adaptation purposes. In this case, the feature's value would be accessible as *map.course.boolean-algebra*.

Additionally, certain built-in user model characteristics (such as preferred language) are available for all courses, and the whole domain model itself (including full history) is available for adaptation. For instance, `map.task.booleanExercises.grade >= .5` (representing the final grade in task that includes a set of exercises about boolean algebra) may be better than a user-provided "feature" value when deciding whether the user really has an idea about boolean algebra. Expressions using values from the annotated user model are used in rule activation, fragment version selection, task finalization conditions, and parameter propagation (that is, pervasively) around the system, and have been implemented using the well-known Java Expression Parser package[1].

3. Editors

Authoring tools for educational AH must provide an interface suitable for authors with good knowledge of their respective domains, but not necessarily intent on an in-depth study of the system's internal workings. The interface should present an overview of the course structure, and allow direct modification of this structure as the course evolves. As described in [8], high-quality AH courses are more likely to evolve than to be completely designed from the ground off, so the interface should allow for both course creation and later maintenance.

Two obvious candidates for structural representation are trees and graphs. Trees provide a familiar top-down model of course organization, as found in the Table of Contents of many reference materials. Most AH systems use trees in their authoring tools, either exclusively or in addition to other representations (for instance, [9,5,7]). The only drawback of using trees is that non-hierarchical relationships are hard to represent inside a tree. These are, however, commonly represented by directed graphs (which are a generalization of trees where individual nodes can have more than one 'ancestor'). Graph-based tools for adaptive hypermedia editing are becoming more common, as in AHA 2.0's graph-based editor[9] and TANGOW's ATLAS editor [12]. Trees tend to be very uniform interfaces, with standard expand and collapse operations, and well-understood semantics. The extra liberty provided by graph representations makes graph-based systems much more heterogeneous.

We present two authoring tools for WOTAN courses, capable of exactly the same tasks, and which differ only in their representation of courses. In fact, both are stand-alone Java applications that share most of their code-base. Course authors are expected to download an existing course from the system as a compressed file (or create a new one with the chosen editor), edit the course locally, and upload it again to the system.

The editors also allow authors to import previous database-bound courses into the new format, and assist a user in editing and adding fragment versions. However, this shall be of no concern in the following subsections, which deal almost exclusively with the structural interface.

3.1. The tree editor

This editor, depicted in Fig. 2, is based on the interface found in a previous web-based editing tool for TANGOW. The component represents a course structure as an expandable tree, where nodes can represent tasks, rules, fragments and fragment versions. Color-coded icons are used to denote each of these elements. Fragment versions are the only real leaves.

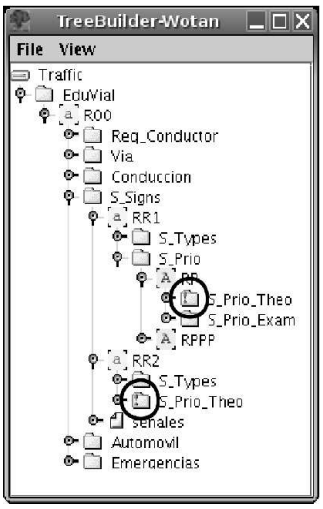


Figure 2. The tree interface, displaying a part of the "Traffic" course. The circles mark a task accessible through several paths.

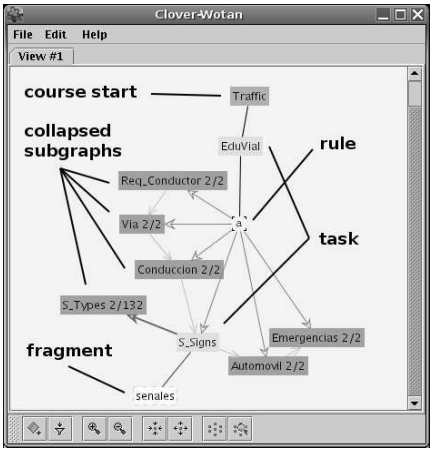


Figure 3. The graph interface, displaying the same course. Annotations have been superimposed in Arial typeface.

The tree can be expanded and collapsed as usual, contains no internal state (so that, should the course structure change due to editing, a simple revalidation will bring the representation up-to-date), and can be filtered according to a "user model" filter.

In Fig. 2, an example illustrates the main drawback of using trees for course structure representation: if a single task is reachable through different routes, it is marked with an exclamation mark (in the example, "s_prio_theo", hightlighted by an arrow). It is difficult to represent these non-hierarchical relationships in an inherently hierarchical representation such as a tree.

3.2. The graph-based editor

The graph-based editor is significantly more complex. Its initial aim was to solve the issue of non-hierarchical relationships encountered with the tree-based approach. However, the use of graphs introduces new problems: graphs are significantly less efficient in their use of screen space, require non-trivial layout to be performed (either automatically or by the user), and result incomprehensible beyond a certain size.

This has motivated the adoption of several information visualization techniques to reduce the complexity of the graphs being presented, while maintaining desirable characteristics that make them comparable to trees when trees alone would suffice. In the first place, automatic layout has been preferred to manual layout, as we understand that manual layout places too much of a burden on users, and automatic layout is both feasible and pleasant as long as the node count is kept low. This is achieved via clustering of related nodes. Navigation throughout the graph is performed by clicking on nodes, which will convert the selected node into the current 'Point of Interest', and expand nearby 'interesting' nodes and collapse distant 'non-relevant' ones. The default behavior of this fisheye-lens[11] (or focus+context) approach can be modified by either freezing certain nodes (so that neither their position nor their collapsed-uncollapsed status will be altered by visualization changes) or modification of the focus variables: number of nodes to be shown, and size of the neighborhood to expand.

Fig. 3 is a screen-shot of the graph interface. Color-coding and shapes are used to distinguish different node and edge types. The edge between "S_Signs" and "S_Types" is a

"cluster edge", that is, contains several edges (see the tree representation of the same course, in Fig. 2, to identify the edges involved).

Animation of transitions is used in order to preserve the user's mental map of the course during navigation, as every change in viewpoint or filtering implies another layout of affected nodes, and suddenly changing node positions tend to be annoying.

The graph-based editor is built on top of CLOVER [10] (Cluster Oriented Visualization EnVironment), a graph visualization framework that uses the JGraph[2] component. CLOVER is not specific to TANGOW courses, or indeed adaptive hypermedia; it has been used for graph visualization in other application domains, such as the reusable content creation system Targeteam[14,10].

4. Experimental Setup

As both editors share the same editing capabilities, the goal of the experiment is to measure author speed, accuracy and satisfaction with each of the editors. First, experiment participants receive a brief primer to WOTAN course structure and an introduction to each of the tools and a description of the tasks to be preformed. They are also presented with a printed page with the symbols used in course representation and a list of requested tasks. Finally, the participant is asked to perform the tasks on an example course, in order, first using one editor and then the other. The order of tool selection is changed for each participant, so that half of the participants start with the tree-based tools, and the other half starts with the graph-based one. As the same tasks are performed twice, once for each tool, by users without previous experience with the system (or AH in general), we expected to find the speed and accuracy of the second run to be better than those of the first run.

4.1. Tasks

All tasks are performed on the example course depicted in the previous tool screenshots. The tasks are designed to be simple but representative of a typical editing session, and are presented in order of increasing difficulty. For each task, the time required to complete it successfully (as determined by the experimenter upon request) is recorded. After all tasks for both tools have been completed, the user is requested to fill in a small "difficulty survey" assessing the difficulty of each task/tool combination.

- Task 1: find and change the name of a given fragment node, identified by a path that leads to it. Subjects will have to navigate using the corresponding tool, which is the only problem presented by this task.
- Task 2: locate all paths leading to a task node. One of the paths is provided in the statement. This task is representative of a typical problem in AH courses, and we expected the graph representation to perform better than the tree representation.
- Task 3: add a rule that will be active only for users with a specific profile (rule location and profile-matching information are provided in the statement), insert a task into this rule, and add a fragment to the inserted task. This tests basic course growth.

4.2. Results

Experiment results can be found in figures Figs. 4 and 5. Only 8 participants were involved, divided into two groups. Charts on the top row represent time in seconds required to perform

each task. In each chart, there are 6 blocks of columns; the first three columns represent the time required to perform each of the 3 tasks with each of the editors, in order, from left to right. Charts in the second row display perceived difficulty of each task with each editor (filled in after performing all tasks with all editors), adjusted so that the least difficult task for each user begins at "difficulty level" 1.

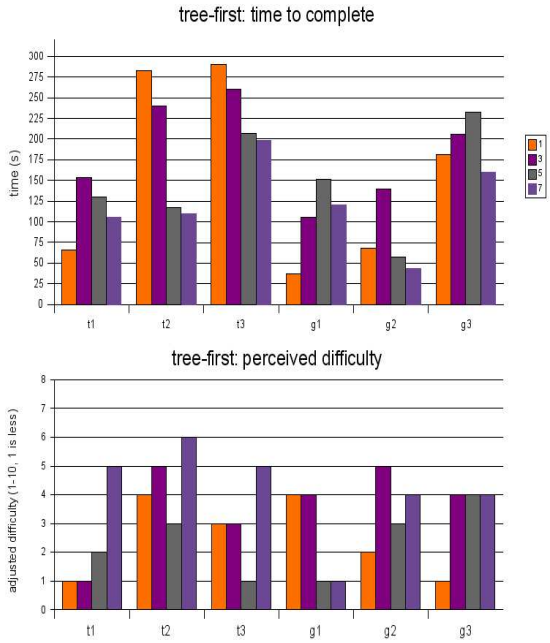


Figure 4. Participants that started with the tree-based editor. Tasks on the *left* were performed first.

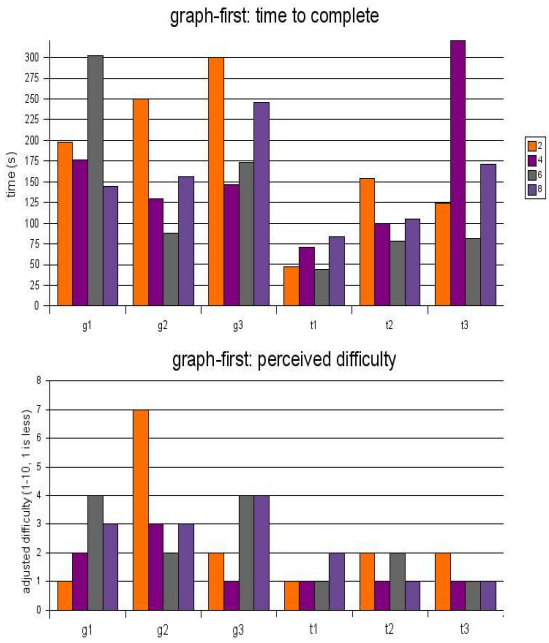


Figure 5. Charts for participants that started out with the graph-based editor.

Statistical analysis is not meaningful with such a small sample, but certain trends are visible. For instance, task times are much shorter in the second run than in the first run, regardless of the editor that was used first. This suggests that familiarity with the system's concepts and the course structure is harder to master than interaction with the editors (users had only received a very basic introduction to both).

Task times with each group's first tool are very similar, although Task 1 seems to take longer when performed for the first time with the graph editor, probably because graph navigation is more complex than tree navigation. As expected, Task 2 is performed slightly faster when using the graph editor. Better initial training should have lowered times much more. Perceived difficulty is also usually lower for Task 2 when using the graph-based editor. It is also interesting that users rate tasks as more difficult when they have started out with the tree editor. This suggests that those using the graph-based editor had a better understanding of the course structure on their second run, and found tasks easier to accomplish.

In general, despite the differences in interface, both tools are roughly equal for non-experienced users when performing simple tasks. The graph tool is perceived as harder to use and takes longer to get used to than the tree tool, but seems to provide better understanding of course layout. Familiarity with the system is gained rapidly, but still accounts for most of the difference between both runs.

5. Concluding Remarks and Future Work

Adaptive Educational Hypermedia authoring tools must choose a suitable interface to present content structure and relationships. The two main candidates are tree-based and graph-based representations. An experimental setup to compare two approaches to course structure representation, one using trees and the other using graphs, is presented. The main feature is that both tools share exactly the same capabilities, differing only in course representation. Although the authoring tools under comparison are both designed with the WOTAN system in mind, we believe that the ideas behind this experiment are applicable to a broad range of adaptive hypermedia systems.

Experimental results suggest that graph-based representations, although harder to master, are better suited to navigation and orientation in AH structures than tree-based ones. However, the evidence is inconclusive: the sample is small, and familiarity with the system and the requested tasks should be factored out of the experiment to provide a more meaningful comparison. A follow-up experiment with a larger sample size is planned for the near future, including a longer introduction to the WOTAN system and both editors, and a wider array of tasks. Interface glitches discovered during this first experiment will be fixed in time for the follow-up.

Acknowledgments

This work has been sponsored by the Spanish Ministry of Science and Technology (MCyT) with project code TIN2004-03140.

References

[1] Java expression parser. http://www.singularsys.com/jep/.

[2] Jgraph java graph visualization component. http://www.jgraph.com/.

[3] Paul De Bra, Geert-Jan Houben, and Hongjing Wu. AHAM: A dexter-based reference model for adaptive hypermedia. In *UK Conference on Hypertext*, pages 147–156, 1999.

[4] P Brusilovsky. Adaptive Hypermedia. In *User Modeling and User-Adapted Interaction*, volume 11, pages 87–110. Kluwer Academic Publishers, The Netherlands, 2001.

[5] Peter Brusilovsky and Gerhard Weber. Elm-art: An adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education*, 12:351–384, 2001.

[6] R. M. Carro, E. Pulido, and P. Rodriguez. Dynamic generation of adaptive Internet-based courses. *Journal of Network and Computer Applications*, 22(4):249–257, October 1999.

[7] R. M. Carro, E. Pulido, and P. Rodriguez. TANGOW: a Model for Internet Based Learning. *International Journal on Continuing Education and Life-Long Learning*, 11(1–2), 2001.

[8] Alexandra Cristea and Lora Aroyo. Adaptive authoring of adaptive educational hypermedia. *Lecture Notes in Computer Science*, 2347:122–132, 2002.

[9] P. De Bra, A. Aerts, D. Smits, and N. Stash. AHA! Version 2.0, More Adaptation Flexibility for Authors. In *Proceedings of the AACE ELearn'2002 conference*, pages 240–246, October 2002.

[10] Manuel Freire and Pilar Rodriguez. A graph-based interface to complex hypermedia structure visualization. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, pages 163–166, New York, NY, USA, 2004. ACM Press.

[11] G. W. Furnas. Generalized fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 16–23. ACM Press, 1986.

[12] J. A. Macías and P. Castells. Interactive Design of Adaptive Courses. In M. Ortega and J. Bravo, editors, *Computers and Education - towards an Interconnected Society*. Kluwer Academic Publishers, Dordrecht (The Netherlands), 2001.

[13] Pedro Paredes and Pilar Rodriguez. Considering sensitive-intuitive dimension to exposition-exemplification in adaptive sequencing. *Lecture Notes in Computer Science*, 2347:556–559, 2002.

[14] J. Schlichter and G. Teege. Web-based information management for university education. In Jari Multisilta and Jari-Pekka Niemi, editors, *Proceedings of LETTeT 98 and MaTILDA 98 joint conference*, pages 99–106, May 1998.

[15] N. Stash and P. De Bra. Building Adaptive Presentations with AHA! 2.0. In *Proceedings of the PEG Conference*, July 2003.

Using Schema Analysis for Feedback in Authoring Tools for Learning Environments

Harrie PASSIER & Johan JEURING
Faculty of Computer Science, Open University of the Netherlands
Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands
Email: harrie.passier@ou.nl & johan.jeuring@ou.nl

Abstract. Course material for electronic learning environments is often structured using ontology and schema languages. During the specification and development of course material, many mistakes and errors can be made. In this paper we introduce schema-analysis as a technique to analyse structured documents, and to point out (possible) mistakes introduced by an author during authoring. With this technique we are able to produce valuable feedback. We show the technique at work using six categories of mistakes and two types of schemata.

Introduction

Electronic learning environments (LE's), comprising eLearning systems, intelligent tutoring systems etc., are often complex tools. Since instances of such systems, for example for a particular course, are often written by non computer science experts, authoring tools have been developed to support the development of such courses. More open-ended authoring tools for LE's allow for more flexibility in both the form of the content and the order of the steps to design an LE [11]. This flexibility implies a higher probability of mistakes such as inconsistencies and inaccuracies. To improve the quality of LE's, an authoring tool should include mechanisms for checking the authored information on for example accuracy and consistency. Murray [11] mentions several such mechanisms. In this article we introduce schema-analysis, with which we are able to detect a number of the possible mistakes that can be introduced by an author during authoring a course.

The results presented in this paper are part of a project in which we will investigate general feedback mechanisms to learners as well as to authors [12]. To be able to produce semantically rich feedback we imagine an environment that contains several types of knowledge, like domain, task, education and feedback knowledge, which are represented by ontologies. These ontologies are the arguments of a general feedback engine, which observes student and author activities and matches these against the argument ontologies. This framework, a general feedback engine and the use of ontologies as arguments, supports the constant requirement for flexibility, adaptability and reusability of knowledge structures in LE's [2]. In this article we focus on feedback to support authoring LE's.

During authoring different aspects of a course, for example content, structure, and the ontologies, will be authored. In this article we focus on course structure and domain ontology, which we represent by IMS Learning Design (IMS LD) [8] and RDF.

Using IMS LD an author defines the structure of a course in a flexible way. IMS LD supports a wide range of pedagogies in online learning. Rather than attempting to capture the specifics of many different pedagogies, it does this by providing a generic and flexible language. With such a flexible language, an author can easily make mistakes. These

mistakes can be partly prevented by using templates. Some drawbacks of templates are: loss of flexibility, because an author must follow the steps prescribed by a template, and problems with maintainability: it is hard to maintain documents produced by means of templates. With schema-analysis we maintain flexibility, are able to produce feedback when an author makes a mistake, and leave the author, as a didactic professional, free to accept or not accept the feedback information [3]. The freedom to accept or not accept feedback is important. When a (possible) mistake is signalled, it is the author's decision to reject or accept the warning. Sometimes, it could be the author's intention to deviate from rules. What the system signals as a possible mistake may be correct from the author's point of view.

To determine the quality of a course, we want to detect whether or not the following properties hold for a course. If such a property holds, this may signal (the absence of) a potential mistake: (1) Completeness: Are all concepts that are used in the course defined somewhere? (2) Timely: Are all concepts used in a course defined on time? (3) Recursive concepts: Are there concepts defined in terms of it self? (4) Correctness: Does the definition of a concept used in the course correspond to the definition of the concept in the ontology? (5) Synonyms: Are there concepts with different names but exactly the same definition? (6) Homonyms: Are there concepts with multiple, different definitions?

Since a course and course related material are represented by means of schema languages such as IMS LD and RDF, we can use schema analysis techniques to answer the above questions, and to produce feedback about possible mistakes for authors. We have implemented the mentioned analyses as six distinct schema-analyses, which we show at work in a simple course structure and domain ontology. We have yet to develop examples in the context of real courses.

Schema analysis techniques are based, amongst others, on mathematical results about fixed points. Since these results are not widely known, we will explicitly show how to use them in the context of schema analyses. Schema analyses will be expressed in the functional, declarative, programming language Haskell, since this allows us to stay close to the mathematical results we use. We briefly explain Haskell, for more information see [7].

This article is structured as follows: Section 1 briefly explains what we mean with schemata and introduces the languages we use to represent them. Furthermore, we extend IMS LD to be able to define more structure. Section 2 introduces schema analysis. The Haskell constructs we use are introduced in section 2.1. Sections 2.2 and 2.3 presents the necessary data structures and algorithms. The last two sections discuss related work (section 3) and conclude (section 4).

1. Schemata and representations

An ontology specifies the objects in a domain of interest together with their characteristics in terms of attributes, roles and relations. Using an ontology many aspects of a certain domain can be represented, such as categories (taxonomic hierarchy), time, events and composition [13]. A composite object contains objects related to other objects using ' has_part' or 'uses' relations. Any object that consists of parts is called a composite object. A composite object has structure: the parts and their relations. Such a structure description is called a script or a schema. In this article we focus on schemata.

Domain ontology - To represent a domain ontology we use RDF, which can be used to represent meta-data as well as the semantics of information in a machine accessible way. RDF is a universal language that describes resources. The basic building block of RDF is a triple: <resource, property, value>, which defined concepts and related concepts. For example the concept 'cycle_wheel' consists of (has parts) the concepts 'rim' and 'spoke' (<cycle_wheel, has_part, rim> and <cycle_wheel, has_part, spoke>).

Course structure - XML is a language for structuring documents. A data type definition (DTD) describes the type of a set of XML documents. IMS LD [8] is a DTD developed to represent structures of e-courses. The content of a course is presented in a structured way, and activities in an activity-structure. For the examples in this paper we focus on the Activity-model, which consists of several elements: Metadata, Objectives, Prerequisites, Environment and an Activity-description. An activity-description consists of nine elements. One of them is the What-element, which contains the instruction for the activity to be performed. Possible instructions are grouped together by the parameter entity Extra-p. To be able to add more specific annotations to content and structure we introduce two new elements in the Extra-p element: *Definition* and *Example*. Furthermore, we introduce a new attribute *Educational-strategy* of the element Activity with two possible values: *Inductive* and *Deductive*. Introducing such elements will make it possible to structurally analyse educational material. These elements serve as examples to illustrate the analysis techniques at work. In practice many elements can be added, depending on the desired analyses. Listing 1 shows only the relevant elements and attributes related to the activity-model together with the newly defined elements example and definition. The new elements and attribute are marked in bold.

```
<!ELEMENT Activity  %Activity-model; >
<!ATTLIST Activity
        …
        Educational-strategy (Inductive | Deductive) >
<!ENTITY %Activity-model "(Metadata?, …, Activity-description)" >
<!ELEMENT Activity-description  (Introduction?, What, How?, …, Feedback-description?) >
<!ELEMENT What  %Extra-p; >
<!ENTITY %Extra-p "(…| Figure | Audio | Emphasis | List | … | Example | Definition)*" >
```

Listing 1. Parts of the activity-model in IMS LD definition

The definitions of the new elements *Definition* and *Example* are presented in listing 2.

```
<!ELEMENT Definition   (Description, Concept, RelatedConcept+) >
<!ATTLIST Definition Id ID    #REQUIRED
                      Name CDATA #REQUIRED>
<!ELEMENT Example (Description, Concept, RelatedConcept+) >
<!ATTLIST Example Id  ID    #REQUIRED
                     Name CDATA #REQUIRED
                     Belongs-to-definition IDREFS #REQUIRED>
<!ELEMENT Description (CDATA)>
<!ELEMENT Concept EMPTY>
<!ATTLIST Concept Id ID    #REQUIRED
                   Name CDATA #REQUIRED
<!ELEMENT RelatedConcept EMPTY>
<!ATTLIST Concept Id ID    #REQUIRED
                   Name CDATA #REQUIRED
```

Listing 2. Definition of the new elements

2. Schema analysis to detect authoring problems

The schemata given in Section 1 represent structural aspects, which can be analysed. In this section we give some examples of schema-analyses that determine whether or not certain properties hold. The results of these analyses form the basis of feedback to the author. The analyses take the schemata as input. In this paper we perform two types of analyses: 1) the analysis of structural properties of a schema, for example the recursive property, and 2) the

comparison of a schema with one or more other schemata, for example to test the correctness of a definition.

2.1 Haskell preliminaries

The *tuple* data type $(t_1, t_2, ...t_n)$ is constructed from component types. It consists of values $(v_1, v_2, ..., v_n)$, in which $v_1 :: t_1$, etc (where :: means 'is of type'). Function *fst* selects the first element of a pair, *fst* $(x, y) = x$, and function *snd* selects the second element. We use the data type *list* extensively. The empty list is denoted by [ ], and the concatenation of two lists x and y is denoted by x ++ y. Prepending an element x to a list xs is denoted by x:xs. In a list comprehension $[x \mid x \leftarrow xs, test\ x]$, a new list is generated from the list xs. Each element x of xs is tested, and, if the test succeeds, added to the new list. Function *map f* takes a list and applies function f to all elements in the list, so *map f xs* = $[f\ x \mid x \leftarrow xs]$. Anonymous functions can be constructed using lambda notation (\), so function $\backslash(x, y, z) \rightarrow (x, y)$ selects the first two components of a triple. Function *null* tests if a list is empty: *null* [ ] = *True*. To check if an element x is an element of list xs, we use the expression *elem x xs*. Function *zip* takes two lists and returns a list of corresponding pairs: *zip* [1,2] [3,4,5] = [(1,3), (2,4)], where extra elements in the longer list are discarded. Functions *head* and *tail* extract the first and the remaining elements of a nonempty list, respectively. Function *inits* returns the list of initial segments of its argument list: *inits "abc"* == [*""*,*"a"*,*"ab"*,*"abc"*], and function *tails* returns the list of all tail segments of its argument list: *tails "abc"* == [*"abc", "bc", "c",""*].
Function composition composes two functions: the output of the second function (g) becomes the input of the first function (f): $(f \cdot g)\ x = f\ (g\ x)$. The type of a function $f :: t_1 \rightarrow t_2 \rightarrow t_3$ can be read as: function f takes two arguments of types t_1 and t_2 and returns a value of type t_3. Not all arguments have to be mentioned in a function definition, so *completeCourse c = null* (*undefinedConcepts c*), with type *completeCourse* :: *Course* $\rightarrow$ *Bool*, equals *completeCourse = null . undefinedConcepts*. Functions can be passed as parameters. For example, in *map isEven* [1,2,3,4] the type of *map* is *map* :: (*Int* $\rightarrow$ *Bool*) $\rightarrow$ [*Int*] $\rightarrow$ [*Bool*]. Choice between conditions is represented by a vertical bar '|'. For example

```
max x y | x >=y      = x
        | otherwise  = y
```

means: if the guard $x >= y$ is true then return x, otherwise return y.

2.2 Data structures and definitions

We represent an ontology with a list of concepts. A concept is a tuple consisting of a concept identifier and a bag (multiset) of related concepts. In this definition we abstract from concept names, attributes and cardinalities.

```
data Ontology          = Ont [Concept]
type Concept           = (Id, RelatedConcepts)
type RelatedConcepts   = Bag
type Bag               = [Id]
type Id                = String
```

Note that a *data* type definition in Haskell introduces a constructor function for the data (*Ont* in the case of *Ontology*), whereas *type* definitions use the constructors of the types used. The structure of the data type Course follows the IMS LD definition (see listing 1) and consists of an identifier and a list of activities. Extra-p is limited to example and definition.

```
data Course            = C (Id, [Activity])
type Activity          = (Id, Estrategy, [Extra_p])
data Estrategy         = Inductive | Deductive
data Extra_p           = Ex (Id, ConceptId, RelatedConcepts, DefRefs)
```

```
                                | Def (Id, ConceptId, RelatedConcepts)
type RelatedConcepts        = Bag
type DefRefs                = Bag
type ConceptId              = Id
```

terminalConcepts are the set of concepts with no related concepts, *nonTerminalConcepts* the set of concepts with at least one related concept, and *allConcepts* the set of all concepts. Function *reachable* :: [*Concept*] → [*Concept*] → [*Concept*] takes as input: (1) a set of 'productions' (a set of concepts) and (2) a set of concepts, and returns for these last concepts the concepts that are reachable using the productions. Function *reachableTerminals* :: [*Concept*] → [*Concept*] → [*Concept*])) returns the set of terminal concepts that are reachable from the set of concepts using the productions.

Example – If *o* = *Ont* [(*a*, [*b,c*]), (*b*, []), (*c*, [*d,e*]), (*d*, []), (*e*, [])] :: *Ontology,* where the letters represent concepts, then:
terminalConcepts = [(*b*, []), (*d*, []), (*e*, [])], *nonTerminalConcepts* = [(*a*, [*b,c*]), (*c*, [*d,e*])], *allConcepts* = [(*a*, [*b,c*]), (*b*, []), (*c*, [*d,e*]), (*d*, []), (*e*, [])], *reachable nonTerminalConcepts allConcepts* = [(*a*, [*b,c,d,e*]), (*b*, []), (*c*, [*d, e*]), (*d*, []), (*e*, [])] and *reachableTerminals nonTerminalConcepts nonTerminalConcepts*= [(*a*, [*b,d,e*]), (*c*, [*d,e*])].

Functions *reachable* and *reachableTerminals* calculate a fixpoint by means of function *limitBy*. Function *reachable* is defined as:

```
reachable productions concepts = limitBy equalConcepts (expand productions) concepts
```

Function *expand* expands *concepts* using *productions*: if production (*β*, *ω*) is used, all related concepts *xs++β ++ys* are expanded to *xs++β ++ω++ys* removing duplicates. Function *limitBy* repeatedly applies function *expand* until a fixpoint is reached, in which each concept is bound to all concepts it can reach. Function *equalConcepts* checks whether or not two sets of bindings of reachable concepts are equal. A fixpoint is reached when *equalConcepts* returns true.

Example – Suppose *productions* = [(*a*, [*b,c*]), (*c*, [*d,e*])], *concepts* = [(*a*, [*b,c*]), (*c*, [*d,e*])] and function call *reachable productions concepts*. At the start *concepts* equals [(*a*, [*b,c*]), (*c*, [*d,e*])]. After one iteration it equals [(*a*, [*b,c,d,e*]), (*c*, [*d,e*])]. After the second iteration the value is unchanged: [(*a*, [*b,c,d,e*]), (*c*, [*d,e*])], so a fixpoint is reached and each concept is bound to the set of all reachable concepts from that concept.

Function *limitBy*, defined by

```
limitBy :: (a → a → Bool) → (a → a) → a → a
limitBy eq h s
  | eq s next  =  s
  | otherwise  =  limitBy eq h next
  where next = h s
```

only terminates if its argument function of type *a* → *a* is continuous on a *complete partial order* or *CPO* [6], which informally means that the argument function should be increasing on a restricted domain.

Function *reachableTerminals* is implemented in similar way. Instead of function *expand* function *derivationStep* is used as argument of function *limitBy*. Using production (*β*, *ω*), all related concepts *xs++β ++ys* are replaced by *xs++ω++ys.* More details about efficient algorithms for computing fixpoints for grammar analyses can be found in [9].

2.3 Solving authoring problems with schema analysis

In this section we describe six algorithms (four briefly and two in more detail), which can be used to signal the (possible) mistakes listed in the introduction of this paper. The complete code can be obtained from:

http://www.ou.nl/info-alg-inf/Medewerkers/en_Passier.htm.

Completeness – We distinguish three kinds of (in)completeness: (1) within a course, (2) within an domain ontology and (3) between a course and an domain ontology. If a concept is used in a course, for example in a definition or an example, it has to be defined elsewhere in the course. The undefined concepts in a course are calculated in three steps: (1) determine the set of concept id's that appear in the right- and left hand sides of concepts within examples and all concept id's that appear in the right hand side of concepts within definitions (used concepts), (2) determine the concept id's that appear in the left-hand side of concepts in definitions (defined concepts) and (3) check that each of the used concepts appears in the set of defined concepts. A course is complete if all concepts used appear in the set of defined concepts. Completeness can also be applied to an (domain) ontology, and between a course and an ontology. The first one checks whether all used concepts in the ontology are defined in the same ontology, the second one if all used concepts in a course are defined in the ontology. The same three steps are performed in both functions.

Timely – A concept can be used before it is defined. This might not be an error if the author uses an inductive instead of a deductive strategy to teaching, but issuing a warning is probably helpful. Furthermore, there may be a large distance (measured for example in number of pages, characters or concepts) between the definition and the use of the concept, which is probably an error. We define the function *timely* to determine whether or not concepts in a course are defined in time and a function *outOfOrderConcepts* to list the concepts that appear to be out of order.

```
timely :: Course → Bool
timely = null . outOfOrderConcepts
```

In function *outOfOrderConcepts,* function *extractActivities* returns for every activity in the course the tuple (*Estrategy,* [*Extra_p*]) and puts these tuples in a list *activities*. Then, using functions *inits* and *tails* every [*Extra_p*] list is split as follows: for every element *x* in the list [*Extra_p*] the list is subdivided into a left part (*epl*), which contains all elements to the left of element *x,* and a right part (*epr*), which contains element *x* as and all elements to the right of *x.* For example, for the input list [*e,d*] we get [([], [*e,d*]), ([*e*], [*d*]), ([*e,d*], [])], where *e* is example and *d* is definition. Finally, function *intime* tests the timely constrains for all tuples (*es*, (*epl,epr*)): if the first element of *epr* is a definition and the educational strategy is deductive, then: 1) a related example appears after the definition, and 2) no related example appears before the definition (tested by *elemBy eqConcept c* in the code below). In case of an inductive activity, a related example appears before the definition and no related example appears after the definition. Function *intime* is always true if *epr* is empty or the first element of *epr* is an example.

```
outOfOrderConcepts :: Course → [Extra_p]
outOfOrderConcepts c =
  let activities = extractActivities c
      split      = [ (es,s) | (es,eps) <- activities, s <- zip (inits eps) (tails eps)]
  in [head epr | (es,(epl ,epr)) ← split, not (intime (es, epl, epr))]

intime (_ , _ ,[])                          = True
intime (_ , _ ,Ex (j, c, cs ,r):_)          = True
intime (Deductive, epl, Def (j, c ,cs):epr) = elemBy eqConcept c epr && not (elemBy eqConcept c epl)
intime (Inductive, epl, Def (j ,c, cs):epr) = elemBy eqConcept c epl && not (elemBy eqConcept c epr)

eqConcept id (Def (i,c,cs))  = False
eqConcept id (Ex (i,c,cs,r)) = id == c
```

Recursive concepts – A concept can be defined in terms of itself. Recursive concepts are often not desirable. If a concept is recursive, there should be a base case that is not recursive. Recursive concepts may occur in a course as well as in an ontology. We define two functions: *recursiveOntology* and *recursiveCourse* which take an ontology respectively a course as argument. Both first extract all concept definitions, and use function *recursiveConcepts.* We show the definition of *recursiveOntology*.

```
recursiveOntology :: Ontology → Bool
recursiveOntology = not . null . listRecursiveConceptsOntology

listRecursiveConceptsOntology :: Ontology → [Id]
listRecursiveConceptsOntology = recursiveConcepts . extractAllConceptsOnt
```

Function *recursiveConcepts* calculates for every concept all reachable concepts, as explained in section 2.2. Every concept in *reachables* is checked for recursiveness: a concept is recursive if the concept's Id is a member of the set of reachable concepts. The recursive concepts are collected in a list.

```
recursiveConcepts :: [(Id, RelatedConcepts)] → [Id]
recursiveConcepts allConcepts =
    let nonTerminalConcepts = filter (not . null . snd) allConcepts
        reachables          = reachable nonTerminalConcepts allConcepts
    in [x |(x, y) ← reachables, elem x y]
```

Synonyms – Concepts with different names may have exactly the same definition. For example, concept *a*, with concept definition (*a*, [*c,d*]), and concept *b*, with concept definition (*b*, [*c,d*]), are synonyms. In general, given a set *productions,* two concepts *x* and *y* are synomyms if their identifiers are different, $Id_x \neq Id_y$, and (*reachableTerminals productions x*) equals (*reachableTerminals productions y*).

We define function *synonyms* to check for synonyms in an ontology: for all concepts in the ontology all reachable terminal concepts are determined. Concepts with the same reachable terminal concepts and different concept id's are collected in a list.

Homonyms – A concept may have multiple, different definitions. If for example concept *a* has definitions (*a*, [*b,c*]) and (*a*, [*d,f*]), then these two definitions are homonyms. To list the homonyms in an ontology, we calculate the concepts that appear at least twice in the left hand side of a definition.

Correctness – The concepts in a course should correspond to the same concepts in its domain ontology. To determine whether or not this is the case, for every concept in a course all reachable terminal concepts are determined by function *reachableTerminals.* The set of productions contains the course's concepts completed with the concepts of the ontology for concepts that are not defined in the course. The result of this calculation is compared against the reachable terminal concepts of the same concept defined in the ontology.

3. Related work

Although many authors underline the necessity of feedback in authoring systems [1][3][11], we have found little literature about feedback and feedback generation in authoring systems. Jin et al [10] describe an authoring system that uses a domain as well as a task ontology to produce feedback to an author. The ontologies are enriched with axioms, and on the basis of the axioms the models developed can be verified and messages of various kinds can be generated when authors violate certain specified constraints. The details of the techniques used are not given, and it is not clear to us how general the techniques are. Our contribution is the introduction of schema analysis as a general technique to produce messages about errors of structural aspects of course material.

Aroyo et al. [1][2][3] describe a common ontology (web) based authoring framework. The framework contains a domain as well as a task ontology and supports an authoring process in terms of goals, and primitive and composite tasks. Based on ontologies, the framework monitors and assesses the authoring process, and prevents and solves inconsistencies and conflicting situations. Their requirements for authoring support are: (1) help in consistently building courseware, (2) discovery of inconsistencies and conflicting situations, (3) modularisation of authoring systems (reusability), (4) production of feedback, hints and recommendations, and (5) allow to accept or reject the proposed

solutions. We think that our framework satisfies all these requirements. Schema analysis as a technique could be positioned in (1), (2) and (4).

Stojanovic et al present an approach for implementing eLearning scenarios using the semantic web technologies XML and RDF, and make use of ontology based descriptions of content, context and structure [14]. A high risk is observed that two authors express the same topic in different ways (homonyms). This problem is solved by integrating a domain lexicon in the ontology and defining mappings, expressed by the author itself, from terms of the domain vocabulary to their meaning defined by the ontology. In our approach these mappings are analysed automatically.

In the Authoring Adaptive Hypermedia community the importance of feedback mechanisms in authoring systems has been recognized [5]. Although we have found an impressive amount of authoring tools for adaptive hypermedia [4], we have not found descriptions of technologies used for providing feedback to authors. We expect our results will be useful for authoring adaptive hypermedia as well.

4. Conclusions and further research

This paper shows schema analysis techniques to analyse structural aspects of learning environment related material, and applies these techniques to several example problems.

We have planned further research on the application of these ideas to support students developing artefacts in a specific domain using modelling languages like the unified modelling language (UML). Besides a domain and a course ontology we then also use a task and feedback ontology to produce feedback. Secondly, we have planned to define feedback patterns based on ontologies for educational elements such as certain types of questions, examples, definitions, etc. using IMS LD as modelling language. Thirdly, we have planned to test these analysis techniques in a real environment.

References

[1] Aroyo L., Dicheva, D., Authoring support in concept-based web information systems for educational applications, in Int. J. Cont. Engineering Education and Lifelong Learning, Vol. 14, No. 3, 2004.

[2] Aroyo L., Dicheva D., The new challenges for e-learning: The educational semantic web, Educational technology & Society, 7 (4), 59 – 69, 2004.

[3] Aroyo, L. Mizoguchi, R., Towards Evolutional authoring support systems, Journal of interactive learning research 15(4), 365-387, AACE, USA, 2004.

[4] Brusilovsky, P. Developing adaptive educational hypermedia systems: From design models to authoring tools. In: T. Murray, S. Blessing and S. Ainsworth (eds.): Authoring Tools for Advanced Technology Learning Environment. Dordrecht: Kluwer Academic Publishers, 377-409, 2003

[5] Cristea, A., Authoring of Adaptive Hypermedia: Adaptive Hypermedia and Learning Environments, book chapter in "Advances in Web-based Education: Personalized Learning Environments", Sherry Y. Chen and Dr. George D. Magoulas. IDEA Publishing group, 2004.

[6] Davey B., Priestly H, Introduction to lattices and order, 2e edition, Cambridge University Press, 2001.

[7] Haskell: www.haskell.org

[8] IMS LD: http://www.imsglobal.org/learningdesign/index.cfm.

[9] Jeuring J., and Swierstra D., Constructing functional programs for grammar analysis problems, In Conference Record of FPCA '95, SIGPLAN-SIGARCH-WG2.8 Conference on Functional Programming Languages and Computer Architecture, pages 259--269, 1995.

[10] Jin L., Chen W., Hayashi Y., Ikeda M., Mizoguchi R., An ontology-aware authoring tool, Artificial intelligence in Education, IOS Press, 1999

[11] Murray, T., Authoring intelligent tutoring systems: An analysis of the state of the art. International Journal of AI in education, 10, 98 - 129, 1999.

[12] Passier, H., Jeuring, J., 2004. Ontology based feedback generation in design-oriented e-learning systems, Proceedings of the IADIS International Conferencee-Society 2004, Avila, Spain.

[13] Russell, S, Norvig, P., Artificial intelligence, A modern approach, Prentice Hall Int. editions, 1995.

[14] Stojanovic, L. Staab, S., Studer R., eLearning based on the semantic web, in WebNet 2001 – World conference on the www and internet, Orlando, Florida, USA, 2001.

Creating personalised quizzes both to the learner and to the access device characteristics: the Case of CosyQTI

Petros LALOS[1], Symeon RETALIS[1], Yiannis PSAROMILIGKOS[2]

1- University of Piraeus
Department of Technology Education and Digital Systems
80 Karaoli & Dimitriou, 185 34 Piraeus, Greece
E-mail: {retal, plalos}@unipi.gr

2- Technological Education Institute of Piraeus
General Department of Mathematics
Computer Science Laboratory
250, Thivon & P. Ralli, 122 44 Athens, Greece
E-mail: jpsa@teipir.gr

Abstract. In this paper, we present CosyQTI, a tool for authoring adaptive assessments, which gives the educator/ author significant flexibility in terms of the adaptation that s/he can incorporate in the assessments s/he builds. The assessments can be accessed via a variety of access devices including desktop and handheld ones. We illustrate the architecture of the CosyQTI tool and advocating that the tool fully conforms to the IMS_QTI and web standard, serving the goal of interoperability.

Keywords: Adaptive educational hypermedia systems, Adaptive assessment authoring systems, Standards.

1. Motivation

Computers are increasingly being used in the assessment process in many educational situations and several initiatives have been set up. For example, the e3an (Electronics and Electrical Engineering Assessment Network) project has developed tools for collecting, storing and disseminating questions [http://www.e3an.ac.uk]. Various vendors have proposed commercial self assessment tools, like QuestionMark [http://www.questionmark.com], Can Studios [http://www.the-can.com], etc.
Although these tools enable instructors the creation of various types of assessment exercises (e.g. multiple choice, fill-in the blanks, hot spots, and so on), they do not offer functionality for the adaptive presentation of exercises based on instructional rules. Adding the adaptation capability to the assessment process has been proven advantageous, primarily for the reason that learners are presented with personalized tests, tailored to their needs, preferences and current knowledge, but also because the number of assessment items required can be adjusted, most of the times resulting in fewer items, which implies a shorter, less tedious assessment [1,2]. Only few such adaptive tools can be found in the literature such as to SIETE [3], CATES [4], TANGOW [5]. Details about the hypermedia adaptive assessment tools can be found in [5].

The goal of this paper is to present an innovative tool, which is called CoSyQTI tool, that supports the authoring process and presentation of personalised and adaptive web-based assessment. Adaptation in our research means two things:

i) the generation of a dynamic sequence of questions depending on learner's responses and estimation of his/her knowledge level, as defined by Pitkow and Recker in [6]. It has been showed that web-based adaptive questionnaires can reduce the number and complexity of questions posed to users.

ii) The adaptation of the content of web-based tests to the learner's access device characteristics such as screen dimensions, storage capacity and processing power. In our user case, the tool can be accessed by a PC system, a personal hand-held device (PDA) in compliance with Wireless LAN technologies (802.11b-g) and a 3G cellular device or one compliant with WAP technologies.

Moreover, the CoSyQTI has another innovative feature. As learners answer the presented assessment items, the learner model is updated in order to reflect the current status of their knowledge level as well as to keep logs of the whole interaction. Although in most assessment systems the learner model is not usually accessible by the learners, CoSyQTI allows learners to access parts of their model. It has been argued that the act of viewing representations of their knowledge level can raise learners' awareness of their developing knowledge, and difficulties the learning process, thus leading to enhanced learning [7,8]. In CoSyQTI the learner can alter his/her knowledge level at will or negotiate the changes with the tool and come to an agreed representation. In the mobile learning arena, few systems such as C-POLMILE and MoReMaths (Mobile Revision for Maths) [9], offer open learner modelling.

In this paper, we illustrate the main features of the CoSyQTI tool. While section 1 gave an overview of the motivational aspects of our research providing a brief introduction to the notion of adaptation in the assessment systems, section 2 will present the adaptation mechanisms of the CosyQTI. We will analyse the adaptation mechanisms, the XML manifest of the subject domain, and the learner modelling techniques. Finally Section 3, will comment on evaluation results and give directions for further research.

2. Adaptation in the CoSyQTI tool

2.1 The adaptation philosophy

CoSyQTI's approach advocates that the author of an assessment, i.e. the educator, should be able to make decisions on matters such as:

- which questions should be considered easy or hard,
- how are grades to be interpreted in terms of the user's knowledge level,
- how many questions are necessary to estimate the learner's knowledge with confidence,
- how is the learner's performance going to affect the learner model and many more.

With CoSyQTI, authors are able to create the following type of questions:

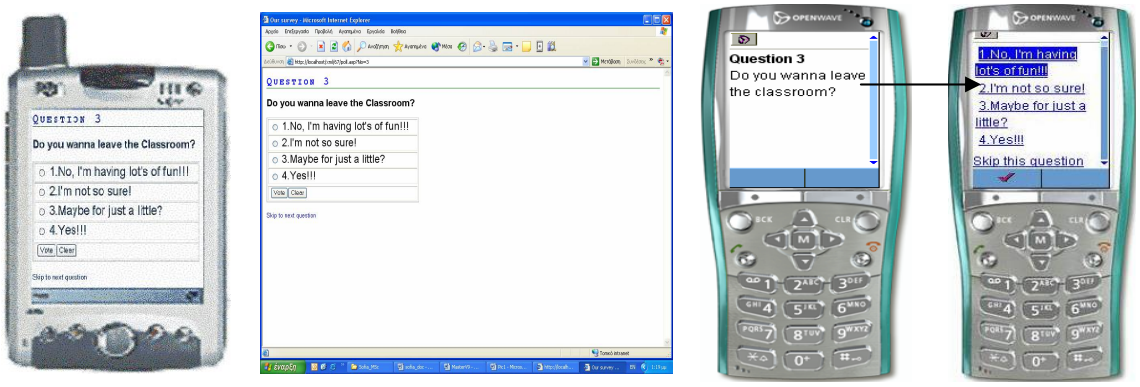
- True/False
- Multiple choice – single, multiple or ordered response
- Image hot spot item

The assessments created fully conform to IMS QTI specification (Question and Test Interoperability) [10], so that they can easily be exported and used by other applications which are also IMS compliant. Furthermore, authors can later open and edit an existing assessment. The IMS QTI initiative proposes the representation of assessment tests in standard XML format, thus allowing interoperability between different assessment tools. QTI structures tests into assessments, sections, and items. An item is the formal name for a question within QTI,

and an assessment is the name used for a test within QTI. The terms "item" and "assessment" are considered more precise and useful for the formal specification, than the looser but more widely understandable "question" and "test" words used in the QTI title.
Nevertheless, CosyQTI innovates by introducing the use of XSL templates (XSLT transformations) designed to adapt the XML formatted questions to the learner's device's specific characteristics like screen dimensions, network availability, storage capacity and processing power.

2.2 Adaptation to access device characteristics

CosyQTI adapts the question items to the access devices' characteristics. The content tagging is a feature that is mainly exploited when adapting the assessment items to the devices characteristics. Predefined elements of the IMS QTI XML manifest are used in order to fragment the assessment content in order to fit into smaller displays like PDA's or WAP cellulars. For example, in Figure 1, the content of a Question is presented in smaller fonts and minimized dimensions when displayed in a PDA compared to a PC, with the use of XSL stylesheets. In an even smaller display, such as a Wap phone, the same content is divided into two element fragment; the Question and the given answers. It should be noted that theexample below is indicative in order to illustrate the look-and-feel and is not part of e-learning activities.



(I) PDA display **(II) Laptop display** **(IIIa) WAP Question display** **(IIIb) WAP Answers display**

Figure 1. Displays of an assessment item in various access devices (PDA, Laptop and WAP cellular).

2.3 Authoring process of adaptive tests

The CoSyQTI tool allows the author to create both adaptive and non-adaptive assessments. The first thing that the author has to do is to create the assessment. As already written in section 2.2, the tool conforms to the IMS QTI specification. Of course, the author does not write any XML code for creating the assessment. In fact he fills in simple web-based forms like the one presented in Figure 2. The editing tool for the assessment supports only desktop and PDA displays and not WAP ones.

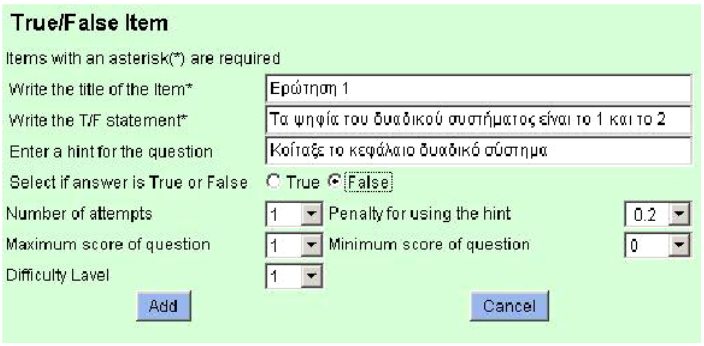


Figure 2. Screen for creating a True/False Question

When the assessment is created, the author can manage the entire assessment from his screen, being able to edit the rules that will be applied to selected items/sections. These rules take the form of IF <condition> THEN <action> rules, where the <condition> refers to the learner's knowledge level on a particular topic, her/his score and/or the assessment's difficulty level, all checked at a given moment during the assessment. This moment is called the rule's trigger-point, it can be any point in the assessment and it is also defined by the author. The <action> refers to the resulting change in the assessment and currently includes a change in the user's knowledge level on the section's topic, a change in the assessment's difficulty level and/or moving the user to a different section. Figure 3 depicts the screen through which the author builds an activation rule and specifies the trigger point, the action that will happen when it will be applied.

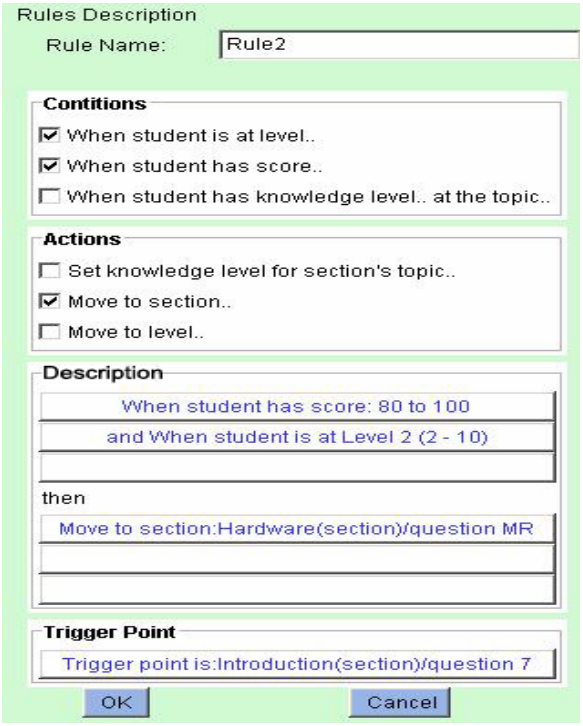


Figure 3. Screen shot of CoSyQTI authoring environment for building adaptive rules for an introductory course on computer science

The questions that will be presented to the learner are displayed on the fly, according to the set of rules that the author creates.

The algorithm used by the CoSyQTI tool to activate and deliver an adaptive assessment can be summarized in the following:

```
If (initialization rule applies) then
        update item selection criteria
Do
        Select appropriate item to present
        Receive Answer
        If ( rule applies) then
                If (rule is of LearnerModel_update type) then
                        update LearnerModel accordingly
                Else if (rule is of item selection type)
                        update item selection criteria
While (NOT end of assessment)
Update LearnerModel accordingly
```

2.4 Learner model description

A learner model in an AHES is essentially the information the system holds about the user and is mainly related to the learning process. This information has to be such that the system can better adapt to the user's individual needs. When observing the learner-computer interaction, several adaptations can take place, some of which do so explicitly and some implicitly, based on the interactive events, the path that the learner has taken, the performance concerning some learning tasks, etc [2]. An adaptive or adaptable educational hypermedia enriches the application functionality by maintaining a "representation of the user" (or "user model") and providing customization mechanisms to modify application features in response to learner model updates. For adaptive educational hypermedia, user model updates are automatically generated by the system (by monitoring and interpreting the user's interactions); for adaptable hypermedia, user model updates are under the user control. The adaptivity/adaptability feedback could be related to the organizational and presentational is-sues of the learning resources (permission to move forward, encouragement to read specific sections, undertake some tasks, etc.).

The user model used by the CoSyQTI has resulted from the selection and combination of elements from IMS LIP and IEEE PAPI [11] standards. Specifically, it consists of the following elements:

- Demographic data - data that remain unchanged, such as age, gender, etc.
- User goals, which are related to the long term and short term learning goals related to learning objectives of specific concepts to be learnt (e.g. "to complete course X").
- User preferences, with respect to the various dimensions of the learning opportunity (e.g. the mode of delivery, accessibility requirements, or assessment).
- User knowledge, which includes the knowledge level about concepts to be learnt and weaknesses and strengths on particular areas, sections or points of the concepts.
- Usage data – historical data which include information like which pages were viewed, in what order, for how long, etc.

The first three elements are considered to be changeable only by the user, whereas the last two are monitored and updated by the system. Goals and preferences are not being exploited in the present version of the system.

There is only one learner model that describes a user of our system, contrary to SIETTE that uses two: a temporary student model with performance information and a more permanent student model, which is also more complete. The learner accesses the hypermedia content of the assessment in the most appropriate user interface design and requests data according to

his/her preference. This request is passed to the server that will return the data matching the requested data to the user profile.The aforementioned approach is shown in figure 4.

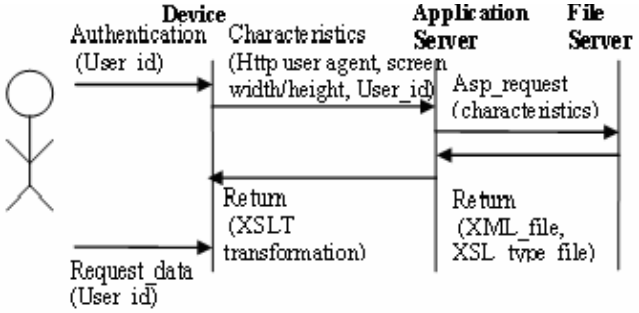


Figure 4. The sequence diagram of the adaptation mechanism in CoSyQTI

3. Evaluation

CoSyQTI is a web-based tool for creating adaptive assessments giving the author the opportunity to define his/her own adaptation rules. The tool can be accessed via various handheld devices. It conforms to the IMS_QTI specification, allowing interoperability. The learner model used has resulted from the combination of elements from IMS LIP specification and IEEE PAPI [11] standard. The CoSyQTI has not been tested in full within real classroom environments. Only small scale usability evaluation studies in a lab have been performed. More specifically, the tool's usability was tested by giving it to 3 users, who are teachers attending an MSc programme on Advanced Learning Technologies. We asked them to perform a few tasks based on specific scenarios that included both the creation of a test as well as completing the test that had been accessed via various devices. We also monitored their reactions. The participants responded that the navigation through the screens was straightforward and that the form fields' meanings were clear. None of the participants had difficulty in creating an assessment, or a section, or an item. It must be emphasized that the they agreed (one strongly agreed) that CoSyQTI is a useful tool, that it was generally easy to use and that it is very useful for an author to be able to create personalized assessments.
In the near future plans we are creating a module that will allow the authors and the learners to access the learner model and change the value some of its elements. Of course, extensive usability tests have been planned.

References

[1] Brusilovsky, P.: *Adaptive and Intelligent Technologies for Web-based Education.* In: Rollinger, C., Peylo, C. (eds.): Special Issue on Intelligent Systems and Teleteaching. Kunstliche Intelligenz, Vol. 4 (1999) 19-25

[2] Kobsa, A.: *Generic user modeling systems*. User Modeling and User-Adapted Interaction, 11(49):49–63, (2001).

[3] Rios, A., Pérez de la Cruz, J.L., Conejo, R.: *SIETTE: Intelligent evaluation system using tests for TeleEducation.* Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98)

[4] Chou, C.: *Constructing a computer-assisted testing and evaluation system on the World Wide Web – the CATES experience*. IEEE Transactions on Education 43(3) (2000) 266-272

[5] Alfonseca E., Marva Carro R., Freire M., Ortigosa A., Prez D., Rodrvguez P.: *Educational Adaptive Hypermedia meets Computer Assisted Assessment*. Proceedings of the International Workshop A3EH in the Adaptive Hypermedia International Conference (AH-2004), Eindhoven (The Netherlands), August 2004

[6] Pitkow, J., Recker, M.: *Using the Web as a Survey Tool: Results from the Second WWW User Survey*. Computer Networks ISDN Systems 27(6) (1995) 809-822

[7] Bull, S., Cui, Y., McEvoy, A.T., Reid, E. & Yang, W. (2004). Roles for Mobile Learner Models, in J. Roschelle, T-W. Chan, Kinshuk & S.J.H. Yang (eds), Proceedings of IEEE International Workshop on Wireless and Mobile Technologies in Education, 124-128.

[8] Bull, S. & Pain, H. (1995). 'Did I say what I think I said, and do you agree with me?': Inspecting and Questioning the Student Model, in J. Greer (ed), Proceedings of World Conference on Artificial Intelligence in Education, Association for the Advancement of Computing in Education (AACE), Charlottesville, VA, 1995, 501-508.

[9] Bull, S. & Reid, E. (2004). Individualised Revision Material for Use on a Handheld Computer, in J. Attewell & C. Savill-Smith (eds), Learning with Mobile Devices, Learning and Skills Development Agency, London.

[10] IMS Question and Test Interoperability Specification, http://www.imsglobal.org/question/

[11] IEEE PAPI Learner http://edutool.com/papi/

Automatic Generation of Exercises for Self-testing in Adaptive E-Learning Systems: Exercises on AC Circuits

Paul CRISTEA, Rodica TUDUCE
University "POLITEHNICA" of Bucharest, Faculty of Electrical Engineering
Spl. Independentei 313, Bucharest, Romania, pcristea@dsp.pub.ro

Abstract. The paper presents aspects of knowledge assessment in the framework of adaptive e-learning systems, focussing on aspect related to self-testing, as a way of providing the input necessary for both system adaptation and user informed decisions. A model of up-ward propagation of evaluation credits and down-ward propagation of validation and acceptance is presented. A prototype example of automatic test generation is described in some details for the special case of electric ac circuits analysis.

Introduction

The spreading of e-learning tools usage is steadily advancing, but it is still lagging far behind expectations. E-learning does not seem to fulfil its promise to become the most important learning methodology, especially in the context of the increased role of continuous and life-long learning. This is often explained by the fact that, despite their recent impressive developments, most of the currently available e-learning tools and environments are still less appealing than the traditional face-to-face teaching methods for both students and tutors [1-2]. From the student's point of view, there are two quite opposite main reproaches: either the complaint about the "lack of human touch", the rigidity of most e-learning tools resulting in the same web pages presented to any user intending to acquire a certain item of knowledge, or the protest against the "over-coaching", the system behaving as knowing better than the user what are his/her needs. The problem of what should be part of a "learner's profile", what and how much of the learner's specific objectives, interests, preferences, and even current state of mind should be tracked as part of the concept, without rising sensitive privacy invasion issues, is still an open question. The natural answer would be to give all the control to the user, to decide how much and what type of help (s)he prefers. Still, there are cases when this approach is simply not feasible, sometimes for the mere fact that the number of parameters to set for controlling the intricacies of the behaviour of an intelligent e-learning environment is too large and full control becomes tedious and repelling in itself. From the tutor's point of view, the main drawback of current e-learning systems is that they tend to require more effort in authoring teaching materials and in preparing lessons, tests and examinations than their classical counterparts, while effectively isolating the student by hiding the whole educational environment, teacher and peers included, behind a machine. The capacity of largely re-using teaching materials, while still filtering them through the tutor personality to an extent that gives the sense of recognized authorship, the permanent synchronous and asynchronous contact among learners and between learners and teachers to a level that

helps establishing an effective cooperative learning environment seems to be the way to the answer in this case. Adequate authoring tools are needed to help the teachers in preparing both learning materials and tests for evaluating the advancement of the students towards their teaching goals. Traditional knowledge assessment is known to be demanding from both parts involved, time consuming, rising emotional aspects that do not contribute to the co-operation among the tutors and students and are prone to trigger a negative attitude from the learners. Nevertheless, a large variety of testing and evaluation tools are required by any intelligent e-learning environment to get the feedback necessary for the adaptation of the teaching system to guide the student along the learning process. Such tools operate best when they are perceived by the user as self-testing helps, giving him/her an effective support and remaining continuously under his/her own control. The learner's full cooperation makes the testing tools useful and an active part of the learning environment.

The paper briefly presents an ILE that implements the computer-supported collaborative work model, focussing on aspects referring to automatic authoring of the self tests for student evaluation the tools used to evaluate and guide the students' advancement towards their learning goals.. A prototype automatic generator of ac circuit exercise is presented in some details, stressing on the general points that could be used for test and exercises in any field of science and engineering involving quantitative analysis.

1. Intelligent e-Learning Systems

The vast majority of the currently used web-based educational systems are powerful integrated systems, like Blackboard [3] or WebCT [4] that provide a large variety of support services to both learners and teachers, but lack adaptability, belonging to the class of Learning Management Systems (LMS). Significant research and implementation effort has been dedicated to develop Intelligent Tutoring Systems [5] and Adaptive Hypermedia [6, 7], able to adapt to learner's objectives, interests, and preferences, i.e., to Learner Profile (LP). Currently, such systems offer remarkable adaptability, but only for specific tasks, lacking integration. Improved distributed architectures, centred on the concept of Intelligent Learning Environments, have been suggested to allow combining the efficiency of LMS with the flexibility of adaptive systems [8-10]. To implement the adaptivity feature, an ILE requires a quite complex structure, with several parallel version of the same learning item (LI), allowing many different learning paths to be selected in accordance with the LPs. This approach requires considerable additional effort in elaborating teaching materials, might require several authors and might need institutional support, but brings the advantage of true flexibility and adaptability. A course, as a teaching material, is no longer a flat juxtaposition of learning items, but a multilevel structure with many parallel branches, along which the ILE recommends an optimal path for a user or for a class of users. As mentioned in [10], it has been argued that authoring learning material and building the structure of adaptive systems tends to become too complicated for the average teacher. Correspondingly, portability – the ability to deploy the content of a system on any other system, and reusability – the ability to search and retrieve learning items (LIs), including lessons, modules, exercises, activities and to reuse them, become strictly necessary features for an efficient implementation of the ILE concept and for the success of the wide scale implementation of this approach. The problem has already been addressed for the case of LMS: the DOD's Sharable Content Object Reference Model (SCORM) ADL initiative [11] aims to provide a comprehensive set of e-learning capabilities enabling interoperability, accessibility and reuse of Web-based learning content. Similar efforts are under way to provide a harmonized set of guidelines, specification and standards for Intelligent Tutoring Systems (ITS) and Adaptive Hypermedia (AH) systems [12, 13]. At the

same time, significant changes in the basic methodology of teaching/learning is to be expected as result of the current advancement and convergence of cognitive psychology and computing science. Paradigms such as just-in-time learning, constructivist approaches, student-centred learning and collaborative approaches have emerged, and are being supported by technological advancements including simulations, virtual reality and multi-agents systems.

2. Methodology, Architecture and Implementation

The system is learner centered, all human and artificial agents being focused on achieving the learning-training tasks. Human agents include, aside the students, authors of teaching materials, tutors, course administrators, and system administrator(s). The pilot web oriented ILE has the server-client distributed multiagent hybrid architecture shown in Figure 1 [14].

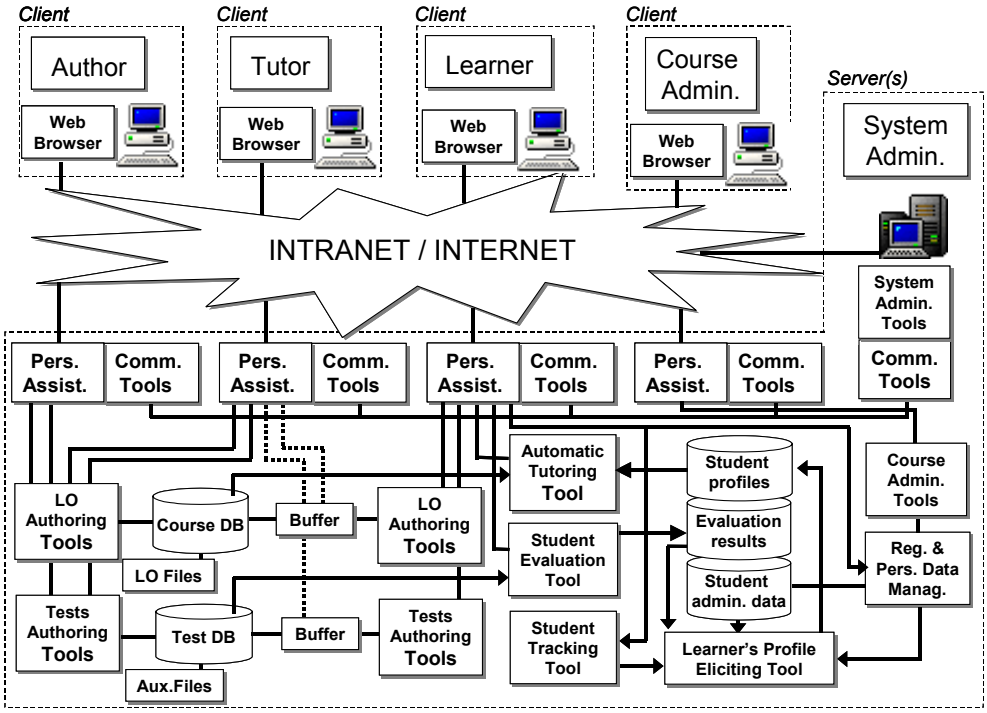


Fig. 1: Architecture of the ILE pilot

The client side requirements are kept to a minimum; each human agent accesses the system via an intranet or internet connection, by using a standard web browser. On the server side, each human agent has a personal assistant (agent) that inter-operates with the other artificial agents providing the system functionality. All personal assistants are linked to communication tools that facilitate the contact among human actors in the system, and provide support to the collaboration among the students. The learner personal assistant (LPA) controls, in the first place, the access to the agents providing the main functions – the Automatic Tutoring Tool and the Student Evaluation Tool, but also to the Learning Item (LI) Authoring Tool and Tests Authoring Tool – thus providing support to a participative learning approach. Students get credits not only for their results in the tests, but also for their involvement in the course development – by contributing with new versions to existing *LI*s, building new questions for tests addressing various *LI*s, etc. When submitted, the student contributions are stored in a buffer, waiting for tutor's validation to be introduced in the course and test databases. The contributions are marked similarly to the successful passing of regular tests. The LPA gives also access to the Registration and

Personal Data Management Tool, and helps tracking student participation and results, passing the corresponding data to the Learner's Profile Eliciting Tool (LPET). The tutor has also a Tutor Personal Assistant (TPA) that provides an interface to all system functionalities offered to a tutor: keeping track of class enrolment, following learners' progress in the training process, reading synthetic data on learners' profile, etc. Authors and tutors have access to authoring tools, to update the course, tests and any other teaching materials. Course administrators have access to all relevant data about courses and students. The system administrator monitors the usage of the resources and takes care of the smooth functioning of the system. The distribution of the functional roles among human actors can be managed from an administrator platform accessed via root privileges.

The server has been implemented as an in-house developed J2EE compatible JAVA web-application, running on the Tomcat Apache Server and using the MySQL database server, both largely accessible and available on UNIX and WINDOWS platforms. The modular approach allows an easy restructuring of the system; e.g., the change of the database affects only the module controlling the database. Database access speed is increased by using a connection pool. Data access (some contained in the MySQL database, some in a system of related directory files) is made through interactive web pages implemented with Java Server Pages (JSP) and Servlets. Each JSP is controlled by a JavaBean which handles the security tasks and data manipulation. Every JavaBean within a module extends the class corresponding to an actor (learner, author, tutor, course and system administrator), being able to control authorized access.

3. Learning Evaluation

Learning evaluation is based on test results, but also on the student involvement in proactive learning, including the contribution to the course continuous development [15]. Care is taken to stimulate students to do more than just recognize textbook material when responding to tests. Students are encouraged to formulate themselves new questions and are given credits when their contribution is included in the question data base (QDB).

To make the system closer to a classic examination that does not require the checking every item in a course, the passing of a higher level *LI* is not conditioned by the passing of all lower *LI*s it contains, but only by the sum of points exceeding the threshold for that level. The acceptance of a higher level *LI* gives credit for all *LI*s below it in the course structure. This results in an up-wards propagation of the awarded points and a downward propagation of the acceptance, as shown in Figure 2. The acceptance of an *LI* is irreversible, so that only warnings and advices are issued if later errors signal possible weaknesses in an already assessed *LI*. The tests are built by randomly choosing from the question data base (QDB) a specified number of questions referring to the given tag (*LI* address or keyword). A test referring to a *LI* can comprise questions attached either directly to that specific *LI*, or to any *LI* placed below it in the course tree structure. The points obtained when making a choice *C* from the set of options O(*Q*) pertinent to question *Q* are recorded at the LI to which the question is attached and

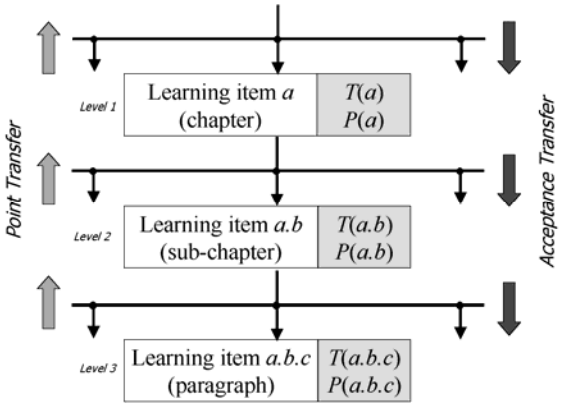


Fig. 2: Evaluation points and acceptance transfers

transferred upwards. The sum of points $P(Q)$ for a question Q results by adding the points for all options selected at Q:

$$SP(Q) = \sum_{C \in S(Q)} P(C), \tag{1}$$

where $S(Q) \subseteq O(Q)$ is the set of options the student has selected at question Q. The correct choices are awarded positive points, the wrong answers – negative points. Assigning negative points to the wrong choices contributes to discourage guessing, but the system success relies on student motivation. The points $P(Q)$ acknowledged for question Q is given by:

$$P(Q) = \begin{cases} SP(Q), \text{ if } SP(Q) < 0, \\ 0, \text{ if } 0 \le SP(Q) < T(Q), \\ SP(Q), \text{ if } SP(Q) \ge T(Q), \end{cases} \tag{2}$$

where $T(Q)$ is the threshold required for the acceptance of the reply to Q. As a consequence, points obtained for a question are taken into account only when exceeding a certain requested minimum. Such a learning appraisal aims at a robust understanding and proper using of the tested knowledge, by discouraging superficial and fragmentary learning. Each *LI* is not evaluated independently, but in the context of the course, in relation with the related *LI*s. The sum of points $SP(LI)$ for a certain learning item *LI* consists not only of the sum of the points obtained for the questions Q referring directly to *LI*, but, also, of the sum of the acknowledged points transferred to *LI* from its children – the learning items *LI'* placed one level below it in the course structure:

$$SP(LI) = \sum_{Q \in LI} P(C) + \sum_{LI' \in C(LI)} P(LI'), \tag{3}$$

where $C(LI)$ are the children of *LI*. Equation (3) ensures that the points obtained for a certain *LI* are transferred upwards, to all its ascendants, without double counting.

The points $P(LI)$ acknowledged for a learning item *LI* depend also on the passing of a certain acceptance threshold $T(LI)$, but, in this case, there is an award $A(LI)$ for the successful completion of the study of *LI* marked by the passing of $T(LI)$:

$$P(LI) = \begin{cases} SP(LI), \text{ if } SP(LI) < T(LI), \\ SP(LI) + A(LI), \text{ if } SP(LI) \ge T(LI). \end{cases} \tag{4}$$

The status $S(LI)$ of *LI* changes from 0 – *pending* to 1 – *studied*, when its threshold, or the threshold of its ascendant, have been passed:

$$S(LI) = \begin{cases} 0, \text{ if } \big(SP(LI) < T(LI)\big) \text{ and} \\ \qquad \big(S(LI') = 0, LI \in C(LI')\big), \\ 1, \text{ if } \big(SP(LI) \ge T(LI)\big) \text{ or} \\ \qquad (S(LI') = 1, LI \in C(LI')), \end{cases} \tag{5}$$

where $C(LI')$ are the children of *LI'*.

Relation (3) shows the up-propagation of the points in the tree-like course structure, while relation (5) corresponds to the down-propagation of the acquired knowledge recognition along the same structure.

4. Automatic Problem Generation

We present in this section a tool that automatically generates sets of ac electrical circuit analysis problems with given complexity. The initial window of the system asks the user, tutor or student, for the global description of the desired set of problems, comprizing the global parameters (set label, the number of problems, number of nodes, number of branches, the range of values for the chord current intensities, twig and chord resistances, twig *emf*s, the number of branches with current source, and the number and type of controlled sources). For each problem in the set, the system generates a distinct topology, starting with building a tree (Fig.3 and Fig.4) and continuing by introducing the chords to generate the circuit graph in Fig. 6 and establish the corresponding essential incidence matrix $[\Lambda_{tc}]$ that describes the circuit topology. In the second part, the circuit parameters (resistances, reactances, independent and controlled sources parameters) and complex variables (complex branch currents and voltages) are chosen or computed sequentially.

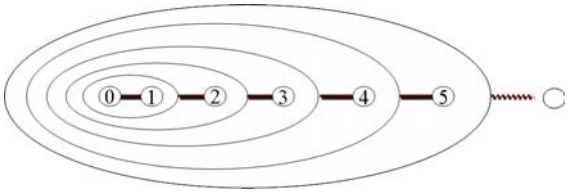


Fig.3. Tree generation: Each additional node is connected by a twig poiting towards previously linked nodes.

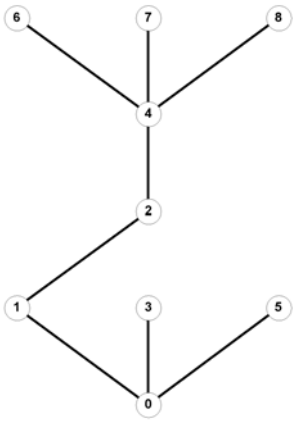


Fig. 4. Example of automatically generated tree with nine nodes

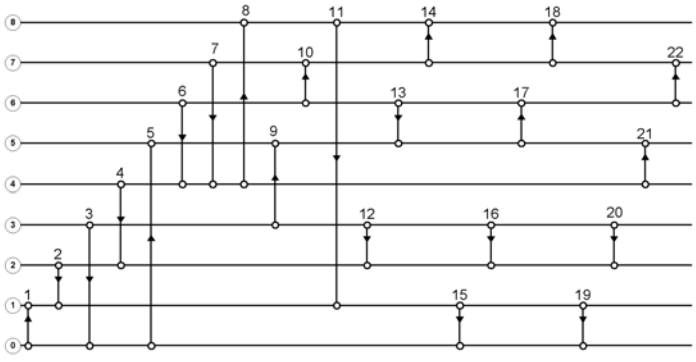


Fig. 5. Example of automatically generated network graph comprising the 8 twigs shown in Fig. 4 and 14 chords. The nine nodes labeled 0 to 8 are shown as connection bars.

The basic idea of the method for automatically generating sets of problems, without the need of solving systems of equations (or, equivalently, of inversing matrices), is to start the procedure from points that allow the simple chain computing off all needed elements. This method can be applied to any similar type of analysis problems, in which the parameters of the system are given, while the variables result as solutions of equations that describe the system.

In the following, all current, voltages, emfs, source currents and impedances are complex magnitudes. Square or rectangular matrices are represented by the symbol of the magnitude between square brackets, while column vectors are represented by a single square bracket to the right of the symbol. The subscript 't' corresponds to magnitudes attached to the twigs (the branches of the tree), while the subscript 'c' – to the chords (the branches of the co-tree).

The successive steps of the algorithm are:

1. Randomly choose the chord current vector $I_c]$;
2. Compute the twig current vector: $I_t] = [\Lambda_{tc}]\ I_c]$;
3. Randomly choose twig complex impedances (resistances and reactances) and complex *emf*s: $[Z_t]$, $[Z_c]$, $E_t]$;
4. Compute the twig voltage vector: $U_t] = [Z_t]\_I_t] - E_t]$;
5. Compute$U_c] = [\Lambda_{tc}]^T\ U_t]$;
6. Compute thechord complex *emf*s : $E_c] = [Z_c]\ I_c] - U_c]$;
7. Concatenate the variables and parameters for all branches:

$$U] = \begin{matrix} U_t] \\ U_c] \end{matrix} \qquad I] = \begin{matrix} I_t] \\ I_c] \end{matrix} \qquad E] = \begin{matrix} E_t] \\ E_c] \end{matrix} \qquad J] = \begin{matrix} J_t] \\ J_c] \end{matrix}$$

8. If independent current sources desired, partially or totally convert some of the independent voltage sources as shown in Fig. 6: $E^{new}] = E] - [\,Z\,]\,J]$;

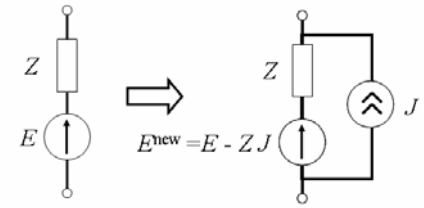


Fig. 6. Partial conversion of an independent voltage source into an independent current source

9. If cross links between branches (controlled sources and mutual inductive couplings) desired, convert some of the independent voltage sources into the selected cross links:

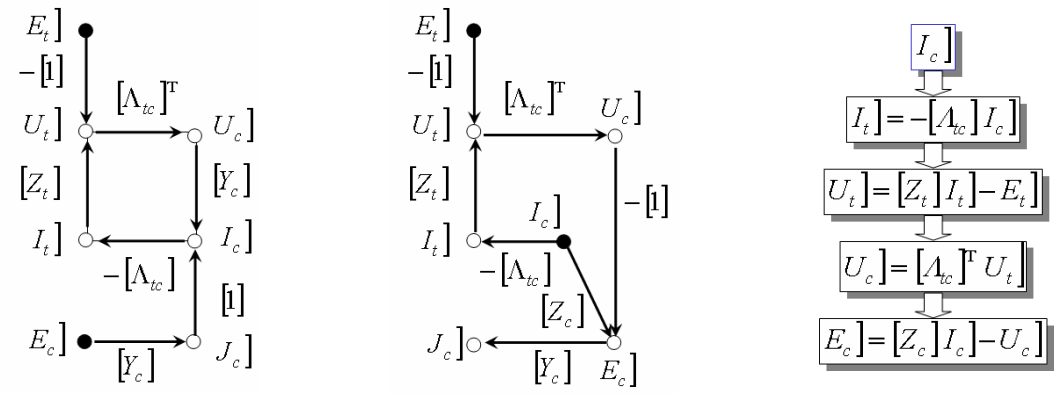
Controlled source (change of independent voltage source emf,s)

$$E^{\text{controlled}}] = [Z_t]\,I] \quad E^{\text{new}}] = E] - [Z_t]\,I]$$
$$J^{\text{controlled}}] = [Y_t]\,U \quad E^{\text{new}}] = E] - [Z][Y_t]\,U]$$
$$E^{\text{controlled}}] = [A]\,U] \quad E^{\text{new}}] = E] - [A]\,U]$$
$$J^{\text{controlled}}] = [B]\,I] \quad E^{\text{new}}] = E] - [Z][B]\,I]$$

Mutual reactances

$$E^{\text{induced}}] = [-\,\mathrm{j}X_M]\,I] \quad E^{\text{new}}] = E] - E^{\text{induced}}]$$

Figure 7 gives the matrix flowgraph representation of circuit equations which are used to generate circuit parameters and variables. The methodology for fast generating circuit variables and parameters implies cutting the loop in the equation flowgraph, so that no matrix inversion is required.



a) Standard analysis problem b) Exercise synthesis problem c) Basic algorithm steps

Fig.7. Methodology for converting an analysis problem (a) into a procedure for automatic generation of exercices (b). Notice the cut of the loop in the matrix flow graph resulting from the change of the initial data.

Because the whole procedure involves no division, it is possible to generate problems that use only integer values, which is a debateble advantage from the engineering training point of view, but can produce more attractive tests. A similar approach can be used for a large class of problems, by chosing variables that open the loops in the system equations flow graph.

5. Conclusions

The paper presents a pilot implementation of an Intelligent e-learning environment (*ILE*) able to adapt to the learner's profile (*LP*) and to encourage active and cooperative learning. Special attention is given to support the authoring of learning items and tests, especially important in an ILE context to assess the students advance towards their learning objectives. Adequate authoring tools can make the tutors task easier, contributing to a better acceptance of e-learning systems, despite the required extra work and can stimulate a pro-active attitude of students. An example of automatic problem generation is given for the case study of a.c. electric circuit analysis. A similar methodology can be applied to convert any engineering analysis problem into a procedure for automatic generation of tests. This approach has the advantage of greatly simplifying the problem of synthesizing system analysis exercises, to the point of making them available to both tutors and students, in the later case for self-testing and computational experiments purposes.

References

[1] D. A. Norman, J.D.Spohrer, Learner-centered education, Comm. of the ACM, 39(4), 1996, pp. 24-27.

[2] M. J. Wooldrige, N.R.Jennings, Intelligent agents: Theory and practice, The Knowledge Engineering Review, 10(2), 1995, pp. 115-152.

[3] Blackboard Inc, Blackboard Course Management System, http://www.blackboard.com/.

[4] WebCT, Course Management System, http://www.webct.com/.

[5] C. Frasson, G. Gauthier (editors), Intelligent Tutoring Systems - At the Cross-roads of Artificial Intelligence and Education, Ablex Publishing Corporation. Norwood, New Jersey, 1990.

[6] De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D. & Stash, N., AHA! The Adaptive Hypermedia Architecture, Proc. of ACM Hypertext Conference, Nottingham, UK, August 2003, pp. 81-84.

[7] P. Brusilovsky, Adaptive hypermedia, User Modeling and User Adapted Interaction, Ten Year Anniversary Issue (Alfred Kobsa, ed.) 11(1/2), 2002, pp. 87-110.

[8] P. Brusilovsky, Student model centered architecture for intelligent learning environments, Proc. of Fourth international conference on User Modeling, 15-19 August, Hyannis, MA, USA. User Modeling Inc, 1994, pp. 31-36.

[9] O. Conlan, C. Hockemeyer, V. Wade, D.Albert, D., & M. Gargan, An architecture for integrating adaptive hypermedia service with open learning environments, In Proc. of ED-MEDIA 2002, vol. 1 of World Conference on Educational Multimedia, Hypermedia & Telecommunications, pp. 344-350.

[10] P. Brusilovsky, A Distributed Architecture for Adaptive and Intelligent Learning Management Systems, Proc. of AI in Education (AIED2003), vol.4., Workshop Towards Intelligent Learning Management Systems, Sydney, Australia, July 2003.

[11] Advanced Distributed Learning (ADL) Initiative, Sharable Content Object Reference Model (SCORM), http://www.adlnet.org/.

[12] A.I. Cristea & A. De Mooij, Designer Adaptation in Adaptive Hypermedia Authoring, Proc. ITCC'03, Las Vegas, US, IEEE Computer Science, 2003, pp. 444-448.

[13] D. Dicheva, L. Aroyo, Alexandra Cristea, Collaborative Courseware Authoring Support, Proc WEB-CATE 2002 – Computers and Advanced Technology in Education, Cancun, Mexico, 2002, pp. 52-57.

[14] P. D. Cristea, Rodica Tuduce, Pilot Implementation of an Intelligent e-Learning Environment, eChallenges e-2004, Vienna, Austria, October 27-29, 2004.

[15] P. Cristea, Rodica Tuduce, Test Authoring for Intelligent e-Learning Environments, Proc. IASTED Intl. Conf., Web-Based Education, Innsbruck, Austria, 2004, pp. 402-407.

Writing MOT, Reading AHA!
- converting between an authoring and a delivery system for adaptive educational hypermedia -

Alexandra Cristea, David Smits and Paul de Bra
Faculty of Mathematics and Computer Science, Eindhoven University of Technology
PB 513, 5600MB, Eindhoven, The Netherlands

Abstract. This paper reports about the recent advances towards establishing a common platform for adaptive educational hypermedia (AEH) authoring. We present the conversion from MOT, a dedicated authoring system, to AHA! used in this context as delivery system for AEH. Moreover, we describe two new representation languages that emerged in the process: a common format for defining the static material, CAF, and an extended adaptation language for the description of the dynamic behaviour, LAG. Finally, some evaluations are shown and conclusions are drawn.

1. Introduction

Adaptive Educational Hypermedia (AEH) [3],[4] is dedicated to bringing personalization in the closed and open corpus hypermedia, e.g., the WWW. The AEH community is increasing in size and volume of research, indicating the importance of this topic at the global scale.
Within AEH, *authoring* is a serious problem, which has only recently started being given more attention by the research community. As authoring of static educational content is quite different from authoring of adaptive content, there is a growing understanding that a great deal of help is necessary for authors. In order to make the authoring process easier, in the sense of 'authoring once, delivering many' [18], there are two major possible approaches:

1. a common language, a *lingua franca*, used by all authors of AEH, and secondly
2. the use of *converters* between AEHs.

Very few such attempts have been made so far, mainly clustering along the second possible approach. In the following we shall discuss the current converters between AEHs shortly.
A pioneer work in the direction of converters was that from Interbook [6] to AHA! [14]. The approach in Interbook to AHA![13] is similar to the one presented in this paper, in the sense that Interbook is used as an authoring tool, and AHA! as a delivery tool for AEH. Moreover, the delivery end is the same. However, here the similarity ends. In Interbook, authors can create AEH applications in simple Word documents [13]. Adaptive behaviour is introduced via annotations in these documents. The main purpose in this conversion was to recreate the layout, look and feel of Interbook presentations, i.e., adaptive multiframe views. The conversion is done via a *compiler*, which doesn't use a semantically relevant intermediate format of representation. Finally, adaptivity in Interbook is mainly based on prerequisite relations and linking to glossary. No higher-level, reusable pedagogical strategies can be represented.
Another similar converter was built between MOT [9] and WHURLE [16]. This approach resembles the one presented in this paper, from the point of view of the authoring end, MOT. However, similar to the Interbook to AHA! converter, the approach in MOT to WHURLE [18] also uses a compiler. Moreover, as WHURLE cannot represent adaptive, but only adaptable

educational hypermedia, the MOT system dynamics (adaptation maps) cannot be part of the conversion. Similarly to the current approach, the MOT to WHURLE system convertor has also been tested in practice with students.
Another interfacing exercise has been done between Claroline [8] and AHA! [14]. The approach in Claroline and AHA! [2] is quite different from the one presented in this paper, as, first of all, there is no conversion as such: Claroline and AHA! function as two separate modules of the same adaptive collaborative delivery system, ASCIL. The adaptive support for collaborative learning is integrated with the information contained in the User Model (student), which is kept in operation by AHA!. That is to say that the adaptive tasks are the result of individual learning in AHA! as a starting point [2].
Finally, an interesting conversion from the point of connecting the academia prototypes with extant commercial applications is the conversion between MOT and Blackboard [18]. Here, a regular educational environment, the Blackboard™ Learning Management System [4], is added adaptivity and personalization by means of authoring the goal-oriented material in an Adaptive Hypermedia authoring system, MOT [11], and delivering it in Blackboard via a conversion to the SCORM specification. This represents the first attempt to connect Adaptive Hypermedia and Learning Management Systems [18].

2. Authoring with the MOT system

MOT [1] is a generic authoring system for adaptive educational hypermedia. Information about MOT (and the software) can be found at [16]. MOT is based on the LAOS model [11] for authoring of adaptive hypermedia, and implements some of the LAOS layers, as shown in the following subsections.
In order for material authored in MOT to be converted and represented in another adaptive hypermedia system, at least some components of the MOT system have to be converted and interpreted. Naturally, the more components will be integrated in the conversion, the more complete and accurate the end-result will be.

2.1 Domain Maps

Domain maps structure and organize the actual resources of the learning environment, as well as their intrinsic characteristics. These resources represent the content that is traditionally authored in any learning environment. Domain maps in MOT follow the LAOS specifications, by consisting of hierarchical domain *concept maps*, that are in turn built of typed attributes. Moreover, MOT also allows, according to the LAOS model, another type of relations between concepts, beside the *hierarchical* ones, called *relatedness* relations. Attributes in MOT represent, same as in LAOS, different content variants (or aspects) of the domain concept they belong to. For instance, the *title* attribute is the most concentrated way of representing a concept, which can, however, also be represented via the *text* attribute, via the *introduction* attribute, a.s.o.
A sample screenshot of how a domain map can be authored in MOT, and thus what has to be converted into another system is represented in Figure 1.

2.2 Goal and Constraints Maps

The goal and constraints model contains *all* information (resources and links between them) of the adaptive hypermedia system relevant for a delivery purpose. According to LAOS, within the educational domain, this model filters, regroups and restructures the domain model, with respect to an instructional goal. Moreover, these maps add the necessary information (content) that is goal-dependent. In MOT, this *goal* is a pedagogic learning goal, therefore these maps are called *lesson maps*. Thus, these maps add all the necessary pedagogic material and linking for the student. These maps are represented also as concept maps, according to the LAOS

model. In MOT, they are implemented as a simplified *overlay* over the domain maps. Only in these lesson maps will the concepts be ordered, as the *order* in which the information is presented to the learner is dependant on pedagogic constraints and teaching strategies. Moreover, other pedagogic information can be authored via these maps, as for instance, *labels* and *weights* can be added to concepts, to semantically annotate the intended use of the respective concepts. Figure 2 shows ordered concepts (see numbers on the left side in the left frame), as well as semantically labeled concepts (e.g., 'beg' for beginners, 'beg_vis' for visual beginners and 'beg_ver' for verbal beginners; the visual-verbal differentiation is taken according to the ILS questionnaire [13]).

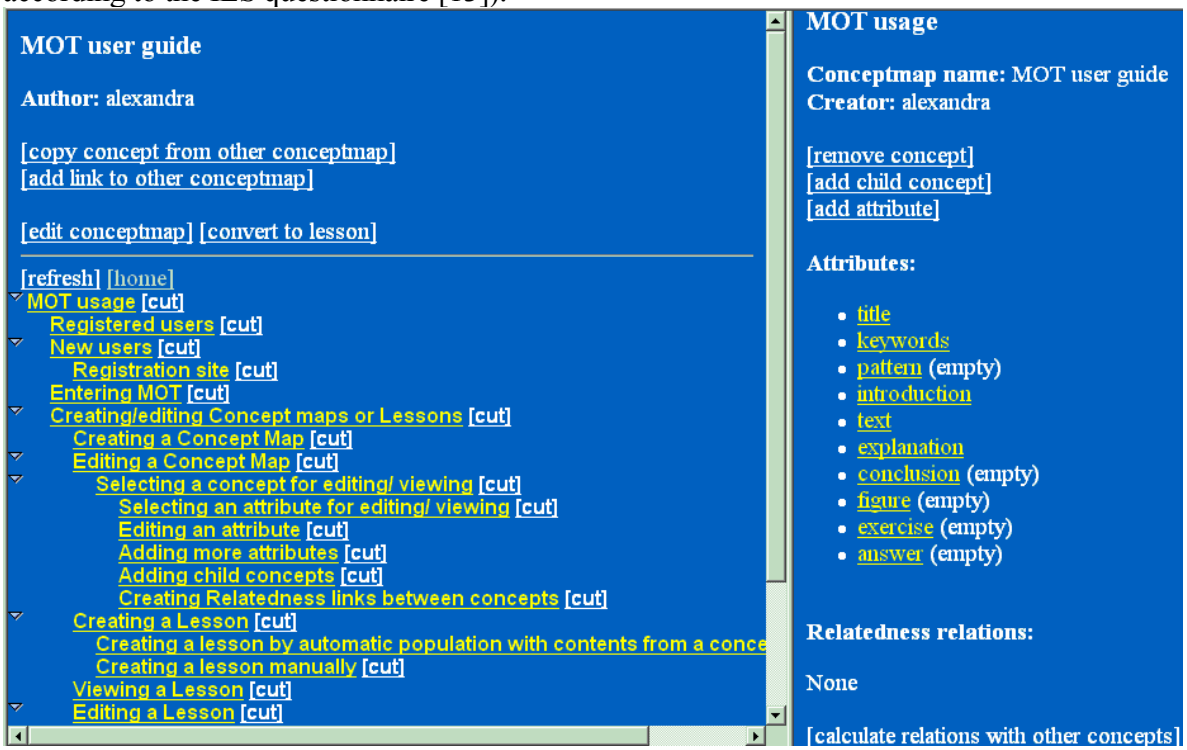


Figure 1. Authoring Domain Maps in MOT

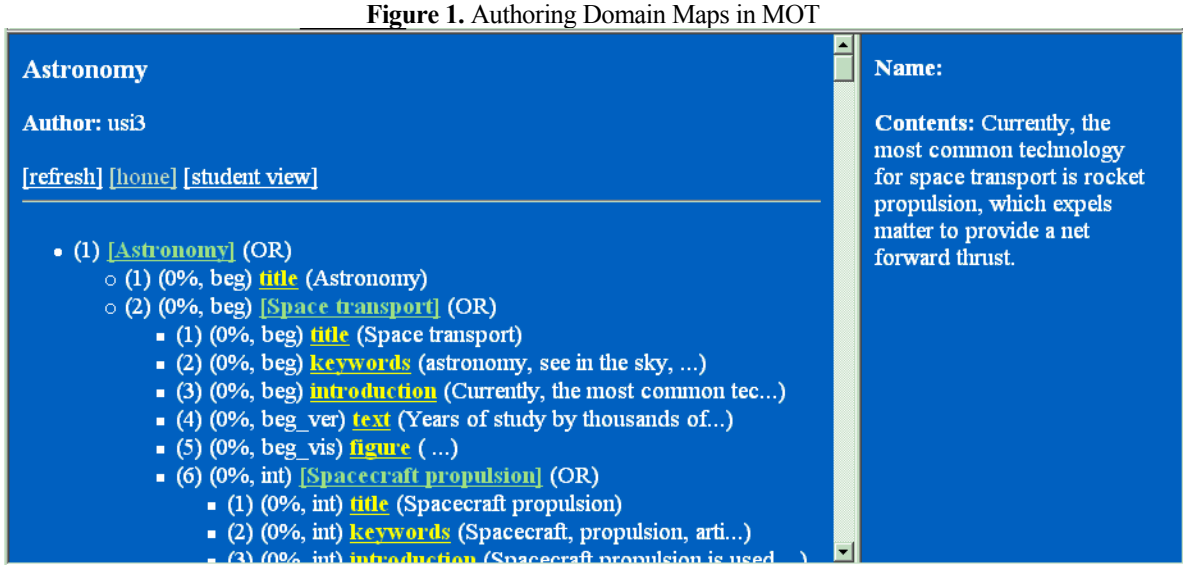


Figure 2. Authoring Lesson Maps in MOT

2.3 User & Presentation Maps

The LAOS model specifies the representation and authoring of separate user maps and presentation maps.

The *user maps* should contain all the necessary variables and initial values that are necessary to represent the user (for educational applications, the *learner*). Typical variables are knowledge level, interests, learning styles, etc. The user model can be an overlay to either the goal and

constraints or the domain model (e.g., the knowledge of a certain topic represented by a goal and constraints model concept). Alternatively, the user model can also contain free variables (such as background knowledge).

Presentation has to take into account the physical properties and the environment of the presentation and provide the bridge to the actual code generation for the different platforms (e.g., HTML, SMIL, XHTML, etc.). Presentation makes the difference between different devices of display, such as handheld devices, desktops, laptops, etc. This part in the LAOS model is concerned with the formatting so that the information appears nicely in the page, with questions such as the ideal page length, where chapters of the presentation should be cut to form pages, how and where multimedia should appear (from the point of view of display possibilities), colours, fonts, etc. In a simplified authoring experience, these settings should be generated fully automatically, in order to make the authoring process easier. For more complex authoring, the authors should be able to specify this type of settings. Taking a further step, presentation should be concerned with other user-related settings that are however not directly dependant on the user, such as adaptation to different load on the network, traffic peaks, scalability, alternate routes, distributed servers, etc.

The most recent implementation of MOT actually contains these two separate authoring systems, and some initial testing has been done with the user map authoring environment. However, in the current conversion into AHA! these systems haven't been used. Instead, a simplified version of user and presentation map initialization have been used, directly represented in the adaptation maps, as follows.

User model variables were defined as (in the following, UM stays for User map, GM for Goal and Constraints (lesson) map, PM for Presentation map and DM for domain map):

- *Overlay user map variables* declaration: e.g., the knowledge variable for every concept in the lesson map:

```
// VARS UM.GM.Concept.knowledge
```

- *Independent user map variables* declaration: e.g., the stereotype variable for any user

```
// VARS UM.GM.stereotype1
```

The actual initialization of these variables is done in the first part of the adaptation map:

- *Overlay user map variables* declaration: e.g., the knowledge variable for every concept in the lesson map is set initially to 0:

```
initialization ( // … other initializations
while UM.GM.Concept.knowledge != 0 ( UM.GM.Concept.knowledge = 0)
)
```

- *Independent user map variables* declaration: e.g., the stereotype variable for any user in using this lesson is set initially to 0:

```
initialization ( UM.GM.stereotype1 = beg )
```

Presentation model variables were defined as:

- *Overlay presentation map variables* declaration: e.g., the display ('show') variable for every concept in the lesson map:

```
// VARS PM.GM.Concept.show
```

- *Independent presentation map variables* declaration: e.g., the Boolean variables determining if a table of contents ('Menu'), 'next' button or list if 'To Do' items appears in the presentation:

```
// VARS PM.GM.Menu, PM.GM.ToDo, PM.GM.Next
```

The actual initialization of these variables is done in the first part of the adaptation map:

- *Overlay presentation map variables* declaration: e.g., the showability variable for every concept in the lesson map is set initially to true for concepts without specific semantic labels, so they can be visualized by any type of user:

```
initialization ( // … other initializations
while GM.Concept.label = null
   ( PM.GM.Concept.show = true ) // … other initializations
)
```

- *Independent presentation map variables* declaration: e.g., the table of contents and 'To Do' list variables are set to false, so that a linear presentation results (a *tour*, in which the learner only needs to press the 'next' button to proceed through the course):

```
initialization ( // … other initializations
PM.GM.Menu = false
PM.GM.ToDo = false // … other initializations
)
```

During the interaction with the user (learner) some of the values of these variables can be dynamically changed (depending on the variable type), as in the following subsection.

2.4 Adaptation Maps

The adaptation model is the only part of the LAOS model describing the dynamics of the adaptation process. Static data exchange is covered also by standards such as SCORM, etc. Whilst the other layers are concerned with the static properties, variables and values required for personalization, the adaptation model brings them together and specifies how they will be used. Hence we can consider that the other sub-models in LAOS specify the ingredients of the personalization process, whilst the adaptation model specifies the recipe. The adaptation model itself is based on the LAG model, the model of three layers of granularity. LAG is a classification method for adaptive techniques as *direct adaptation rules*, *adaptation language* and *adaptation strategies*. The repartition on layers follows the increasing level of semantics. MOT is implementing all three levels, for increased compatibility and flexibility.

Direct adaptation rules build the assembly language of adaptation. These rules are the basis of every adaptive system, but are, just as every assembly language, system-specific. In MOT, an example of such a (generic) assembly level IF-THEN rule is, e.g., a rule about showing concepts with a weight above a certain threshold:

```
if GM.Concept.weight > 10
then  ( PM.GM.Concept.show = true )
```

The semantics of the rule above is not evident. It may mean, for instance, that the weight represents the *importance* of the respective concept, and therefore concepts with importance above 10 should be shown. This would imply that important concepts have a threshold above 10. Such a rule doesn't help the author who has to label the concepts with weights, and might consider important concepts with a threshold above 20, for instance. Also, the same rule might have a completely different semantics, such as the weight representing the *difficulty* of the concept. Then, the rule can be read as: show concepts with a difficulty above 10.

Moreover, the typical assembly rules are not *generic*, as the one above, but *specific* (i.e., they can only be applied to a given, specific concept in the domain or lesson map). In MOT, specific rules can be applied as follows. The example rule checks if the title of the specific concept called 'video presentation' has been accessed, and deduces then that the user is visual:

```
if '\Main topic\video presentation.title'.access == true
then  ( UM.GM.stereotype1 = vis )
```

The LAG *adaptation language* [12] is developed to increase the level of semantics of rules applied. Therefore, it allows, e.g., *repetitive* actions to be performed within a WHILE statement (as opposed to a list of IF-statements). For instance, a rule stating that, as long as there are non-labeled concepts still available, make them readable, is presented below:

```
while GM.Concept.label == null
  (  PM.GM.Concept.show = true )
```

The LAG language has also constructs inspired from games, such as the notion of *enough* conditions satisfied to proceed to a next level. For instance, a rule stating that, if the concepts are labelled as being for intermediate ('int'), and the user is an intermediate, then the concepts should be shown, is written as follows:

```
if enough(
    GM.Concept.label == int
    UM.GM.stereotype1 == int
    , 2 ) then
```

```
( PM.GM.Concept.show = true )
```

Within the 'enough' construct there are two conditions, and the number following shows how many of those have to be satisfied in order for the concept to be shown. As the number is 2, both conditions have to be satisfied. However, other combinations are possible with this construct.

Moreover, the LAG language borrows structural semantics from the domain or lesson maps. For instance, attributes such as the type or order of a given concept in the domain map can be addressed (although cannot yet be converted into AHA!), as follows:

- Type of Attributes (in domain maps) usage: e.g., selection of titles:
  `DM.Concept.Attribute.Type == Title`
- Order of Attributes (in lessons) usage: e.g., selection of the first lesson in a lesson container:
  `GM.Concept.Order == 1`

The LAG language has also other constructs that introduce structural semantics, such as *generalize* and *specialize* commands [12], which are not further detailed here.

The *adaptation strategies* build the actual *adaptation maps* and are written as LAG programs that can represent teaching or pedagogic strategies. A typical LAG program conform to the latest development consists of the following four parts:

```
// Description
// Variables
initialization ()
implementation ()
```

The *description* represents the semantic label of the strategy at the highest level of the LAG hierarchy.

In MOT, adaptation strategies can be reused in different contexts and for different combinations of domain and lesson maps, as required by the LAOS model. Therefore, the *variable declarations* can be used for checking the compatibility between domain, lesson and adaptation map. For instance, if a strategy uses a the label `GM.Concept.label` variable of the lessons to decide what to show to the user, it only makes sense to apply it to a lesson where these labels are not null and have the same domain (labels as in Figure 2, left frame). Moreover, if more than one adaptation strategy is applied to a domain, a possible clash can result from two strategies using the same variable. The *initialization* part is used for instantiating all the user map variables that are used by the strategy, as well as for initializing the presentation map variables (such as what will be shown to the user the first time he enters the system). As said, these two initialization types can be omitted if the user model is initialized as according to LAOS, in a separate interface, but for the current project they are initialized here. The implementation is used for the actual description of the strategy steps, to describe the interaction with the user (learner).

3. The Common Adaptation Format (CAF)

In order for the MOT static content to be converted to AHA!, an intermediary step was introduced, in the form of the conversion to a so-called 'common adaptation format' (CAF). This format represents the same elements as extant in the MOT mysql database, but in an XML language, which is more appropriate for web conversions, as well as semantically clearer for humans. Moreover, this representation of MOT is nearer to the LAOS XML Schema representation [1]. The idea is that with this format it will be easier to transfer between systems, as well as that it can be used as an intermediate language for a platform whereto and wherefrom systems can write and read complete adaptive hypermedia descriptions. Therefore, although we proceed via a 'converter between AEHs' approach in this research, the results can be used towards defining a 'lingua franca' for AEH description. CAF therefore represents a

proposal for the static component of the AEH description, whilst the LAG-language describes the system dynamics component. The CAF language DTD is presented in Figure 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT CAF (domainmodel?, goalmodel?)>

<!ELEMENT domainmodel (concept+)>
<!ELEMENT concept (name, attribute*, concept*)>
<!ELEMENT attribute (name, contents)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT contents (#PCDATA)>

<!ATTLIST contents
    weight CDATA ""
    label CDATA ""
>

<!ELEMENT goalmodel (lesson)>
<!ELEMENT lesson (contents*, lesson*)>
```

Figure 3. The CAF Language

The CAF format encodes *domain* and *lesson* maps. As the figure shows, domains need to encode at least one concept, which have lists of attributes and sub-concepts, just as in the MOT hierarchy (see also Figure 1). Attributes can have a name, representing the type of attribute, and the actual data they contain.

Figure 4 shows a domain map with at least one concept, named 'Adaptive', which has at least one sub-concept, named 'Adaptive Hypermedia'. The latter concept has at least one attribute, with the name 'Title' and the contents 'Adaptive Hypermedia'.

```
<CAF>
<domainmodel>
    <concept>
      <name>Adaptive</name>
      <concept>
        <name>Adaptive HyperMedia</name>
            <attribute>
              <name>title</name>
              <contents>Adaptive HyperMedia</contents>
            </attribute>
      ...
      </concept>
...
</domainmodel>
</CAF>
```

Figure 4. Domain Maps in CAF

Lesson maps are built of a hierarchy of sub-lessons, just as in MOT (see also Figure 2). Figure 5 shows an extract of a lesson description in CAF, with a high hierarchy lesson container with two sub-lessons, one pointing to the title attribute of the 'Adaptive Hypermedia' concept of the 'Adaptive' concept map, and the other one to the text attribute. This lesson container also contains at least one more lesson container. The content of the sub-lessons is actually a pointer to the corresponding concept attribute, just as in MOT. Sub-lessons can also have weights and labels, as shown in the figure.

Currently, CAF translates all contents in the MOT domain and lesson maps with the exception of the relatedness relations.

```
<CAF>
 ...
<goalmodel>
  <lesson>
    <contents weight="0" label="">Adaptive\Adaptive
HyperMedia\title</contents>
    <contents weight="0" label="">Adaptive\Adaptive HyperMedia\text</contents>
    <lesson>
       <contents weight="0" label="">Adaptive\Adaptive
              HyperMedia\Welcome\title</contents>
       <contents weight="0" label="vid">Adaptive\Adaptive
              HyperMedia\Welcome\video</contents>
    </lesson>
      ...
  </lesson>
<goalmodel>
</CAF>
```

Figure 5. Lesson Maps in CAF

4. AHA! as a delivery system for AEH created in MOT

AHA! is an open source adaptive hypermedia Web-based adaptive engine, based loosely on the AHAM model [20]. Information about AHA! (and the software) can be found at [1]. AHA! was created written in an '*assembly language*' of adaptive hypermedia, in the sense that any higher-level adaptation paradigm can be expressed in terms of AHA! and simulated by the AHA! engine [13]. AHA! offers lower level *direct adaptation rules* (as defined by the LAG model), based on the fact that, behind the diversity of ways of adaptation, there is a limited set of basic adaptation methods and techniques [5],[7].
The conversion engine from MOT to AHA! uses the CAF format domain and lesson map descriptions and the adaptive strategies written in LAG (user, presentation and adaptation maps) to create AHA! applications (adaptive presentations in AHA!).
Figure 6 presents an example of an AHA! application authored with MOT and delivered in AHA!. The central frame of the image presents a number of sub-lessons from the lesson map, belonging to the same lesson container.

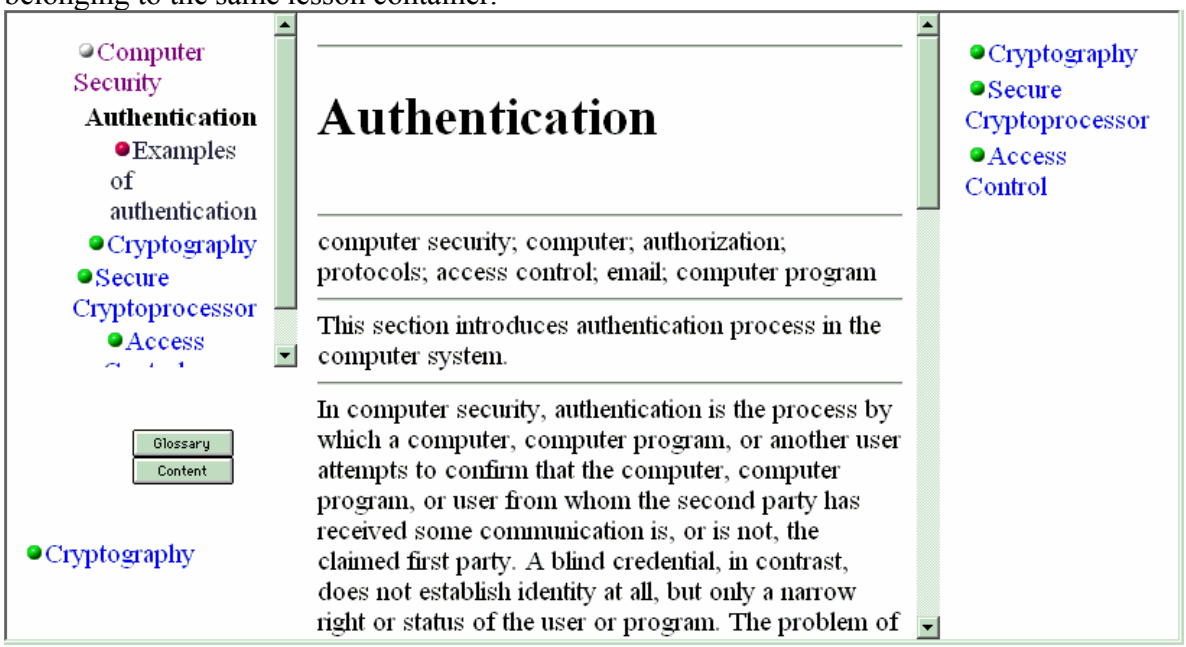


Figure 6. An adaptive application in AHA! authored with MOT

These sub-lessons are delimited with a separator line in-between them. The actual decision if to display the sub-lesson or not is dependent on the adaptation strategy contents, as described in section 2.4.

The left upper frame presents the 'Table of Contents'. This is adaptively annotated with green bullets for concepts that are ready to be visited, white bullets for already visited ones and red bullets for concepts for which the user is not ready yet, according to the standard AHA! color framework. Again, the decision about the color of the bullet in front of the respective link is determined by the adaptive strategy.
The right frame presents a list of 'To Do' (i.e., to read) concepts, that are recommended by AHA! based on the adaptive strategy.
The left lower frame presents only one concept, the 'Next' concept. This is the same as the first concept in the 'To Do' list.
The existence of these frames in the converted resulting AHA! application is also dependant on the settings in the adaptive strategy, as explained in section 2.4.

5. Discussion and Conclusions

This paper presents one of the first attempts to convert information between two different adaptive educational hypermedia systems: an authoring system and a delivery system. It is interesting to note that the two systems start from a different basic paradigm:

- MOT is based on LAOS and therefore stresses the notion of separation of concerns: domain, lesson, adaptation maps are authored separately, with great focus on explicit semantics. This notion is required by the 'authoring once, delivering many' authoring paradigm: if the adaptive educational hypermedia content has an intertwined representation of static and dynamic material, the reuse of this content will be difficult, if not impossible, in a different setting. Moreover, this separation of concerns allows authors to specialize in only one of the authoring targets (e.g., a domain specialist writing domain maps can be different from a pedagogy specialist, which creates adaptive pedagogic strategies), collaborating with each-other and reusing each-others materials.
- AHA! is based on the assembly-language approach. Moreover, in AHA! the statics and dynamics of the AEH are interlinked. This approach functions perfectly for delivery purposes, as it allows faster response to the user (student). However, reading the semantics out of the model, or reusing it in other contexts is not possible.

The process of conversion from MOT to AHA! has been evaluated with the help of a class of 4th year undergraduate students of Computer Science at the Eindhoven University of Technology. The students were following a course on Adaptive Hypermedia, and were asked to perform a project constructed based on this conversion process. The first results show that students unfamiliar with any of the two systems were able to produce materials via MOT, convert it and visualize it in AHA!. Students came with interesting alternatives for pedagogic labels for the material in the lesson maps, with their own variations of the adaptive strategies, and most of them created rich multi-media material presentations. The students replied to a set of questionnaires which are not discussed here due to lack of space.
It is important here to note that, as a result of the first experiments, some changes to the CAF language were performed. However, more seem to be required for the future, such as the use and expression of relatedness relations in MOT, for instance. Also, CAF needs to be extended according to LAOS model.
Extensions for the adaptation language are also being considered. Similarly to the extension required for the CAF language, the processing of relatedness relations is necessary, in order to be able to display, given a certain condition, for instance, the related concept content. Other extensions planned for the near future are the explicit differentiation between the display of a link to a certain content versus the display of the content itself. Also, parent-child relations within concept hierarchies have been introduced after these first tests have taken place, and are going to be extended with level information.

A change that occurred immediately after the first tests was the moving of the three systems (MOT, converter and AHA!) on-line. The next generation tests were performed with this new environment.
This paper represents yet another step to establish a common platform and a common format for the representation of adaptive educational hypermedia. From our first tests with different systems it is clear that this common format has to have a rich semantics.

Acknowledgements
This work is supported by the ADAPT project (101144-CP-1-2002-NL-MINERVA-MPP) and the PROLEARN network of excellence.

References

[1] AHA!. http://aha.win.tue.nl.

[2] Arteaga, C., Fabregat, R., Eyzaguirre, G., Merida, D. Adaptive Support for Collaborative and Individual Learning (ASCIL): Integrating AHA!and CLAROLINE, Adaptive Hypermedia conference (AH'04), Eindhoven, August 2004, The Netherlands, Springer, LNCS 3137, 279-282.

[3] Beaumont, I., and Brusilovsky, P. (1995) Adaptive educational hypermedia: From ideas to real systems. In H. Maurer (Eds.), Proceedings of ED-MEDIA'95 - World conference on educational multimedia and hypermedia, Graz, Austria, June 17-21, 1995. Charlottesville, AACE. - pp. 93-98.

[4] Blackboard, http://www.blackboard.com/

[5] Brusilovsky, P. Adaptive hypermedia. User Modeling and User Adapted Interaction, 11(1/2), (2001), 87-110.

[6] Brusilovsky, P., Eklund, J., and Schwarz, E. (1998) Web-based education for all: A tool for developing adaptive courseware. Computer Networks and ISDN Systems (Proceedings of Seventh International World Wide Web Conference, 14-18 April 1998) 30 (1-7), 291-300.

[7] Brusilovsky, P., Methods and techniques of adaptive hypermedia. In P. Brusilovsky and J. Vassilieva (eds.), User Modeling and User-Adapted Interaction 6 (2-3), Special Issue on Adaptive Hypertext and Hypermedia, 87-129.

[8] Claroline, Open Source e-learning, http://www.claroline.net/

[9] Cristea, A. I. & De Mooij, A. Adaptive Course Authoring: My Online Teacher. Proceedings of ICT'03, Papeete, French Polynesia, 2003.

[10] Cristea, A. What can the Semantic Web do for Adaptive Educational Hypermedia? International Peer-Reviewed On-line Journal "Educational Technology and Society" 7(4), Special Issue on Ontologies and the Semantic Web for E-learning, IEEE, LTSC, pp 40-58, 2004.

[11] Cristea, A., De Mooij, A. LAOS: Layered WWW AHS Authoring Model and its corresponding Algebraic Operators. In Proceedings of WWW'03, Alternate Education track. (Budapest, Hungary 20-24 May 2003). ACM.

[12] Cristea, A.I., and Calvi, L. The three Layers of Adaptation Granularity. UM'03. Springer.

[13] De Bra, P., Santic, T., Brusilovsky, P., AHA! meets Interbook, and more... Proceedings of the AACE ELearn 2003 Conference, Phoenix, Arizona, November 2003, pp. 57-64.

[14] De Bra, P. and Calvi, L., AHA! an open adaptive hypermedia architecture. The New Review of Hypermedia and Multimedia 4 (1998), 115-139.

[15] Felder, R.M. & Soloman, B.A. (2000). Learning styles and strategies.
http://www2.ncsu.edu/unity/lockers/users /f/felder/public/ILSdir/styles.html

[16] Moore, A., Brailsford T. J. & Stewart, C. D. (2001) Personally tailored teaching in WHURLE using conditional transclusion. Proc. of the twelfth ACM conference on Hypertext and Hypermedia, Denmark.

[17] MOT (My Online Teacher), http://wwwis.win.tue.nl/~acristea/mot.html

[18] Power, G., Davis, H.C., Cristea, A.I., Stewart, C. and Helen Ashman, H., Goal Oriented Personalisation with SCORM, ICALT'05, IEEE, July 5-8, 2005, Kaohsiung, Taiwan.

[19] Stewart, C., Cristea, A.I., Brailsford, T. and Ashman, H. (2005) 'Authoring once, Delivering many': Creating reusable Adaptive Courseware, WBE'05, February, 2005, Grindelwald, Switzerland.

[20] Wu, H. A. Reference Architecture for Adaptive Hypermedia Applications, doctoral thesis, Eindhoven University of Technology, The Netherlands, ISBN 90-386-0572-2, 2002.

Evaluation of Adaptive Course Construction Toolkit (ACCT)

Declan Dagger, Vincent P. Wade
Knowledge and Data Engineering Group, Department of Computer Science, Trinity College Dublin, Ireland
{Declan.Dagger, Vincent.Wade}@cs.tcd.ie
http://kdeg.cs.tcd.ie

Abstract. The ability to create pedagogically driven, activity oriented personalized eLearning is a key factor in the success of future eLearning. The ability to empower course developers with intuitive environments which provide pedagogic, subject matter, activity and personalization support during the process of creating a personalized course is fundamental to the adoption of adaptive educational hypermedia systems. The ability to support the output of standards conformant educational experiences is a key enabler in the integration of adaptive educational systems and services within existing educational infrastructures. This paper describes the evaluation of the Adaptive Course Construction Toolkit (ACCT) [3], an environment that supports the rapid composition of pedagogic and active personalized eLearning experiences.

Introduction

eLearning dropout rates in the last decade have been as high as 80% [5]. A key factor in the high failure rate of eLearning offerings is the lack of active learner engagement within the pedagogic process. Even with the introduction and adoption of rich multimedia content, the lack of learner engagement is still prominent. This is also evident in the area of adaptive educational hypermedia systems where the absence of solid pedagogic foundations in the educational experience is clearly evident [4]. A key factor attributed to these failure rates is the lack of pedagogic support when building an eLearning experience and also the absence of intuitive and user friendly tools in the area of personalised course composition. The Adaptive Course Construction Toolkit (ACCT) was developed to address these challenges. The ACCT provides a course developer support environment for designing, composing and deploying personalised eLearning experiences. More specifically it facilitates the representation and construction of subject matter knowledge through the Subject Matter Concept Space editor. It supports the course developer in creating personalised eLearning designs by providing pedagogical, activity, subject matter, personalisation and learning content scaffolding. More specifically it provides an environment for graphically building pedagogically-driven personalisable eLearning narratives using an extensible range of modelled information sources, namely pedagogical models, activity models, subject matter models, personalisation models and learning content search facilities. The ACCT provides an interface to the Adaptive Personalised eLearning Service (APeLS) [2], facilitating the rapid test and appraisal of the course developers personalised eLearning designs.

This paper describes the process and results of evaluating the ACCT. The paper identifies the goals and objectives of the ACCT evaluation. The paper then illustrates the

process of the evaluation. The results will be described in the evaluation analysis section. The conclusion section will describe how the goals and objectives where reached.

1. Goals and Objectives of Evaluation

The two core goals of this evaluation are to assess the usability of the Adaptive Course Construction Toolkit (ACCT) and to appraise the beneficial contribution of this research to the educational development process. Satisfying the first goal will provide insightful analysis about the course developer's satisfaction with the support environment of the ACCT, including the model support and the interface support. By satisfying the second goal of the evaluation will provide analytical information about how the ACCT will fit into the course composition process used by real educators. This will look at how the trial participants use the ACCT when composing their adaptive personalised courses. The goals and objectives will be observed through both a qualitative and quantitative process as illustrated in the evaluation strategy section of this paper.

1.1 Usability of the ACCT

According to [6] there are four suggested principles of good interface design. Firstly, the state and the action alternatives should be visible. Secondly, there should be a good conceptual model with a consistent system image. Thirdly, the interface should reveal good mappings that reveal the relationships between stages. Finally, the user should receive continuous feedback.

Based on these key principles the ACCT will be evaluated in terms of its

- informative feedback/responses to user interactions, such as file opening, action cancelled, drag and drop and error feedback dialogs.
- communicated conformance with elements of the Adaptive Course Construction Methodology
- identifiable cyclical approach to building an adaptive course
- terminology used by the ACCT, referring to Concept Space, Narrative Structures, Narrative Attributes and Learning Activities.

These are essential in given the user a sense of control, consistency and predictability using the interface of the ACCT.

Within the process of evaluating this research, the multiple modelled elements used in developing personalisation eLearning will be evaluated for their flexibility, reusability and accessibility. These main design elements are the Subject Matter Concept Space (SMCS), the Narrative Structures (modelled pedagogical guidelines), Narrative Attributes (modelled personalisation axes) and Learning Activities.

1.2 Integration with Existing Course Development Processes

As previously mentioned, one of the key barriers to the mainstream adoption of adaptive educational hypermedia systems, is the inherent difficulty of placing a person who is not an adaptive system engineer in an environment that allows them to build the "assembly language" required for adaptivity. Based on this, the trial participants were chosen from an educational background; instructional developers, subject matter experts and general educators.

The goal of this section of the evaluation is to understand and identify if both the adaptive course construction methodology and the ACCT would fit into an educators process of creating educational experiences while enhancing the produced educational experience with "personalisation". This will also aim to illustrate how the ACCT transparently disguises the technical difficulties inherent in authoring adaptive systems. This will lead to initial findings

into whether or not the people, who we need to be using these adaptive educational systems i.e. the educators, will actually adopt adaptive personalised educational systems as a tool in their day to day teaching.

2 Evaluation Strategy

In order to evaluate this research within the area of next generation personalised eLearning development tools a series of four personalised eLearning development workshops were held. The workshop participants, ranging from 4-25, varied from adaptive system engineers to instructional design experts to secondary-level school teachers. The general schedule of the workshops consisted of introductory presentations of personalised eLearning, followed by demonstrations of personalised eLearning systems and then a live demonstration of the Adaptive Course Construction Toolkit (ACCT). The interactive part of the workshop consisted of the participants carrying out a prescribed series of tasks such as creating a course package, importing and customising existing personalised eLearning models, building a customised activity-oriented pedagogical strategy, searching and selecting learning resources and publishing and testing their adaptive personalised eLearning course. These tasks would demonstrate the potential of the ACCT to trial participants and also provide them with an opportunity to thoroughly investigate all aspects relating to ACCT usability. Analytical data regarding the ACCT, the adaptive course construction methodology and the workshop organisation was collected through a series of qualitative and quantitative evaluations. The participants each completed a 42point Likert-type questionnaire with a variety of open and closed question types, covering aspects of their background skills, their ability to complete the given tasks using the ACCT, their interpretation of the usability of the ACCT and their satisfaction with the trial environment. The trial participants then participated in open discussion regarding all aspects of this research.

By implementing this evaluation strategy a series of steps would be followed:

1). Performance analysis of the trial participants with the set scripted tasks of the evaluation. The following tasks would to be completed during the trial (in the given order).

1. Course Package
    a) Load a course package
2. Subject Matter Concept Space
    a) View a Subject Matter Concept Space (Domain Ontology)
3. Narrative Design
    a) View Narrative
4. Course Package
    a) Create a custom course package
    b) Import information models from an existing course package
5. Subject Matter Concept Space
    a) Save a Subject Matter Concept Space (Domain Ontology)
    b) Edit a Subject Matter Concept Space (Domain Ontology)
    c) Export a Subject Matter Concept Space (Domain Ontology) as SVG
6. Narrative Design (Building a personalised eLearning design)
    a) Apply Narrative Structures (Pedagogical Strategies)
    b) Sequence Learning Activities
    c) Associate Subject Matter Concept Space
    d) Attach Narrative Attributes
    e) Save Narrative Model
7. Search for and Select Learning Resources
    a) Create a new Search
    b) Use Quick Search

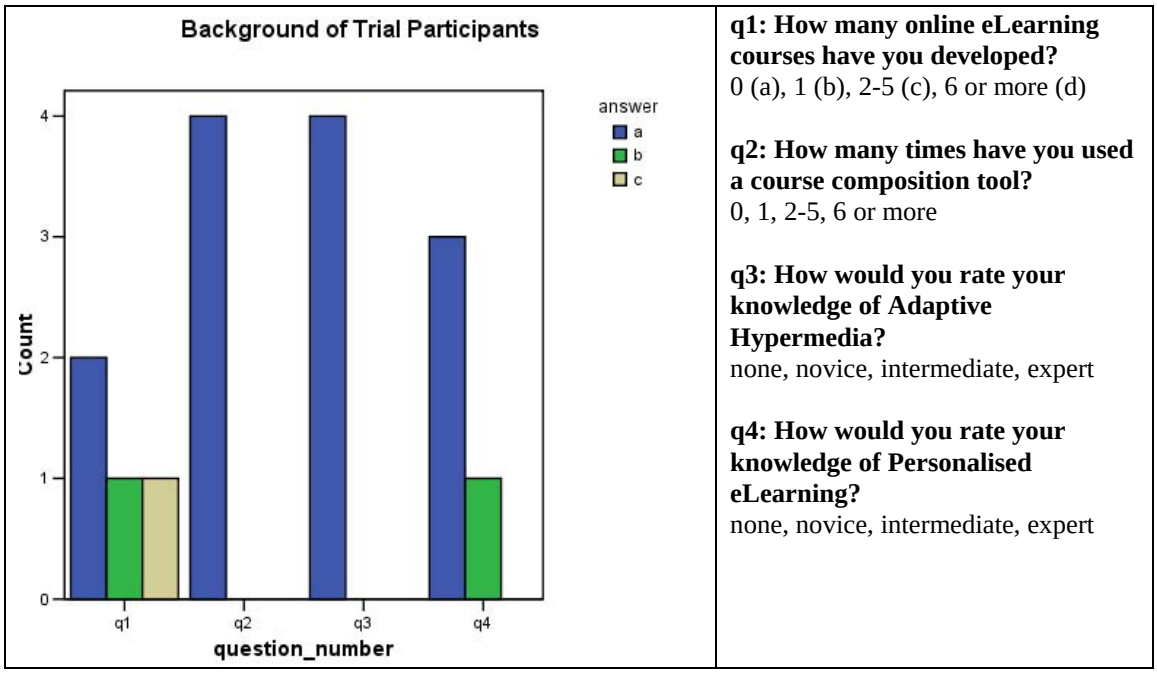
8. Publish Course Package
    a) Test Publication Connection
    b) Publish Course
9. Course Verification
    a) Build a sample instance of a Learner Model
    b) Run Adaptive Course against the Learner Model

2). Usability analysis of the ACCT and modelled personalised eLearning design elements. Based on the trial participant's completion of the above tasks, measures of usability were obtained through focused questionnaires.
3). Terminology analysis of ACCT and modelled personalised eLearning design elements. This aspect of the evaluation elicited qualitative and quantitative feedback from the trial participants regarding the terminologies involved.
4). A general discussion session at the end of each workshop was used to elicit any further comments, compliments and criticisms regarding the "realistic" adoption of personalised eLearning as a powerful and usable tool for educators, the methodology that forms the foundation of the ACCT, the modelled design elements of personalised eLearning and general design issues of the ACCT.

4 Evaluation Analysis

The evaluation analysis phase concentrated on three core components, namely the representation of disparate models used in the development of personalised eLearning, the usability of the ACCT development environment and finally the general usability of personalised eLearning in educational settings. The evaluation presented in this paper is the most recent evaluation of the ACCT carried out with 4 subject matter experts, instructional designers and educators from Intel Ireland and Skoool.ie.

Figure 1) Results of Background Questions



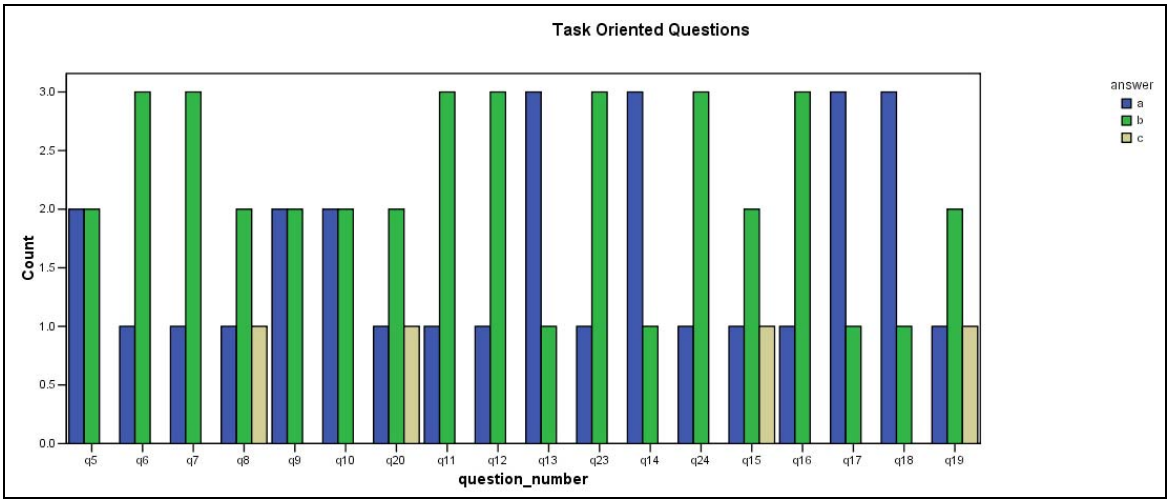
As illustrated in figure 1 above, the majority of trial participants, although general educators, have little experience in creating eLearning courses or using course composition tools. It can also be noted that these participants were not adaptive systems engineers or personalised

eLearning developers. This illustrates that the base of the trial participants was solidly rooted in traditional educational paradigms.

4.1 Personalised eLearning Development Models and the Adaptive Course Construction Methodology

The unique approach used by the ACCT expands the extensive research carried out by the knowledge and data engineering group at the department of computer science in Trinity College Dublin. Namely, it extends the flexibility of the multi-model meta-data driven approach to adaptive systems engineering [1]. Extensions in pedagogical, subject matter area, personalisation axes and learning activity modelling were made to increase flexibility and reusability of these disparate information sources for personalised eLearning development. The ability to compose adaptive personalised courses from these information sources based on the adaptive course construction methodology using the ACCT has proven very successful.

Figure 2) Results of Task-Oriented Questions



The questions illustrated in figure 2 aimed at addressing the user's ability to perform the specified tasks of the trial. As a general analogy, "a" type answers reflect a very positive response to the ACCT's ability to support the performance of a task and "d" type answers reflect a very negative response to the ACCT's ability to support the performance of a task. From the distribution in the graph, it can be claimed that in general, the ACCT can positively support the performance of the multiple tasks of personalised eLearning composition. This part of the evaluation also consisted of several open type questions where participant's views and comments were elicited.

From a technical point of view, the standards independent representation of the disparate design elements functionally promotes and supports their reuse on a number of different platforms and environments. For example, the pedagogical and learning activity models can be represented in IMS Learning Design, the subject matter concept space can be represented in OWL and the personalised eLearning narrative can be represented in IMS Learning Design level B.

From an educational aspect, the participants felt that the separated models provided an insightful view of the roles played by the different information sources in the composition of and also the execution of the personalised eLearning designs. More specifically, questions 19 and 20 addressed the participants understanding of the roles of the subject matter representation and the personalised axes, respectively, in the creation of personalised eLearning experiences. Through the course publication mechanism, question 18, the course

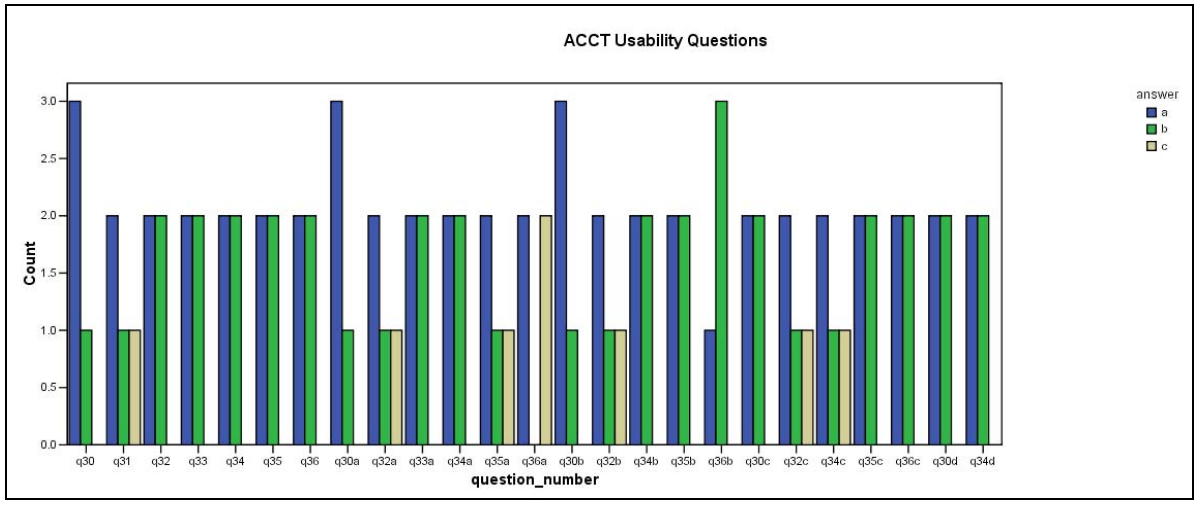
developers could see and interact with the courses that they created, in real time, and then use the ACCT to edit and modify the models and republish their courses. This facility was of key importance in the participant's realisation of the roles played by the disparate models in a personalised eLearning experience.

The participants complemented the graphical representation of the subject matter concept space (in the open questions), commenting that the visualisation of subject matter concepts and their interrelationships formed a very intuitive and real representation of subject matter areas. By keeping the subject matter concept space independent of content, the participants identified the flexibility of this approach in facilitating and promoting the reuse of this knowledge. Based on open discussion, it was identified that the trial participants felt that the workspace for building the personalised eLearning design (Narrative) was very intuitive and supportive. The supporting models of pedagogy, learning activities, subject matter and personalisation axes made the task of composing the adaptive course easy to perform. This flexibility supports the rapid prototyping of personalised eLearning course structures. The ability to interact with and select content from multiple remote learning resource repositories facilitated the instantiation of the realistic personalised eLearning experiences.

4.2 Usability of the Adaptive Course Construction Toolkit (ACCT)

The interface of the ACCT is an easy to use drag and drop composition environment supporting the course developer in building personalised eLearning experiences. Based on the responses of the trial participants, the ACCT interface is intuitive, easy to use and very supportive in feedback and closure notifications when actions are performed.

Figure 3) Results of Usability Questions



The usability questions illustrated in figure 3, addressed the ACCT's use of terminology, messages and informative feedback, error prevention and error correction, predictability and reliability from a completely technical perspective. As the distribution of the graph depicts, the trial participant's responses to the general usability of the ACCT were typically positive. For example, question 32 relates to the informative feedback (through dialogs and systems responses) provided by the ACCT and question 33a relates to the predictability of performing a system operation.

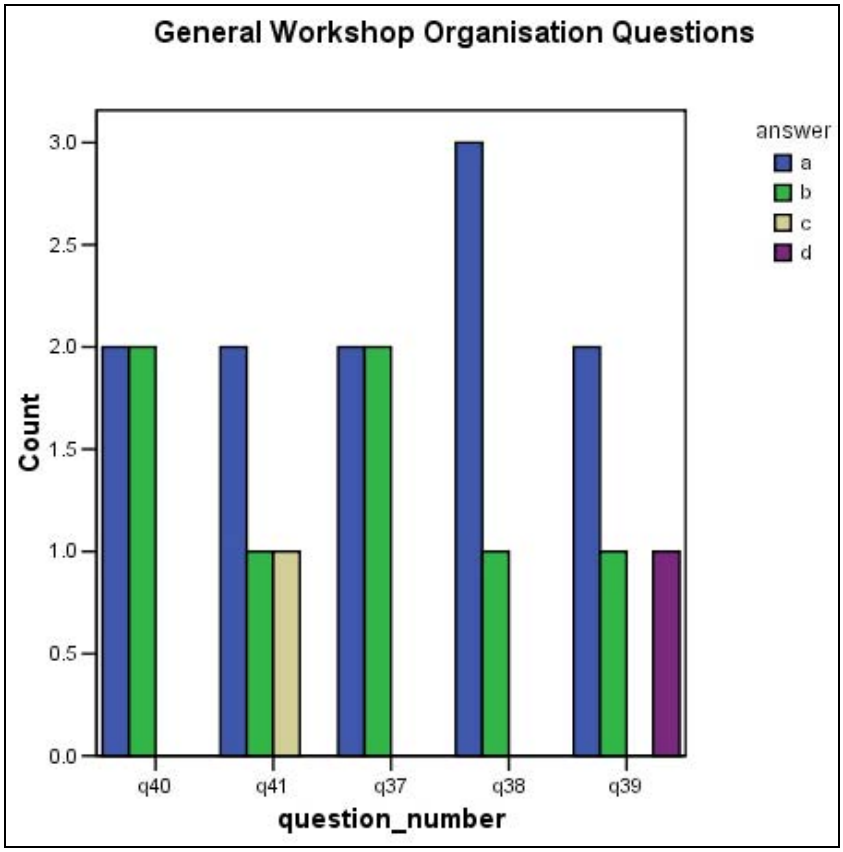
However, some higher-level usability issues with the ACCT were identified at this stage of the evaluation. For example, the flat list representation of subject matter concepts available during the composition of the adaptive course does not effectively visualise the graphical nature of the relationships that exist within the concept space. This inadequate

visualisation of the subject matter concept space slowed down the course composition process, according to comments from the trial participants. Based on suggestions from the trial participants, the visualisation of instantiated narrative, i.e. narrative with content, was confusing since the associated learning resources where not displayed in the graphical view. Having to right click on a concept and select "view candidate resources" slowed down the course composition process. These and other usability issues are currently being addressed.

4.3 General Usability of Personalised eLearning

Although the beneficial contributions, to day to day education, offered by personalised eLearning was realised by the trial participants, one of the key findings of this evaluation and the primary obstacle identified by the trial participants involved the terminology of adaptive educational systems, as illustrated by question 39 in figure 4. This became very clear from both the open questions of the evaluation and the general discussions of the workshops. The educators who we need to be using these systems are not familiar with the terminology and notation of adaptive hypermedia or the semantic web. They therefore do not initially understand what terms such as ontology, narrative, adaptive axes, subject matter concept space actually mean. This was one of the main criticisms from trial participants; the learning curve involved with identifying and understanding the meaning behind the terminology of personalised eLearning was quite steep.

Figure 4) Results of General Workshop Questions



5 Conclusions

Current research focus will develop mechanisms to support exporting standardised representations of the disparate information models specified by this research. We are building a plug-in to facilitate the ACCT in producing IMS Simple Sequencing and IMS

Learning Design so that personalised eLearning experiences developed with the ACCT can run on standards conformant Learning Management Systems (LMS). A residual affect of this evaluation is the pioneering of a fundamental requirements specification phase for effective personalised eLearning development environments (PEDE).

Based on the evaluation of this research it has been identified and proven that with an environment such as the ACCT it is possible for technical and more importantly non technical course developers to use, reuse and repurpose the disparate models of personalised eLearning in order to compose adaptive personalised eLearning experiences. The evaluation proved that non technical course developers can understand how the disparate models are used in concert to produce personalised eLearning experiences without having to understand the underlying technologies and representation languages. Based on the enthusiasm of the trial participants it indicates an appetite for developing and using personalised eLearning in their day to day teaching.

References

[1] Conlan, O., Wade, V., Bruen, C., Gargan, M. (2002) *Multi-Model, Metadata Driven Approach to Adaptive Hypermedia Services for Personalized eLearning*, Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, Spain, May 2002.

[2] Conlan, O., Wade, V. (2004) "Evaluation of APeLS - An Adaptive eLearning Service based on the Multi-model, Metadata-driven Approach", Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2004) Proceedings, Eindhoven, The Netherlands (2004)

[3] Dagger, D., Wade, V., Conlan, O., (2004), *Developing Active Learning Experiences for Adaptive Personalised eLearning*, Adaptive Hypermedia and Adaptive Web-Based Systems, AH2004,

[4] De Bra, P. , Aerts, A. , Smits, D. , Stash, N. (2002), *AHA! meets AHAM*. Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, May 2002. Springer LNCS 2347, pp. 381-384.

[5] Forrester, (2000) *Online Training Needs A New Course*, ©2000 Forrester Research, Inc.

[6] Norman, K., (1991) *Models and the mind and machine: Information Flow and Control between humans and computers*, Advances in Computers, 32, p 119-172

Evaluation of Interoperability of Adaptive Hypermedia Systems: testing the MOT to WHURLE conversion in a classroom setting

Alexandra CRISTEA[1], Craig STEWART[2], Tim BRAILSFORD[2] and Paul CRISTEA[3]

[1] *Information System Department, Faculty of Mathematics and Computing Science, Technical University Eindhoven, Den Dolech 2, PO Box 513, 5600 MB, Eindhoven, The Netherlands, a.i.cristea@tue.nl*

[2] *School of Computer Science and Information Technology, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham, NG8 1BB, UK,{cds,tjb}@cs.nott.ac.uk*

[3] *Digital Signal Processing Laboratory, "Politehnica" University of Bucharest, Spl. Independentei 313, Bucharest 77206, Romania, pcristea@dsp.pub.ro*

Abstract. The creation process of adaptive hypermedia is rarely evaluated. Moreover, conversion between different adaptive hypermedia systems has barely been proposed, yet alone tested in realistic settings. This paper presents the evaluation of the interoperability of two adaptive (educational) hypermedia systems, MOT and WHURLE, the one serving as authoring system, and the other as delivery system. The evaluation is performed with the help of a class of thirty-one students enrolled in the fourth year of the "Politehnica" Unversity of Bucharest, who were taking a one-week intensive course on Adaptive Hypermedia. This paper describes and interprets our first experiments of the "write once, deliver many" paradigm of adaptive hypermedia creation.

Introduction

In recent years Adaptive Hypermedia has arisen as a response to the almost ubiquitous dominance of the 'one size fits all' approach to hypermedia on the Web. Whether a person is surfing the web for a product (as a customer), medical information (as a patient), educational material (as a student) or information about governmental bureaucracy (as a citizen), web sites usually offer the same content to each visitor. Thus a person has to search through potentially large amounts of material much of which is often irrelevant to him or her.

Adaptive Educational Hypermedia (AEH)[4] deals with the issue of providing a personalized educational experience. Rather than a learner having to read through every piece of content whether or not it is appropriate, an AEH system will adapt its presentation of content to the learner's needs – this adaptation being informed by a User Model.

However the *creation* of adaptive content can be a complex and time consuming task. Imagine an AEH system that adapts around the learners Visual::Verbal Learning Style [8]. At it's simplest an author would have to create two instances of the same lesson, one

for those students who learn more effectively from primarily visual information and one for those who are more inclined to textual or verbal learning. As most AEH systems adapt to many more learner characteristics than this, it is easy to understand why, notwithstanding the global stride for customization, AEH have not yet been widely adopted in educational institutions.

One of the issues that aggravate the problem of AEH authoring is the fact that often each AEH has its own authoring system. Materials authored for a given system are only viewable within that system. As AEH systems undergo research and improvement [4], it can be almost impossible for a non-technical author to stay up to date. Also the system they are currently using may loose its support and development team, for example, as academic project funds dry up or the team moves onto new areas of research.

In response to this we have proposed a new paradigm for AEH authoring, '*write once, use many*' [11], whereby an author only has to learn to author for a single AEH system, and the materials from this system can be converted into any other AEH system. This is of course only a single step towards a greater change: ideally all systems should be interoperable using a 'many to many' methodology.

So far we have used the AEH system MOT as an authoring system for three other Educational Hypermedia systems, AHA! [5], WHURLE [10] and the commercial system Blackboard [2]. It is hoped that each conversion will reveal the fundamentals required for a more generic conversion system. However each conversion system needs to be tested with the audience it has been designed for, the AEH authors themselves, as the aim of this work is not only to reduce author load but to ensure that the authoring process itself is as easy and trouble free as possible.

In this paper we describe an experiment in which we examine the authoring process. A class of 31 students has been introduced to authoring for MOT & WHURLE in an intensive short course; their progress and responses to this new methodology were recorded in a series of questionnaires.

The remainder of this paper is structured as follows: Sections 2 & 3 introduce the two AEH systems involved (MOT as an authoring environment and WHURLE as the delivery environment), with Section 4 briefly describing the conversion system [11]. Section 5 details the experimental settings, and Section 6 the hypotheses we evaluated. Section 7 presents the results. Finally we conclude and draw inferences for future work in this area in Section 8.

1. MOT Presentation

MOT [6] is a web-based generic adaptive hypermedia authoring system based on the LAOS framework [5]. For the purpose of our current paper, this means that MOT allows the creation of *domain concept maps* (DM), containing the actual resources clustered as content alternatives, and the creation of *lessons* (GM), based on these domain maps, that allow a restructuring and filtering of the contents. These contents are stored in a MySQL database, which means that MOT's adaptation is based upon the queries sent to the database from the MOT delivery engine.

One of the interesting features of the *lesson* layer, which was of use to the students during the tests presented in this paper, is the functionality of pedagogical labelling of previous concept (attributes) from the *domain* maps. As Figure 1 shows, attributes

(containing concept alternatives, such as text, figure, etc.) can be pedagogically labelled (e.g., the figure attribute is labelled 'vis' for visual, and the text attribute is labelled 'ver' for verbal, as according to the ILS learning style questionnaire [8]).

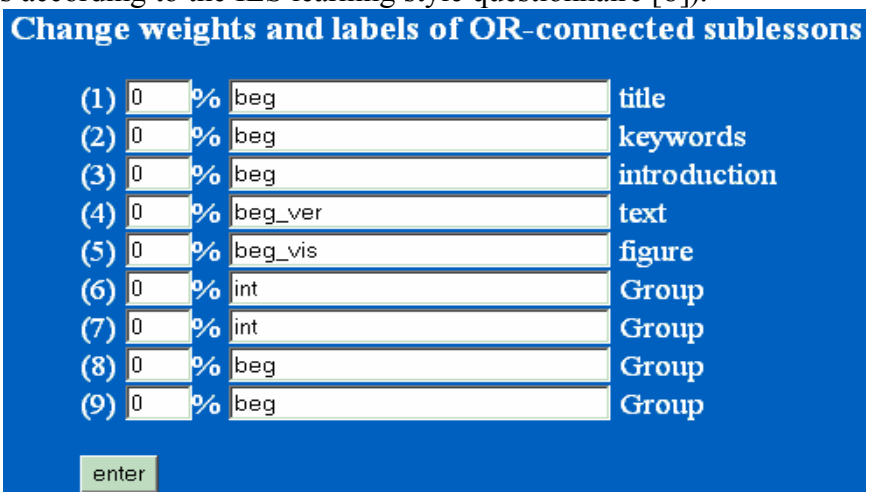


Figure 1: Weights and labels for the attributes of a MOT concept

MOT was the content creation environment used by the students. For more information on MOT, refer to [6].

2. WHURLE Presentation

WHURLE is an XML based, on-line integrated learning environment, which is designed to deliver content that is personalised to the needs of the learner. The learner is presented with a lesson, which is constructed from a collection of underlying educational resources. The basic lesson structure is defined in a *Lesson Plan* and filtered according to rules specified in the WHURLE user model [9].

The underlying content objects in WHURLE are resources called *chunks*. Each chunk is a single text file describing a conceptually discrete piece of information with no links to other resources (such as external web pages), written in the WCML (WHURLE Chunk Markup Language, an XML application). Owing to the flexibility provided by WHURLE's use of chunks (through the conditional transclusion of chunks appropriate to each learner [10]), adaptation may be implemented at the content level to determine which chunks are made available to each class (group or 'stereotype') of learner.

Authoring materials for use in WHURLE is a time consuming task. Authors write material using standard XML editing tools (facilitated by preview stylesheets) – a daunting task for authors not well versed in XML.

3. MOT to WHURLE conversion

Authoring materials in MOT (section 2) and delivering these materials in WHURLE are both simple tasks. In the former an intuitive web-form based process is used to create and order materials; with the later, the learner only has to register for a given lesson and the relevant materials will be adapted for a personalized delivery.

The conversion itself is currently slightly more complex than either of these processes. Initially we must understand the similarities between the two systems. The GM (Goal and Constraints Model, describes *lessons*) used by MOT is a hierarchical structure organised by 'concept'. The Lesson Plan used in WHURLE is also a hierarchical structure, organised by 'level'. Each MOT concept has 'attributes' and each WHURLE level has 'chunks' (collected into a 'page') that define the actual content. From this basic description we can begin to derive an initial conceptual mapping of MOT to WHURLE. MOT has several default standard attributes, of which 'title' and 'keywords' are common to WHURLE chunks. Therefore, in any conversion process it is necessary that these common elements are included in every chunk created.

The conversion system uses the MOT 'weights' (shown in Figure 1) to map groups of MOT concept-attributes to a single WHURLE Chunk. Thus initially the correct weights have to be applied to each MOT concept-attribute. It is by using these that the MOT to WHURLE converter can identify which concepts are to be delivered to a given learner. Whilst there are a range of weights and labels that can be used during the conversion the students in this exercise were told to use the weights and labels in Table 1 to keep the authoring process straightforward. NOTE: a Concept Weight value of '0' defines the 'common' elements (such as 'title' and 'keywords'), and as such defines which elements will be available to everyone.

Table 1: MOT weights & labels (Figure 1) used to identify which concepts would be appropriate for learners with a given preference for the ILS learning style [8].

MOT Concept Weight	MOT Concept Label	WHURLE learner group
0	n/a	Everyone
35	Vis	Visual preference
75	Ver	Textual preference

Using these weights and labels to group concepts for similar types of learners, the conversion system maps the MOT structure into a similar structure in WHURLE. Each of MOTs lesson concept groups are associated and transformed into WHURLE chunks, with each chunk being appropriate for a given learning preference. Table 2 shows how a series of MOT concepts (as described in Figure 1) can be associated using their weights & labels to create WHURLE chunks.

Table 2: Creation of WHURLE chunks from MOT concepts.

MOT components			WHURLE components		
Attribute	*Weight*	*Label*	*'everybody' chunk*	*Visual prefs chunk*	*Textual prefs chunk*
Title	0		✓	✓	✓
Keywords	0		✓	✓	✓
Introduction	0		✓	✓	✓
Text	75	Ver			✓
Conclusion	0		✓	✓	✓
Figure	35	Vis		✓	
		Chunks:	C1	C2	C3

Table 2 shows how using the concept labels as described in Figure 1, the conversion process will create three WHURLE chunks (C1, C2 & C3) by associating: all of the weights of '0' (C1), associating all of the weights of '0' and '35' (C2) and associating all of

the weights of '0' and '75' (C3). Note that the 'common' elements are placed in every chunk, therefore C2 and C3 contain the same elements as C1 – as '0' weight elements are common to every chunk being created. Once these weights and labels have been established, the conversion program will automatically convert the MOT lesson into a WHURLE LP with chunks, and register the LP within the WHURLE mysql database. In this way, for each concept, three types of possible display will result (Table 2): a chunk to be seen by all students ('everybody' chunk), one for students with visual preferences (visual prefs chunk), and one for students with textual preferences (textual prefs chunk). More details on this conversion process can be found at [11].

4. Experimental Settings

A class of 31 students, in the 4th year of study for a technical Masters degree at the University of Bucharest, Romania, was required to attend an intensive week long course on Adaptive Hypermedia, within a Socrates mobility exchange course. The students were supposed to have a combined course of theory and hands-on experience. The week started with two half day lectures, whereby the basics of the subject were introduced, before moving onto discuss the specifics of the used systems. Specifically, students:

- followed the lectures on Adaptive Hypermedia, Learning Styles, LAOS, MOT, WHURLE, MOT to WHURLE conversion;
- performed the assignment attached to this course (authoring with MOT, converting into WHURLE, visualizing & analyzing in WHURLE).

The assignment was performed by breaking the class down into six groups (of 5-6 students); in the last three days of the course each group was asked to:

1. Create 2-3 MOT Domain Concept Maps, with approximately 5-10 concepts on the http://e-learning.dsp.pub.ro/mot/ MOT server
2. The attributes of each concept were: title; keywords; introduction; text; conclusion and figure. With limits placed on the type and amount of content in each one (this was done so as that each group would not spend their limited time creating a vast corpus of information).
3. Create a single MOT Lesson (Goal & Constraints Map) using their Concepts maps.
4. Alter the lesson so that the weights and labels of each concept agreed with those described in Table 2.
5. Run the 'mot2whurle' conversion program and copy the files to WHURLE.
6. Check that the WHURLE XML files are well-formed.
7. Run and login to WHURLE to check that the lesson matches their design and make any necessary changes.
8. Finally at the end of the week, each student was asked to complete a series of questionnaires: three generic SUS (System Usability Scale [3]) questionnaires, one for each system (MOT, mot2whurle and WHURLE) and a single specific questionnaire designed to determine their level of knowledge about each system, as well as to gather non-statistical information.

The students were told from the very beginning that their response to the questionnaires will not affect their mark. In fact, the marks were given to them before the

questionnaires were processed. Further details of the task each group undertook can be found at: http://wwwis.win.tue.nl/~acristea/AH-Ro/
During the course the students had access to support mechanisms, in the form of the course moderator (either in person or via email), and their own peer support mechanism within each group.

5. Hypotheses and Evaluation Goals

We have decided to evaluate different aspects of the experimental setup as follows. We wanted some generic information about the students' experience with MOT and WHURLE, although our experimental focus was on the conversion program, MOT2WHURLE, which has never been tested before. The reason why we also tested MOT and WHURLE separately is, on one hand, the possibility of letting the students express their opinions fully and unrestrictedly about all the separate parts of the experiment, and also because WHURLE was not previously evaluated via a SUS questionnaire.

For pure usability we used the SUS questionnaire three times. SUS is a simple yet flexible usability scale consisting of 10 questions, ideal for the generic assessment of a system's usability. The results of each question can not be considered on its own but should be summed to form a final SUS Score (from 0 – 100), the higher this score the more 'usable' the system.

However, the SUS questionnaire did not answer some specific issues we wanted treated, and therefore we also added a generic questionnaire on these issues, such as the level of student understanding, the type of tasks performed, their specific difficulties, etc.
Some of the hypotheses that we checked with the help of the experiments (student work and questionnaires) are as follows:

1. *The systems (MOT, mot2whurle, WHURLE) are simple and intuitive to use, with a minimum amount of explanation.*
2. *The students understood the theoretical background (Adaptive Hypermedia, LAOS, Adaptive Strategies) of these systems.*
3. *The students understood the connection between LAOS and MOT.*
4. *The students used MOT purely for authoring adaptive hypermedia, and perceived it as such.*
5. *The students used WHURLE solely for delivering adaptive hypermedia, and perceived it as such.*
6. *Students consider automatic conversion between one-to-many or many-to-many adaptive hypermedia systems useful.*

Beside these hypotheses, the aim of the evaluation and testing was to gain information for further development of the conversion system in particular, and the other two systems.

6. Experimental Results

First let us analyze the numerical results, from the point of view of validating or refuting our hypotheses. Figure 2, Figure 3 and Figure 4 show the average SUS results in the form of a radar chart. As a simple interpretation, the more the chart resembles a star, the greater the students' conviction that the system is readily usable. Following each chart are the actual SUS scores.

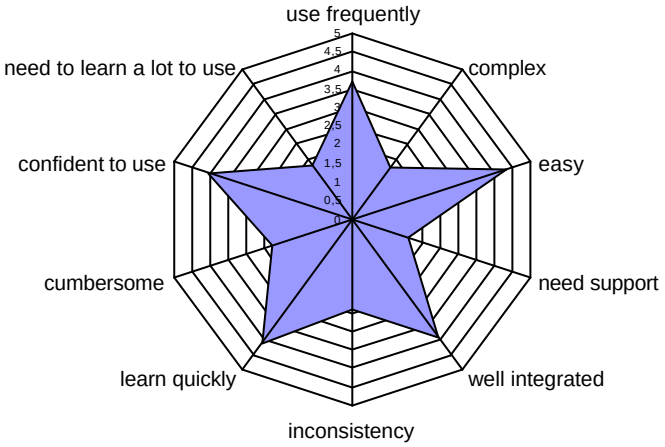


Figure 2: MOT SUS results.

The SUS score for the MOT usability is 75%. There were 29 students answering, with an average variance of their overall usability estimation of 15%.

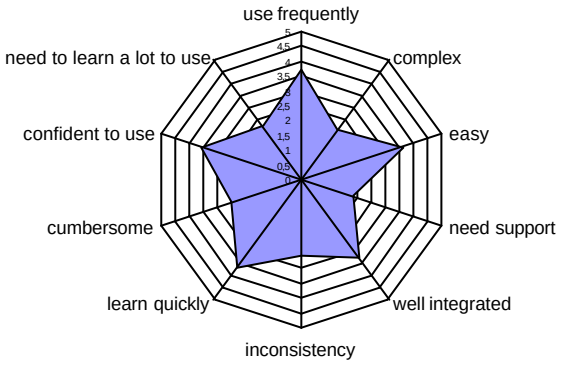


Figure 3: WHURLE SUS results.

The SUS score for the WHURLE usability is 66.6%. There were 28 students answering, with an average variance of their overall usability estimation of 19.1%.

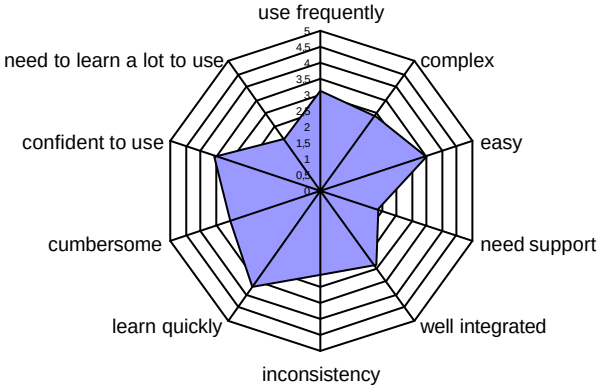


Figure 4: MOT2WHURLE SUS results.

The SUS score for the MOT2WHURLE usability is 60.7%. There were 29 students answering, with an average variance of their overall usability estimation of 19%.

The scores above sustain to some degree hypothesis 1 with empirical numerical data. We can also draw the conclusion that, as the author-centred MOT received the highest SUS score (75%), we have chosen a comparatively simple and 'usable' environment for authors to create adaptive materials within. WHURLE received a slightly lower SUS score (67%), which supports our view that using MOT as an authoring environment is advantageous for the author concerned. Finally the MOT2WHURLE conversion system had the lowest SUS score (61%) and as such requires, comparatively, the most work to ensure ease of use for AEH authors.

Hypothesis 1 is also sustained by the qualitative fact that the students were able, with minimal explanation, during a one-week period, to produce content with MOT, convert and visualize it in WHURLE.

Hypotheses 2 to 5 have to be extracted from the extended, specific questionnaire. Hypothesis 4 is validated by a vast majority of 25 students out of 29 selecting MOT to be an adaptive hypermedia *authoring* system (one student classified it as an adaptive hypermedia system, and the rest did not answer). Hypothesis 5 is validated by 21 students selecting WHURLE to be an adaptive hypermedia system.

However, the hypotheses 2 (average of about 70% with a dispersion over 24%) and 3 (average of 70% with 21,7% dispersion) are not validated conclusively. It seems that students were much more confident in their system usage, than in their comprehension of the theory behind it. Although this might be unfortunate from academic point of view, from the point of view of system testing, this means that the systems could be used even with a vague understanding of the theoretical background.

Hypothesis 6 is confirmed by the students' reply to the generic questionnaire. The students' replies were very positive to this question, with an average of 4.57 (out of 5), and variance of 0.77.

Furthermore, from the specific questionnaire we were able to gather some points of possible improvements of the systems that the students were asked to use, as free-style textual answers. By far the greatest number of suggestions and comments were made for the MOT2WHURLE conversion program, for example:

Question: "What are the major difficulties that you encountered when working with the MOT to WHURLE conversion program, in your opinion?"

Sample Answers:

- "The major difficulty encountered was that we had to redo several times the conversion as we didin't respect the maximum no. of concepts and subconcepts."
- "Interface" (3 students)
- "We had to correct some parts of the *.wcml files created through conversion (closing/opening tags)"
- "None, just that there should have been a single tool fetching all needed files, launching the converter, and then uploading the changes."
- "The structure is not being kept as in the lesson."

Comments: The majority of these responses alluded to the problems with the conversion interface – indeed as the conversion program is an offline, command line system, the requirement for a more advanced interface was already suspected. From these comments it

would seem that integrating the conversion system into MOT as an online process would be desirable. The comment addressing the correction of the '*.wcml' files alludes to a bug in the conversion program which is being addressed. There were also a few comments concerning MOT, however these were matters of additional functionality rather than fundamental operating principles (as suggested for the MOT2WHURLE program) – as would be expected from its comparatively higher SUS score.

Finally we asked our novice authors to suggest any other adaptation strategies that they would like to author in MOT and see used in WHURLE. They answered that "… sequential versus global adaptation strategy would seem interesting for me." and 'Some combinations of visual and audio would make it more attractive". As the MOT system can author pedagogically flexible materials and WHURLE can deliver different types of adaptation (by use of different adaptation filters), these two suggestions could easily be implemented using these systems.

7. Discussion & Conclusion

This paper is, according to our knowledge, the first attempt to *empirically* test the conversion process between two completely separate adaptive hypermedia systems, by using one for authoring and the other for delivery of AEH materials. We have gathered data about the process, some of it validating our efforts into:

- creating a more flexible authoring environment, such as MOT
- creating a conversion system between this environment and others, such as WHURLE
- interfacing adaptive hypermedia systems
- striving towards a 'write once, deliver many' paradigm.

The comments gathered from our test authors, confirm our primary and most important hypothesis that *using a single authoring platform to write materials for multiple delivery systems* is indeed greatly desired by authors of AEH contents.

Author comments however also indicate the great deal of additional work required to improve the user interface and connection between these systems, with many areas for improvement (or areas of misunderstanding) being highlighted. For future work, we are going to integrate the students' comments into improving the MOT2WHURLE conversion. In the meantime, MOT2AHA was also tested, so it will be interesting to extract commonalities between these processes.

Another important lesson learned from our experiments is that to create an AEH interoperability tool there must be either

- a significant degree of *commonality* between systems, or
- a significant degree of *generality* in the authoring system.

The conversion between MOT and WHURLE alone does seem to imply that some similarity does exist. Both systems were developed entirely independently, and yet they employ a similar conceptual approach in their development. Current work upon MOT to AHA! and MOT to Blackboard conversion tools suggests however a degree of generality in MOT that extends beyond the limits of MOT & WHURLE similarity. Hence the goal of a 'write once, use many' authoring and delivery system interoperability would seem eminently feasible, starting with a MOT-like common description of authoring content and

dynamics. Of course, to have any hope of achieving this goal on the long term, the interoperability process and/or common description would require *standardisation*. At the moment there are standards for the structure of learning data (e.g. LOM [14] for metadata), however there are none for the dynamic aspects of an adaptive system. IMS LD [12]and IMS SS [13] both fall short of being able to fully describe the flexibility of an AEH.IMS LD is not a hypermedia design model, but a high-level framework for educational activities specification, therefore not dealing with specifics of adaptive hypermedia (reordering, hyperlinks, etc.). IMS SS does not deal with adaptive content or adaptive presentation, and its adaptive navigation model uses preconceived manifests. Therefore an author *has* to describe every outcome in the manifest, they can not automatically generate any lesson their learners require at run time – unlike most adaptive systems.

8. Acknowledgements

This work is supported by the Minerva Socrates project ADAPT [1] (101144-CP-1-2002-NL-MINERVA-MPP), and the Socrates mobility exchange program. SUS was developed as part of the usability engineering program in integrated office systems development at Digital Equipment Co Ltd., Reading, United Kingdom.

References

[1] ADAPT. 2004. http://wwwis.win.tue.nl/~acristea/HTML/Minerva/index.html
[2] Blackboard Inc, *Blackboard*, retrieved March 7, 2005 from http://www.blackboard.com/
[3] J. Brooke, *SUS - A quick and dirty usability scale*, In Jordan, P.W.; Thomas, B.; Weerdmeeester, B.A. * McClelland, I.L. (eds.), Usability Evaluation in Industry, pp 189-194. London, UK: Taylor & Francis, 1996.
[4] P Brusilovsky, Adaptive hypermedia, *User Modeling and User Adapted Interaction,* Ten Year Anniversary Issue (Alfred Kobsa, ed.) *11*(1/2), pp87-110, 2002.
[5] A. Cristea, & A. De Mooij. LAOS: Layered WWW AHS Authoring Model and its corresponding Algebraic Operators. In ACM Proceedings of WWW'03, Alternate Education track. Budapest, Hungary 20-24 May 2003,.
[6] A.I. Cristea,. & A. De Mooij. Adaptive Course Authoring: My Online Teacher. Proceedings of ICT'03, Papeete, French Polynesia, 2003.
[7] P. De Bra, A. Aerts, B. Berden, B. De Lange, B. Rousseau T. Santic D., Smits & N. Stash, AHA! The Adaptive Hypermedia Architecture. *Proc. of the fourteenth ACM conference on Hypertext and Hypermedia*, Nottingham, UK, 81-84, 2003.
[8] R.M. Felder, B.A. Soloman. *Index of Learning Styles*, retrieved September 23, 2004 from http://www.ncsu.edu/felder-public/ILSpage.html.
[9] A. Moore, C. Stewart, D. Martin, T. Brailsford, H. Ashman, Links for Learning: Linking for an Adaptive Learning Environment. *IASTED Intl Conference on Web-Based Education*, Innsbruck, Austria, Feb 2004.
[10] A. Moore, T. J. Brailsford and C. D. Stewart, Personally tailored teaching in WHURLE using conditional transclusion. *Proceedings of the twelfth ACM conference on Hypertext and Hypermedia*, Denmark., 2001
[11] C. Stewart, A. Cristea, T. Brailsford, & H. Ashman, 'Authoring Once, Delivering Many': Creating Reusable Adaptive Courseware. *4th IASTED International Conference on Web-Based Education - WBE 2005*. Grindelwald, Switzerland, February 21-23, 2005.
[12] IMS LD, IMS Learning Design Specification, retrieved May 19th from http://www.imsglobal.org/learningdesign/index.html
[13] IMS SS, IMS Simple Sequencing Specification, retrieved May 19th from http://www.imsglobal.org/simplesequencing/index.html
[14] LOM, *Draft Standard for Learning Object Metadata*, retrieved May 19th, 2005 from http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

Authoring and delivering adaptable learning objects in SINTEC

Valentin CRISTEA, Stefan TRAUSAN-MATU
Politehnica University of Bucharest
Computer Science and Engineering Department
313, Splaiul Independentei
Bucharest, Romania

Abstract. The paper presents the solutions adopted in SINTEC, a knowledge-based e-Learning framework, to support authoring and delivering adaptable learning content. One of the main features is personalized training using enterprise / institutional knowledge repositories developed with knowledge management tools based on Web services and XML technologies. SINTEC includes tools for content development and packaging of learning objects, which can be dynamically adapted to student profiles tracked in the delivery process. The framework is compliant with different learning models, such as self-study, collaborative learning, just in time learning, and learning on demand. The environment uses e-Learning standards, such as IMS and SCORM and is based on open Web application technologies.

Introduction

The paper presents the concepts and ideas behind the authoring and delivery of adaptable e-learning content. The authors focus on content authoring and learning personalization, and on how they can be achieved by using semantic Web technologies. These concepts have been applied in the design of the SINTEC framework, which is under development at the National Center for Information Technology in the University Polithenica of Bucharest. The framework incorporates software tools for: content creation and reuse from the Internet, intelligent search of learning materials on the Web, knowledge extraction and summarization, and intelligent tutoring. They are based on open standards and technologies (including the IMS standard for e-learning). The environment may be used in various training scenarios, ranging from simple support of courses and lectures, to virtual classes and complex intelligent tutoring processes.

Developing adaptive personalized eLearning has been approached in several works. In [1] the authors describe the Adaptive Course Construction Toolkit (ACCT), which offers support based on pedagogy, instructional design principles, knowledge domain ontology descriptions and learning resource selection. In [2] the authors present the ongoing research to personalize the learning experience through adaptive educational hypermedia.

Intelligent Tutoring Systems are using students' models that typically include students' knowledge ([8]). Such models are constructed in relation with a declarative knowledge base of the considered domain, which is acquired through a conceptualization process, the result being an ontology. Such ontologies may be reused for many different applications and they are easily extensible. Several standards exist related to Web languages for ontology exchange (e.g. OWL) or languages for representing such data (DAML, DAML+OIL).

Our approach is combining the ideas of ITS, Cognitive Psychology, dynamic generation of Web pages ([7]), Emotional Intelligence (EI, Goleman) with the facilities offered by reusable ontologies on the Web and rule-based programming (e.g. CLIPS). The tools for content creation and dynamic delivery adaptation were developed as components of the SINTEC framework, but their re-design as as open Web services permits their shipping with the learning objects and easy integration with client's platform.

The structure of the paper is as follows. Section 1 presents the architecture and components of the SINTEC environment. It serves as a basis for understanding the description in Section 2 that refers to the tools for content creation and dynamic delivery adaptation. Section 3 presents conclusions and future work.

1. SINTEC architecture

Personalization is a key premise for an improved learning experience. Personalization is closely associated with e-learning and refers to the following issues:

- **Interface personalization** can range from presenting some items on user's display in accordance with user's options, up to more complex processes that include establishing the user emotional profile and adapting the interface according to the result.
- **Content personalisation** involves authoring adaptable learning materials, constant evaluation of student's knowledge level, and adaptation of learning materials accordingly. Evaluation may not always be proactive from the user's perspective (such as on-line assessments). Advanced techniques have been studied such as text mining applied to the user's public messages.

Due to the use of knowledge-based technology that includes not only domain knowledge but also psychological and pedagogical knowledge, SINTEC is able to intelligently tailor the learning process according to users' profile and progress. This makes it very useful for companies and institutions where the users profile and knowledge vary in a very large range. Moreover, it can be used around any particular domain ontology.

The framework includes a collection of tools and repositories that integrate collaborative techniques on the web with knowledge-based methods, and multiple purpose XML-based annotation (metadata, exchange of reusable components, knowledge representation) that empowers personalization. SINTEC dynamically builds and monitor ontology-based learner models, which can be further used to adapt the instruction strategies (sequences of learning objects) to learner characteristics. In addition, it tracks students' activities and interactions with the learning material, analyses their answers and texts they write, identifies needs or interest and evaluates students' psychological profile and learning style. Socio-emotional intelligence issues are also considered for tailoring the learning process [3].

Content creation and reuse is one of the main problems for e-learning systems. Professors find difficult to develop learning modules according to e-learning standards. Particular problems are the structuring and the organization of learning materials in conceptual units and the addition of metadata definitions. Other requirements for e-learning are:

- The addition of new knowledge to the knowledge pool should be easy and straightforward to persons not familiar with advanced IT technologies. Therefore, the architecture requires two different types of ontologies:
  - A *pedagogical ontology* containing concepts such as learner, learning task, activity, grade, etc. This contains knowledge about the structure and usage of the knowledge system itself.

- Several *domain ontologies*, each containing the knowledge pool for a certain domain, such as Algorithms Analysis, Compilers, Operating Systems, etc.

- The development of new domain ontologies should be straightforward even for teachers without a technical background. This is quite difficult to achieve and the viable method will be not only sets of clear instructions and samples about how to build an ontology, but also specialized editors.

The SINTEC platform is currently applying the idea of the above approaches; therefore it allows the exchange of the following types of information with other similar applications:

- Exchange of the user profile and background information, including estimated knowledge level and full training history is accomplished through the implementation of the IMS Learner Profile Information (LIP) specification
- Exchange of several types of learning content (e.g. lecture notes, practical exercises, course support materials) is accomplished through the use of IMS Metadata and Content Packaging specifications
- Exchange of test information, including questions, tests, grading and evaluation information, as well as full result history is accomplished through the use of IMS Question & Test Interoperability (QTI) specification.

The architecture of the SINTEC system is illustrated in Figure 1. From the knowledge perspective, it comprises three main groups of modules:

- Content creation and management,
- Knowledge server
- Intelligent tutoring

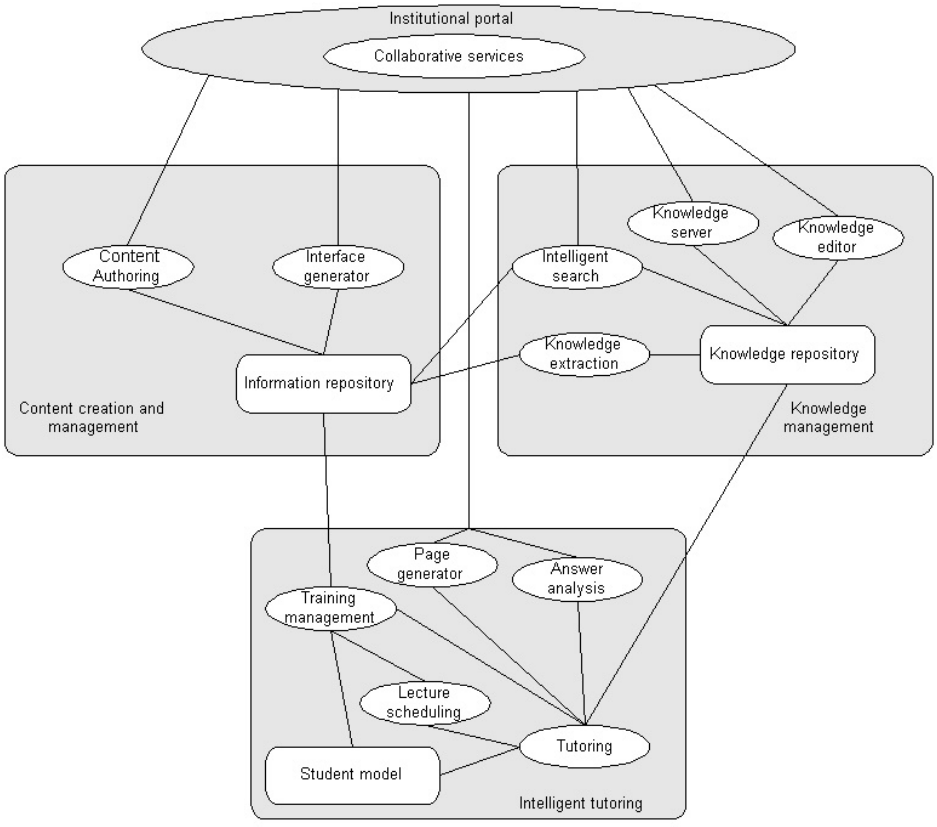


Figure 1. SINTEC architecture

For providing knowledge-based, intelligent tutoring, adaptive learning tools for the development of student models are used [3]. These tools track student's activity and her

interactions with learning materials, analyze her answers and texts she writes, identifies needs or interest and evaluate her psychological profile and learning style. One important component of the student model is the knowledge level: what knowledge she has, what knowledge she does not have and what knowledge she has wrongly. These facts are derived from answers to different questions, from the analysis of essays written by students, from students' interactions. The facts may be further used for dynamic web pages and test generation and lesson planning.

SINTEC uses several types of ontologies. The most important is the *domain* ontology that includes the basic concepts and relationships in the domain taken into account. In addition, an ontology for *pedagogy* is used for the generation of flexible, personalized learning processes. This ontology includes also the Bloom taxonomy.

A part of a *cognitive psychology ontology* that contains the concepts related to learner profiles and emotions is used for assuring a user-friendly human-computer interaction. For the selection of the relevant documents in a given learning context and of the manner of presentation, a *document ontology* is useful. Such an ontology is helpful also for the processes related to handling documents repositories and text mining.

In addition to declarative, ontological knowledge, procedural knowledge is represented in SINTEC by using production rules (in a Jess-like language offered by the Protégé environment). These rules refer to concepts, relationships and individuals in the above ontologies. They use data about the learner (e.g. his actions or results at tests) to infer and update the learner model and for planning the next learning steps.

For example, such a rule might say that:

IF the learner has not obtained a given score at a given test
AND the learning process has not been longer than the possible limit of learner's interest and attention
THEN give to the learner the task of reading the modules (or some dynamically generated web pages [6]) that covers the concepts that resulted (by inference using specific rules) to be not understood from the tests.

The training environment:

- provides a flexible and easy to use platform for both students and tutors;
- uses adaptive content based on user preferences and preparation level, both for course and test preparation and analysis;
- adapts easily to a specific domain by incorporating an adequate specific ontology;
- provides interoperability with other applications conforming to a similar set of standards contain both presentation and content authoring services Flexible, standardized, adapted to enterprise needs and to trainees profiles (including emotional intelligence).

The design aimed at obtaining platform independent components that permits a rapid deployment on different premises.

2. Tools and technologies for authoring and adaptation of learning content

This section will discuss the tools that are to be shipped with the learning objects and some of the technologies used to create learning objects and the tools themselves. We selected the virtual class scenario, in which a tutor prepares learning materials for a class and then adapts them to students according to their profiles. A tutor has the necessary tools to:

- Repackage the learning objects in the organization form best suited for the trainees.

- Define the sequencing information for the new organization.
- Alter the original metadata for revisions, authors, technical requirements, student prerequisites, etc.
- Track student progress through the activities of a course and programmatically obtaining the sequence of activities best suited for a student based on the sequencing information associated with a course package.

In order to simplify the applicability of these functions, the learning objects are shipped with two different applications:

- The *Content Authoring Center* that deals with the first three issues related to content and packaging of learning objects.
- The *Tutor Aid Center* which constitutes a tool for the instructor at delivery-time (i.e. during the course).

Figure 2 clarifies this organization. The tutor aid helps to interpret sequencing information that is not in a readable form. The IMS Simple Sequencing specification is quite complex and contains several types of rules:

- *Rollup-rules* specify how the result of a **child activity** (which is part of a larger activity) is reflected in a result for the **parent activity** (some of the simplest possible rules are: *Satisfied-If-All-Satisfied* - the parent activity is completed if all parts are completed; *Satisfied-If-Any-Satisfied* - the parent activity is completed if any part is completed).
- *Selection rules* specify how future activities are selected based on previously collected progress information. This, in turn, leads to the possibility of personalized content delivery.
- *Sequencing rules* specify how the selected activities are ordered and how the trainee may undertake them (in sequential order, by choice, etc.)

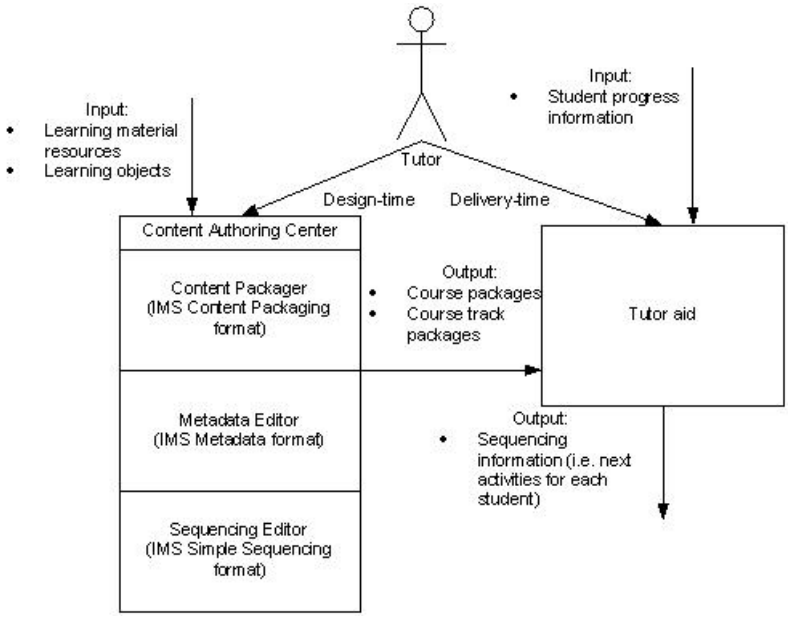


Figure 2: Instructor tools

Therefore, a tutor may find difficult applying and computing all this information manually. The *Tutor aid* helps an instructor to programmatically select the best activity track for each student, based on the information on student progress and the sequencing information found in the learning objects package. On the other hand, the *Content Authoring Center* helps defining a package structure from the learning objects provided.

This includes the basic organization, the metadata information and the sequencing information for the three standards previously discussed.

SINTEC makes use of the IMS metadata specification to represent knowledge about an item. The format chosen for the knowledge representation for a learning item (for example, a web page or a presentation document) is a string representation of a hash table, in <Ci, ri> pairs. **Ci** is the actual name of the concept addressed by this piece of learning material, while **ri** is a percentage describing relevance of that concept in the context of the current material. Since the concepts themselves are part of the domain ontology, the metadata editor is provided where instructors can simply choose the desired concepts from a knowledge base. In the simplest case, the relevance is automatically considered to be equal for all concepts involved, but the instructor may use different other policies. In Figure 3, the approach taken in SINTEC for relevance dynamic estimation is presented.

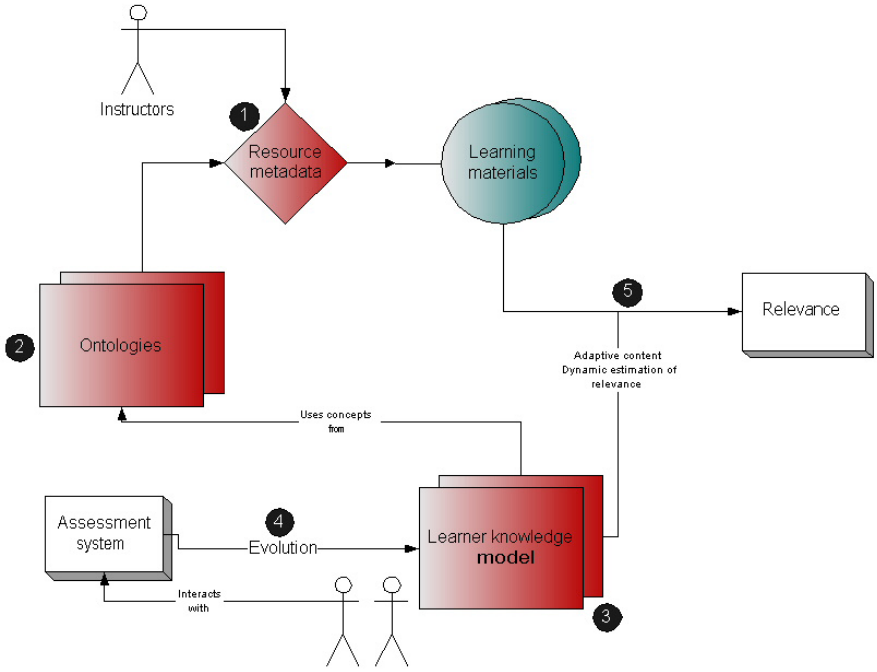


Figure 3. Dynamic estimation of relevance

The model used in the *Tutor aid* for representing user's knowledge is based on the notions of learning activity and action. A *learning activity* is an interaction between the learner and the system, which has a clearly defined goal of either transferring or assessing the transfer of a unit of knowledge to the learner. Examples are: reading learning materials, homework, group projects, taking an online assessment. An *action* is a piece of the learning activity that produces a feedback to the system that can be quantified. An example of such an action is answering a question in an online assessment.

We also define a *state* for each concept in the user knowledge data, as well as a positive and negative *score*. The states may include: unconscious-known, wrongly-known, self-learner, well-known, etc. The scores come from the quantification of learning activity actions. The number of actions providing positive or negative score is also necessary for computing averages. An example of a user concept network is depicted in Figure 4.

The network can “evolve” upon the occurrence of new positive or negative actions. When new scores arrive, they are added to the concepts addressed by the action; at least some portion of the new score must be propagated to the neighboring concepts. New concepts may be added if they are in a direct relationship to the concept whose score increases. The *propagation algorithm* we use always guarantees a limit to the number of

propagation steps. This is achieved through the gradual diminishing of the propagated value until it falls under a constant threshold.

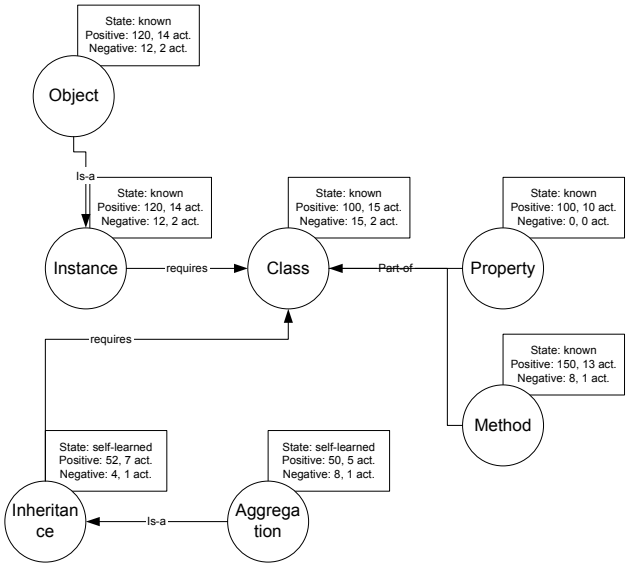


Figure 4. Part of a concept network

The querying and updating of the concept networks and ontology system is done through Algernon, a *rule-based system* developed at the University of Texas at Austin. This system has a good Java API and interface very well with the Protégé ontology engine. Since the updating of the knowledge information is done asynchronously, the production rules can be used to govern these processes.

The *Tutor aid* can use the knowledge information in several ways. Two are mentioned here: providing a personal agent for each learner that gives recommendations on certain learning materials and/or activities that a student can use to improve his/her overall knowledge level; providing a *personalized activity tree* for each learner (solution adopted in SINTEC). For example, an individual with some advance knowledge may skip some activities that he/she already covered (as indicated by his/her knowledge level).

One important issue pertaining to the proposed implementation is the means of actually developing these tools and a discussion of the technologies involved. First of all, the tools shipped together with the learning objects rely heavily on XML-based technologies (since the IMS specifications themselves are based on XML). We have chosen Java as the implementation language because of its inherent portability and its applicability to Web-based applications. Furthermore, we recognize that the intended target for this environment (that is, other e-learning research and development centers, in the academia or elsewhere) already have in place some software and hardware platforms intended for training purposes. There are two important issues here:

- Since the tools are developed in Java and use XML, the problems pertaining to compatibility are greatly reduced.
- However, with the goal of interoperability in mind, we do not impose the use of specific applications to the mentioned audience. Therefore, another version of the same tools described here *will be available as Web Services*. This greatly increases the possibility of integration with the client platforms. Besides the standalone Java versions, the URL endpoints of these Web Services that provide the same functionality, as well as the development information and Java clients for the named Web Services will be shipped as part of the same package.

3. Conclusions

As mentioned in the introduction, the experience with online course delivery of the National Center for Information Technology has led to several stages of development for an e-learning environment intended for both students and the general public. The proposal herein is the latest development from this center and focuses on advanced learning techniques and technologies. This environment attempts to achieve greater quality of learning through:

- High quality of learning materials;
- Personalized content delivery;
- Adaptation of course structure and additional information to a variety of scenarios.

Some of the most important goals in the e-learning world are reached and implemented by this environment, namely:

- Flexibility – the curriculum can adapt both to different scenarios and to a great number of student profiles within one given scenario.
- Extensibility & modularity are directly derived from the learning object structure and the ease of re-packaging these objects into flexible structures.
- Interoperability is achieved through the use of open standards such as IMS and XML, as well as the use of Java and Web Services for the implementation of tools.

References

[1] Dagger, D., Wade, V., Conlan O. (2004), *A Framework for Developing Adaptive Personalized eLearning,* retrieved May 2005 from https://www.cs.tcd.ie/Owen.Conlan/publications/elearn2004_daggerd.pdf

[2] De Bra, P., Aroyo, L., Cristea, A. (2004), *Adaptive Web-Based Educational Hypermedia*, retrieved May 2005 from http://www.win.tue.nl/~debra/webdynamicsbook/paper.pdf...

[3] Conati, C., Zhou, X., *Modeling students' emotions from cognitive appraisal in electronic games*, in Cerri, S., Gouarderes, G., Paraguacu, F. (eds.), Intelligent Tutoring Systems, Springer 2002, pp.944-954.

[4] Dimitrova, V., Self, J., Brna. 2000. *Maintaining a Joinly Constrcted Student Model.* In S.A.Cerri (ed.), *Artificial Intelligence, Methodology, Systems, Applications 2000*, Springer-Verlag, ISBN 3-540-41044-9, pp.221-231.

[5] Sowa, J. (1999), *Knowledge Representation: Logical, Philosophical and Computational Foundations*, Brooke Cole Publishing Co., Pacific Grove, CA,.

[6] Trausan-Matu, St., Maraschi, D. and Cerri, St. (2002), *Ontology-Centered Personalized Presentation of Knowledge Extracted From the Web*, in S.Cerri, G.Gouarderes (eds.), Intelligent Tutoring Systems 2002, Springer, Lecture Notes in Computer Science number 2363, pp 259-269.

[7] *Question & Test Interoperability Specification v 1.2.1*, IMS Consortium, 2003

[8] G.H. Von Wright (1971), *Explanation and Understanding*, London: Routledge & Kegan Paul

[9] *Content Packiging Specification v 1.1*, IMS Consortium 2003

[10] *Metadata Specification v 1.2*, IMS Consortium 2003

[11] *SCORM Final Specification v2.0*, Advanced Distributed Learning Consortium

[12] Cerri, S., Gouarderes, G., Paraguacu. 2002. *Intelligent Tutoring Systems.* Springer.

[13] Sleeman, D., Brown, J.S. 1982. *Intelligent Tutoring Systems*, Academic Press, 1982.

[14] P. Ciancarini et al., Coordinating Multiagent Applications on the WWW: A Reference Architecture, *IEEE Trans on Software Engineering, 24* (5), May 1998, 362-375.

[15] G.H. Von Wright, *Explanation and Understanding* (London: Routledge & Kegan Paul, 1971)

[16] Chepegin,V., Aroyo, L., De Bra, P., Heckmann, D. (2004), *User Modeling for Modular Adaptive Hypermedia*, retrieved May 2005 from http://wwwis.win.tue.nl/~vchepegi/publications/SWEL04.pdf

The Cost of Authoring with a Knowledge Layer

Lichao LI, Judy KAY
School of Information Technologies,
The University of Sydney, NSW 2006, Australia
{lli1, judy}@it.usyd.edu.au

Abstract. We have recently added a knowledge layer to a learning tool, Assess, developed to help students develop programming skills. This paper describes the authors' view of the new Assess and an evaluation of the authoring interface. We discuss the advantages of adding a knowledge layer and report the study of the effort of authoring.

1 Introduction

Many adaptive hypermedia (AH) [1] educational systems exist today, such as ELM-ART [2], WebTutor [3], INSPIRE [4] and TANGOW [5]. An advantage of an adaptive teaching system over others is that if offers a personalized learning environment and/or learning experience. Moreover, such systems are relatively simple to construct compared to traditional Intelligent Tutoring Systems. However, from an authoring perspective, an efficient AH is not at all simple to design [6]. There has been considerable work towards the support of powerful and flexible authoring for authors of AH, such as the LAOS [6] and MetaLinks [7].

In this paper, we present Assess [8], a programming education system that facilitates student self-evaluation and provide adapted learner feedback. Our research is primarily concerned with the addition of a knowledge layer to the system so that students' knowledge can be modeled and more intelligent feedback on learner's progress can be provided. In this paper, we discuss in detail the authoring of teaching material in the system. For more details of the full system and evaluations, from a student perspective, see [8].

The following section provides a brief overview of Assess from the student perspective. Section 3 describes the process of authoring of exercises in the system. Section 4 describes evaluations and Section 5 is conclusion.

2 Assess: A Self-evaluation Tool

The work reported here was conducted in the context of teaching/learning programming in C and Java in our undergraduate subject, Software Development Methods I, which is a second year programming course that aims to teach programming in C in a UNIX environment.

In Assess, exercises for self-evaluation take the form of tasks. Each task has a programming problem that students need to answer. The system allows students to provide solutions to these problems and self-evaluate their solutions against a set of marking criteria provided by authors. More importantly, it provides students with example solutions to assess.

They are normally not the 'perfect' solutions, but they do provide students some ideas to think about and evaluate. These example solutions have been pre-assessed by task authors. The student is meant to evaluate them as she did for her own solutions. Her assessment is then compared to the tutor's assessment of the same example solution. The comparison gives the student feedback on: how the teacher marked the example, the difference between the teacher's assessment and the student's assessment and why the teacher assessed it that way. This information is used to update the system's belief of students' learning progress, which can be viewed in the student's user profile. Our system's current approach to student assessment is quite unique, different from other existing systems, such as CourseMaster [9] and InSTEP [10], both of which automatically evaluate students' codes and provide instant feedback. The whole design was intended to help students reflect on code, taking the perspective defined by the criteria.

The process of student self-evaluation is shown in Figure 1. This system was used in 2004 in the Software Development Method I course. We recognised that there was a lack of knowledge representation in the system, thereby preventing intelligent and informative feedback to students. To overcome this limitation, we added a knowledge layer to Assess (See Figure 2), so that all the ad-hoc elements were replaced by a systematic knowledge layer that defines the learning objectives, user model components and domain ontology. To accommodate this new layer, the process of task authoring in Assess was changed considerably. In this paper, we present the new task creation process and its evaluation.

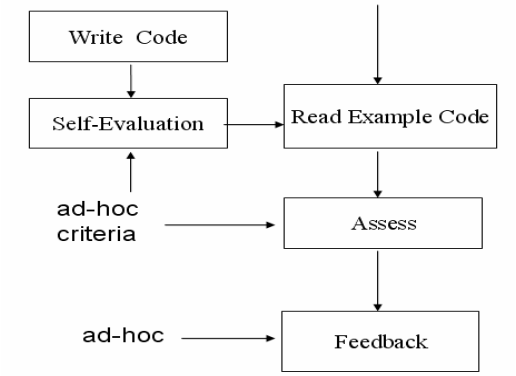


Figure 1 The Old Student Self-evaluation Process

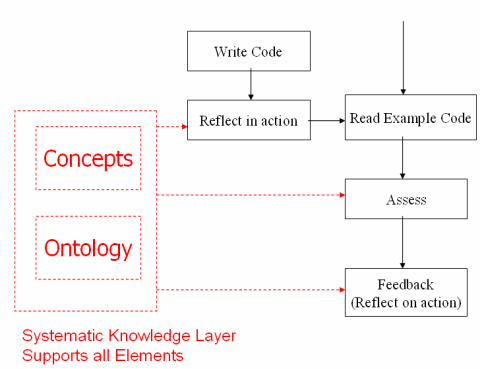


Figure 2 The New Student Self-evaluation Process

3 Task Creation with a Knowledge Layer

With the inclusion of a knowledge layer, we need to set up the learning objectives in the system. These objectives are concepts to be taught/learned, defined by teachers according to the learning outcome requirements. They are what the system attempts to teach and are used to specify teaching/learning goals of individual tasks. Moreover, they also define the knowledge that the system tries to model each student as knowing or not in the student's learner model. The addition and removal of the objectives is achieved using a web interface, as shown in Figure 3. After defining these objectives, there are three stages to follow to author a task in Assess:

Stage One. Create task statement and associate learning concepts;
Stage Two. Edit marking criteria and;
Stage Three. Create example solutions.

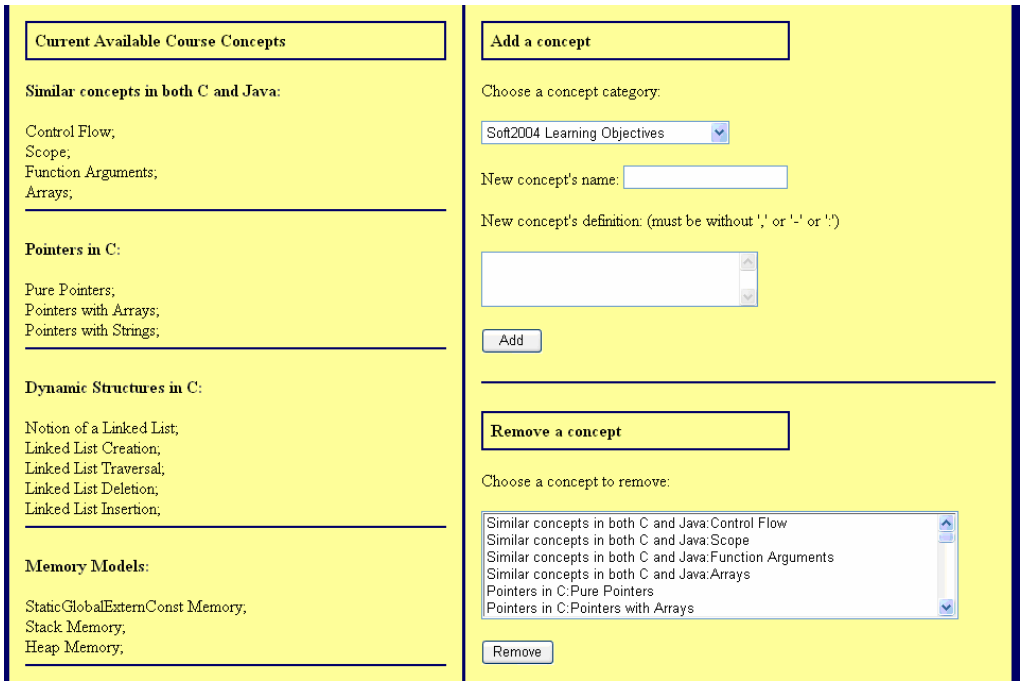


Figure 3 Concept Editing Interface (The left side of the interface shows the concepts that already exist in the system. They are organised into different predefined categories. The top right part of the screen allows new concepts to be added. Each new concept must be selected from a predefined category. The bottom right part of the interface allows concepts to be removed from the system.)

In the first stage (see Figure 4), an author provides the problem statement (i.e. the task that student needs to solve) and, optionally, a skeleton answer to help the student. They also need to indicate the difficulty of the problem and specify whether to force the student to save and assess their own answer before viewing the example solutions. More importantly, the author must relate the task to relevant learning objectives. Each chosen learning objective represents a teaching/learning goal of the task. A task must have at least one goal associated with it to denote what it aims to teach.

The second stage in task creation is the editing of marking schemes. See Figure 5. A default set of marking schemes is automatically generated based on the teaching goals for the task. As a result, when students assess their own solutions and our example solutions with them, they should concentrate on the task's learning goals. However, as the automatically generated marking schemes' criteria are not always meaningful, we allow them to be edited.

Stage Three involves creating, editing and assessing example solutions. A new example can be created at this stage. The author can edit the example solution's marking schemes. All example solutions of a task share a same set of marking schemes that were created in Stage One and Two. However, it is possible for each solution to have additional marking schemes. This means that each task has core teaching goals but each example solution can have additional elements. Figure 6 exhibits how additional marking schemes can be created and removed for example solutions. However, the task's main learning concepts cannot be edited here; as noted above, these are core to the whole task and can only be altered at the task level, i.e. at Stage One and Two.

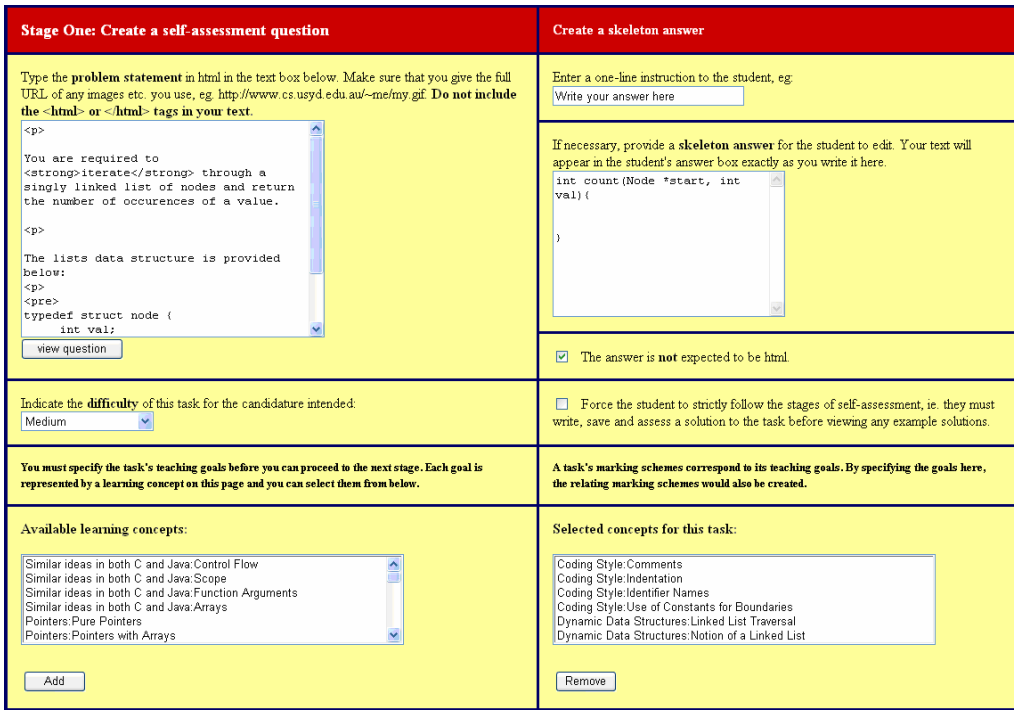


Figure 4 Stage One of the New Task Creation Process with Teaching Goals Selection (The top left part of the interface provides a text box for authors to type the problem statement. To its right, the authors can provide an optional skeleton answer. At the bottom of the interface, authors can select learning goals of the task. The bottom left box contains all the learning objectives available in the system that have not been associated with the task. The box to its right contains the learning goals of this task. A task cannot have duplicated learning goals.)

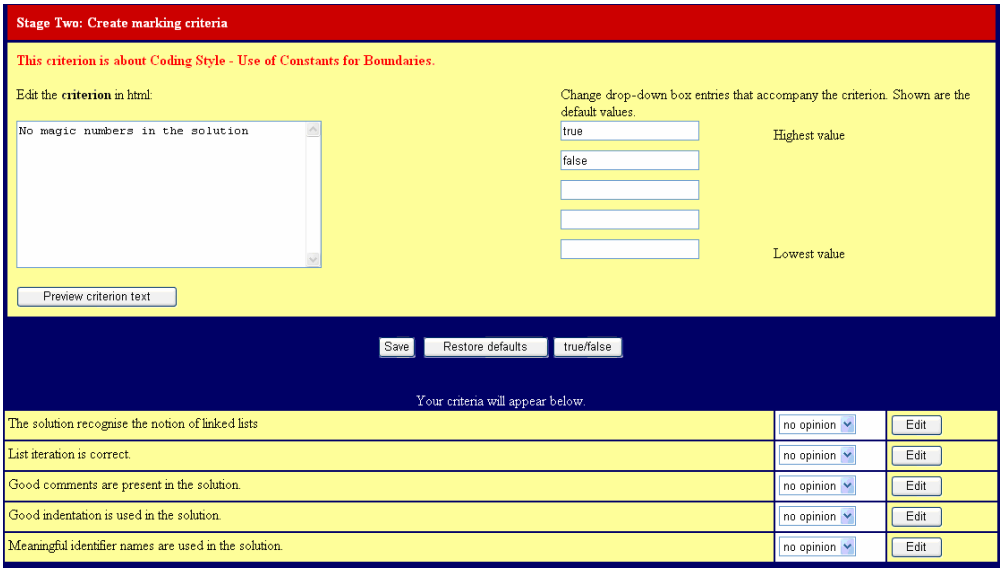


Figure 5 New Task Creation Stage Two, Editing of Marking Schemes (All marking schemes are displayed at the bottom of the page, with their criteria to the left and their marking options to the right in the pop-up menus. When an author *edits* a scheme, the scheme will be taken off from the bottom section and appear at the top section of the interface. Its criterion will be displayed in the text box at the left and its marking options are displayed in the text boxes to the right, so they can be edited. Once the author finishes authoring, she can *save* the changes.)

After the marking schemes are updated for an example solution, an author can assess the solution with the complete set of marking schemes and provide an explanation for the assessment (Figure 7). When a student assesses this example solution, her

assessment is compared with the author's assessment. The discrepancy indicates how well she understands the learning concepts associated with the marking schemes, and is recorded in her individual learner model to provide adaptive learning feedback. We can illustrate this with an example: A concept, *Flow of Control – While Loop*, is selected to be a learning objective of a task and its marking scheme is created automatically. The marking scheme's criterion is "The while loop used in the solution code is correct" and its marking options are *true* and *false*. When a student assesses an example solution with this scheme, she thinks the loop used in the code is correct but the author of the task thinks otherwise, this shows the student cannot recognise the elements that make up a correct while loop and so does not understand this learning objective yet. This information is recorded in her learner model.

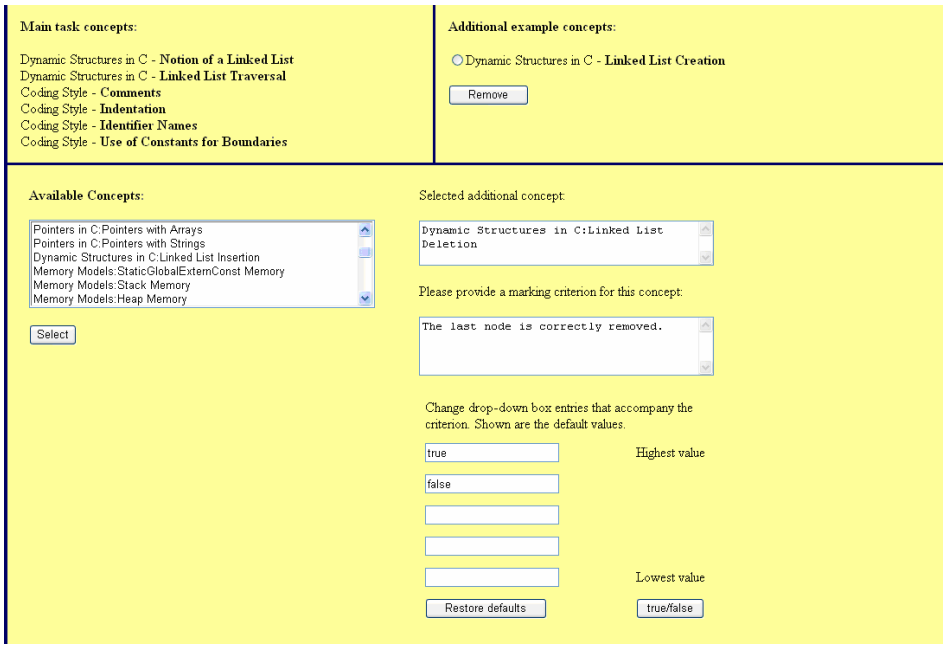


Figure 6 Editing Example Learning Concepts for One Example Solution for One Task (The top left section shows the main learning/teaching objectives of the task. The top right section displays the learning goals of this particular example solution, in this case, *Dynamic Structures in C – Linked List Creation*. They can also be removed from here. The bottom part of the interface allows additional learning goals be added and corresponding marking schemes to be created.)

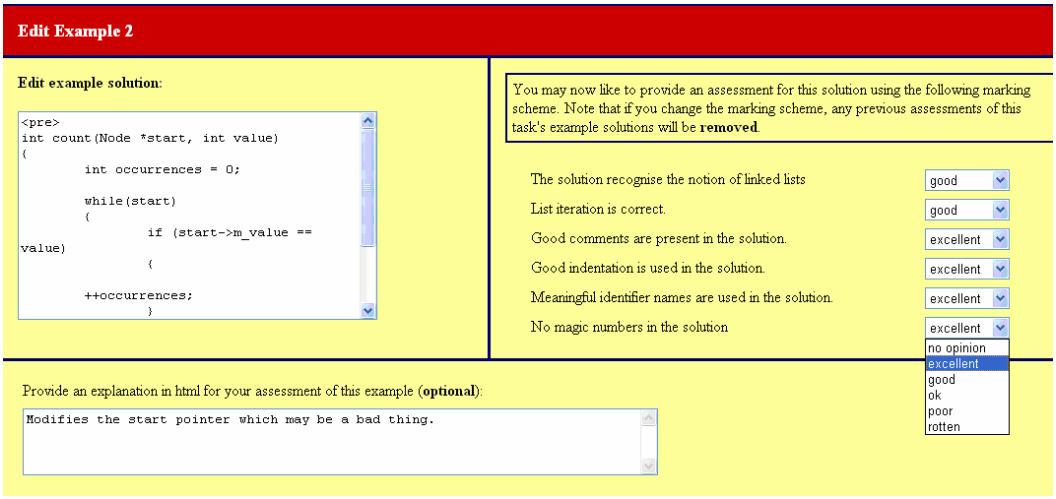


Figure 7 Assessing Example Solutions (The interface allows the example solution in the top left text box to be edited. In the top right section, the author can rate the code against the marking schemes. She can also provide an optional explanation in the text box at the bottom.)

When the author finishes creating and assessing example solutions, and is content with the entire task, she can publish it for students to view.

4 Evaluation

We have conducted a preliminary experiment on the author's perspective of Assess. It was designed to assess the intellectual effort and time involved in the creation of tasks in Assess as well as the usability of interfaces. In particular, we want to ask "What effort and time is involved in entering a new task into Assess?" To answer this question, we evaluated:

- How effective and usable are the interfaces?
- How quickly and accurately first-time authors can create a task?

We selected five[1] participants from our computer science honours and fourth year students to take part in this user trial. They were from different backgrounds. Participant 1 was an experienced tutor, but had not tutored Software Development Methods I before. Participants 2 and 3 were tutors of the subject in 2004. Participant 4 has never tutored and Participant 5 was a tutor for only a brief period in 2003. Participant 3 was the teaching assistant for SDM in 2004. All participants were in the top 15% of the class when they completed the course, so they were all familiar with it. Though this is clearly a highly qualified, technically elite group, it represents the class of users qualified to define and enter tasks for the subject.

At the start of a session, we demonstrated Assess from a student's perspective to show how the system works. This allowed participants to get some insight into Assess, but did not bias the experiment by letting them see what they would need to do. We also supplied the participants with materials they need to put into the system. This included a problem statement and two example solutions that were taken from the original Assess system. They were asked to re-create this task in Assess. They were also told the task's and example solutions' teaching goals. We did not ask the participants to create new tasks because creation of teaching materials is always a time consuming process and requires deep understanding of the big picture course goals. We were not trying to determine how people tackle the more intellectually demanding task of choosing a task and providing solutions. We had also pre-typed the teaching materials on a text editor, so participants could simply cut and paste them into the interface. This reflects a typical scenario of Assess task creation where the lecturer has set an exam question, graded student answers and then developed and tested example solutions. In such a case, all the materials made available to participants would have been complete at the time the lecturer added the task to Assess.

For the experiment, participants were asked to create a task in the system with the supplied material and grade the teaching material creation process in terms of its usability separating the intellectual effort of it from the ease of use of the interfaces. Participants were asked to think aloud so we could take note of any difficulty they encountered.

All participants completed the user trial successfully. The intellectual effort required for each stage of the task creation was rated from 1 (minimal effort) to 6 (a lot of effort). Participants' ratings are illustrated in Figure 8, which shows only modest intellectual effort is required. Participant 4 considered Stage One and Two required a lot of effort. Participants tended to consider Stage Two required the highest (or equal highest) level of intellectual effort as creating questions that can properly evaluate students' understanding of a marking criterion's corresponding learning concept is not an easy process. In terms of

[1] As Recommended for think-aloud usability evaluations: How many users to test (Jakob Nielson's Alterbox), http://www.useit.com/alertbox/20040719.html, 22 Oct, 2004

interface usability, most participants have positive opinions. Participant 3 rated the interface in Stage Two not easy to use because the participant thought there was not enough guidance for authors (See Figure 5). Participant 5 rated the interface of Stage One not very usable and suggested that there was too much information presented (See Figure 4). Overall, the new functionalities required by the knowledge layer did not increase the intellectual effort required to create a task greatly or introduce too much complexity to the interfaces.

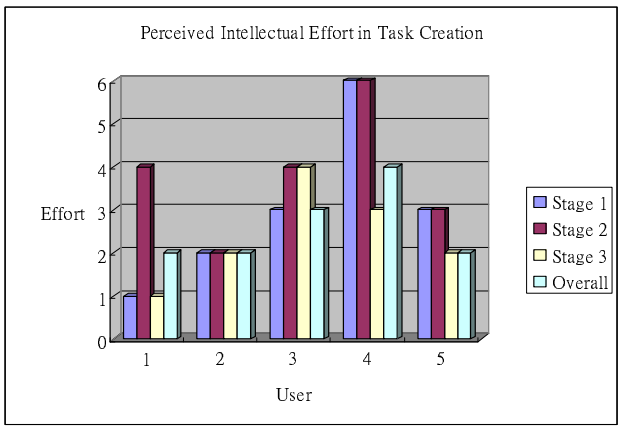


Figure 8 Illustration of Participants' Rating of Intellectual Effort Required for Each Stage of the Task Creation Process (1 = minimal, 6 = a lot of effort)

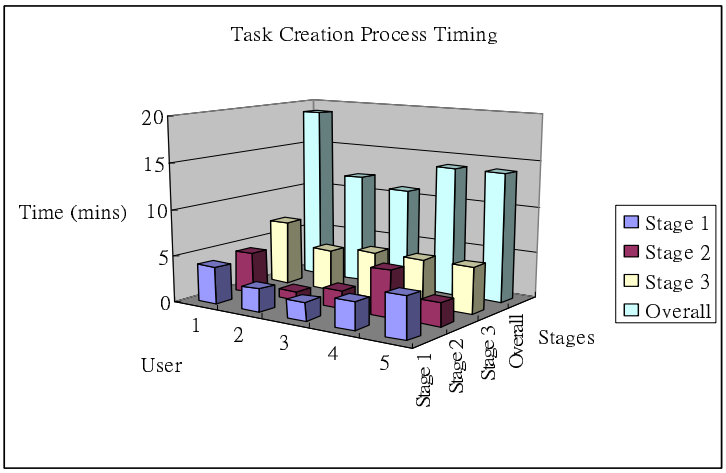


Figure 9 Illustration of Different Participants' Timing of Each Stage of the Task Creation Process

Since this was a think-aloud evaluation, timing data must be interpreted cautiously. However, it gives an indication of the effort and time involved and, taken with the observations of participants, is valuable for establishing indicative times for task creation. The time each participant spent in different stages of the task creation process is listed in Figure 9. All but one participant finished creating the task within 15 minutes. It took the first participant 19 minutes to finish as it was the first user trial run and the participant gave considerable feedback on the experimental procedure during the user trial. Participants 2 and 3 took a shorter time than the others because they were familiar with the course concepts as they had been tutors of the subject as would be the norm for a typical task author. Stage Three slowed for all participants because there were more functionalities involved and more interfaces to view. It is to be noted here that the overall time is not only the sum of time the participants spent on the four stages. It includes also time that

participants spent reading instructions on the tutor home page and task home, which are not part of any specific stage. Of course, it also included any time talking to the observer.

5 Conclusion

We have presented Assess, a student self-evaluation tool, and explained the exercise authoring process of the system with a consistent knowledge layer. We also described a user trial, with results showing that:

1. The intellectual effort required to enter a new task to Assess is modest and the interfaces that allow the creation of a new task are relatively usable;
2. The entire authoring process in Assess takes about 15 minutes, which again shows the effort and time involved, introduced by the knowledge layer, are minimum.

They indicate that the new knowledge layer adds only modest complexity to the task of authoring teaching materials in Assess. At first, one might imagine that the addition of a knowledge layer to a conventional learning tool might require more effort from task authors. From our experience, it seems that the knowledge layer may actually reduce the work of defining a new task and improve the quality of the task as the learning objectives are explicit so helping authors concentrate on what they want students to learn. Moreover, the marking criteria in the new Assess system are automatically generated and they correspond to the teaching/learning objectives of the task. This means that authors do not need to create the questions that test the learning objective for the task and avoids the risk of forgetting to include them. Furthermore, the knowledge layer makes it feasible to provide learners and authors with learner models that capture learning progress, supporting reflection.

References

[1] Brusilovsky, P. (2002) *Adaptive hypermedia*, User Modeling and User Adapted Interaction, Ten Year Anniversary Issue 11 (1/2), 2002. 87-110.

[2] Brusilovsky P., Schwarz E., and Weber G. (1996) *ELM-ART: An intelligent tutoring system on World Wide Web.* In Frasson, C., Gauthier, G., and Lesgold, A., eds., Proceedings of the Third International Conference on Intelligent Tutoring Systems, ITS-96. Berlin: Springer. 261-269.

[3] Pérez T.A., Gutiérrez J. (1996) *WebTutor*, Un sistema Hipermedia Adaptativo para la educación en WWW, Actas del V Congreso Iberoamericano de Inteligencia Artificial, IBERAMIA'96. Cholula, Puebla, MÉXICO, 1996.

[4] Grigoriadou, M., Papanikolaou, K., Kornilakis, H., & Magoulas, G. (2001) *INSPIRE: An INtelligent System for Personalized Instruction in a Remote Environment.* In P. D. Bra, P. Brusilovsky, & A. Kobsa (Eds.), Proceedings of Third workshop on Adaptive Hypertext and Hypermedia, July 14, 2001. Sonthofen, Germany, Technical University Eindhoven. 13-24.

[5] Carro R., Pulido E., Rodríguez P. (1999) *Task-based Adaptive learNer Guidance On the WWW: the TANGOW System*, Second Workshop on Adaptive Systems and User Modeling on the Web, en la Eighth International World Wide Web Conference. Toronto, Canadá. Mayo 1999.

[6] Cristea A. (2004) *Authoring of Adaptive Hypermedia; Adaptive Hypermedia and Learning Environments*. Book chapter to appear in "Advances in Web-based Education: Personalized Learning Environments", Sherry Y. Chen and Dr. George D. Magoulas (eds.). IDEA Publishing group.

[7] Murray T., (2002) *MetaLinks: Authoring and affordances for conceptual and narrative flow in adaptive hyperbooks.* International Journal of Artificial Intelligence in Education, 13 (1).

[8] Li L., Kay J., (2005) *Learner Reflection in Student Self-Assessment,* TR 568, School of Information Technologies, The University of Sydney, ISBN 1864877170.

[9] CourseMaster. (17 May, 2005). CourseMaster [Online]. School of Computer Science & IT, The University of Nottingham, UK., Available: http://www.cs.nott.ac.uk/CourseMaster/cm_com/index.html

[10] Odekirk-Hash, E. (2001). *Providing Automatic Feedback To Novice Programmers.* Unpublished MA, The University of Utah, Utah.

Telemedicine ontology for AIED

O. Ferrer-Roca, A. Figueredo, K. Franco, A. Diaz-Cardama.
CATAI. *UNESCO Chair of Telemedicine. University of La Laguna. Faculty of Medicine. 38071. Tenerife Canary Islands. Spain (phone: +34-922-642015; fax: +34-922-641855; e-mail: catai@ teide.net; http://www.teide.net/catai)*

Abstract.Telemedicine (TM) challenge is to create an intelligent tool that delivers personalized training to professionals with different backgrounds. We present an adaptive retrieval system that used vocabulary and ontologies founded on the telemedicine body of knowledge (TM-BoK) hierarchy and Medical Sub-headings (MeSH). The XML-manifest that contains a Navigable Knowledge Map and a separate Rule-extension executed by Agents during the process of navigation. The result is an adaptive and adaptable TM knowledge delivery tool.

1. Introduction

Professionals with very different educational backgrounds use Telemedicine (TM) (e.g. doctors, engineers, computer scientists, etc.). It is difficult to find experts in every key subject being a challenge to build an intelligent tool that provides personalized distance training with up-to-date information from any source, including the Internet.

Medical information retrieval has been based on keyword matches of resource descriptions or metadata. Syntax and semantics used to tag contents, have been progressively incorporated into professional indexes. Those specific taxonomies or vocabularies, such as Medical Sub Headings (MeSH) [1], provide a certain level of standardization.

For teaching purposes educational content and user profiles (e.g. IEEE-Learning Object Metadata (LOM) paradigm [2], and IMS-Learning Information Profile (LIP)[3]) are required. Included in SCORM (Sharable content object reference model)[1], it allows re-usability, interoperativity and extensibility. Learning Objects using XML, are capable of being understood by most e-learning tools. And its Content aggregation model and Run-time environment specifications, aggregate and display the same pool of learning objects in different orders or with different views with an intelligent Learning Management System (LMS) able to deliver *adaptive* and *adaptable* data.

Interactive-adaptabliliy such as to DILE (Distributed Intelligent Learning environment) based on a Multiagent systems (MAS)[4]. in order to set rule-building strategies for learning delivery actions; it takes into consideration Student Cognitive State, Teaching strategies and Knowledge acquisition assessment.

Interactive-adaptive a step further, represents a run-time learning delivery strategy based on detected skill management during the e-learning time [5]. In this case the platform dynamically re-adapts, exchanges, re-uses and shares learning objects (assets) according to user feedback, thus optimising skill acquisition.

The present paper presents a tool capable of managing students' interests and skills applied to TM e-learning. The objective was to build and test an intelligent tool capable of handling specific TM ontologies and at the same time electronically deliver personalized TM content depending on user knowledge and learning process.

2. Design

[1] www.adlnet.org

Our starting point was the IST-1999-12503- Knowledge on Demand (KOD) project[2]. Based on Agent technology, it builds an e-learning tool with the following properties: automation, adaptability, intelligent management and re-usable learning objects.

I.A System Description

We specify the TM modifications introduced in the above system. Adjusted to standards [2-5] the final system is integrated by *Authoring Components* ready to interact with a *MAS* FIPA compliant [6]. See **Figure 1**.

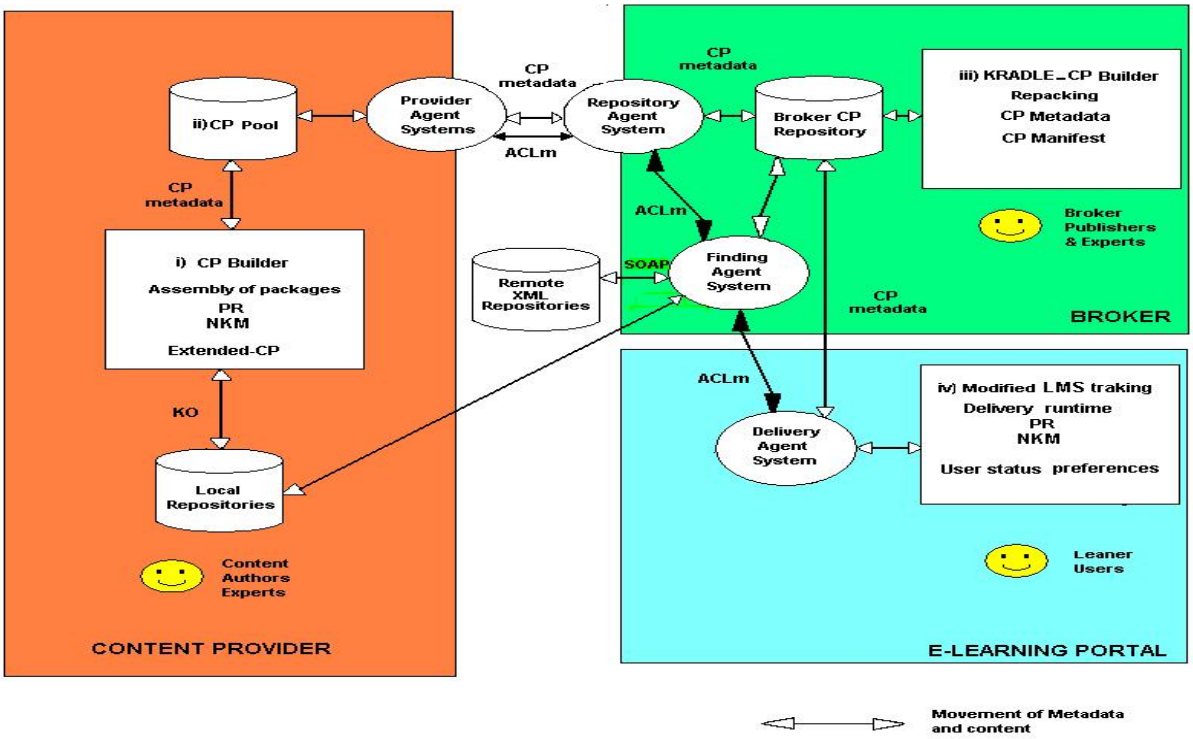


Figure 1. KOD system Description. Interaction between *Authoring components* and *Multi-Agent system*.

ACL= Agent Communication Language; AS= Agent System; CP= Content Package; KO= Knowledge Object; KRADLE= KOD Reusable Adaptive Learning Content Exchange; LMS= Learning management system; NKM= Navigable Knowledge Map; PR= Prescriptive rules; SOAP= Simple Object Access Protocol.

The *Authoring Components* are: i- a Content Package builder, ii-a Package Pool, iii-a KOD Reusable ADaptive LEarning Content Exchange Broker (KRADLE), iv- a Learning Environment, and v- an Educational Metadata Editor Manager.

i- The Content Package builder packs the information into an "extended" IMS-CP[3] standard. **Figure 2** shows the *knowledge rules* and *navigable knowledge map* extensions. Rules are packaged separately in order to be re-used. The knowledge map is a domain map representation (see below II.B.3)) built with items connected by attributes, concepts and available resources .

ii- The Package Pool collects and publishes packages and is able to detect new packages using Agents.

iii- The KRADLE is the broker of the remote repository of packaged metadata, whose Content Package Manifest is available for Agent interaction.

iv- The Learning Environment is the vertical learning portal for publishing, accessed via WWW. It includes a "modified" Learning Management System (LMS) that keeps

[2] http://sharon.cselt.it/projects/jade/whoIsUsing/KODAgentsandeLearning.doc

[3] According to the IMS-CP (Instructional Management System-Content Packager) specifications of 2001, the learning packages are collections of different "organizations", each one including a number of "items" (learning paths); every item refers to one resource, which can include a number of learning objects.

learner performance and profile updated for Agent handling together with the usual LMS activities. These are student registration, sequencing instructions, content administration, assignment and recording performance, collection and data management.

v- The Educational Metadata Editor Manager is a tool to define, generate, export, and validate extra metadata. This is a Java applet editor suitable for modifying the document where the XML tag definitions are stored (Document Type Definition -DTD).

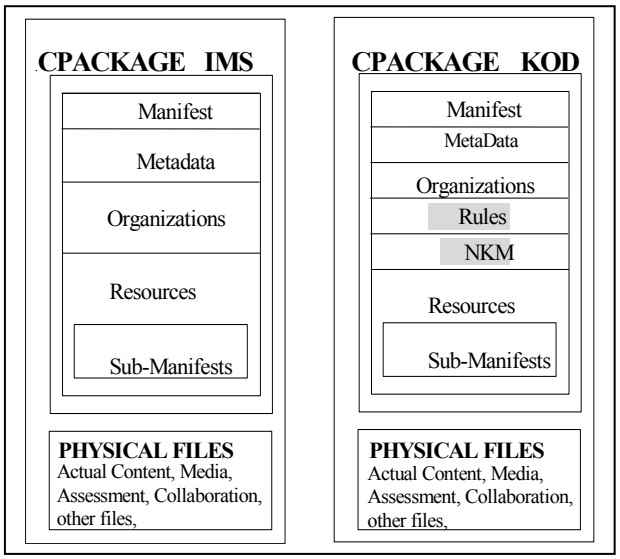


Figure. 2.- XML Content Package structure. The standard IMS CP on the left, versus IMS KOD CP on the right. NKM = Navigable Knowledge Map

With respect to the *Multi-agent System* (**Figure 1**), messages between Agents are passed via Agent Communication Language (ACL) in *custom ontologies* (see below II.B.3. Managing attributes and enhanced data) based on adaptation rules, ontology, language and content. The agent architecture contains three functional layers: publication, brokerage and delivery. Each layer has agent systems as listed below; agents are named in brackets.

1. Knowledge Package publication layer. This has a Provider Agent System (Knowledge Package Publication monitor; Publisher Contact agent)
2. Knowledge package broker: 2.1. Repository Agent System, responsible for receiving and managing incoming packages. (Knowledge Package receiver, Upstream Informer, ACT-ACL translator); 2.2 Finding agent systems. (Broker Package Finding, SOAP[4] Client Agent),
3. Knowledge package delivery layer, which is the e-learning service provider containing a Delivery agent system (e-Learning Service Provider Package Finder agent)

Except for the SOAP Client agent, none of the above can be duplicated in a platform.

I.B TM System Design

It consists of:

1) An *Extended-Content Package:* Composed of TM knowledge learning objects packed in XML and tagged with metadata (e.g. IMS-LOM metadata). In addition to this, several navigation descriptions are included such as the *Table of Contents*, *Prescriptive Rules or Knowledge Maps*.

The default Table of Contents (TOC) hierarchy is extended into three levels allowing user adaptive navigation through telemedicine items. These levels are:

1. Conditional branching (ADL-SCORM[1]) with Boolean conditions on lesson status.

[4] SOAP= Simple Object Access Protocol

2. Prescriptive sequencing Rules that control content activation based on the Learning Management System (LMS) tracking, taking into account user status and/or preferences. (See **Table III**).

3. Knowledge maps capable of changing the domain organizational views, depending on the Prescriptive Rules.

2) An ontology adapted to Telemedicine: For this learning environment we established specific vocabularies and domain ontologies capable of being used by metadata handling Agents.

a) Telemedicine classification: This contains categories whose entities are assigned according to one or more established criteria. There are twelve main categories in the Telemedicine Body of Knowledge (TM-BoK) [7]: *[1]* History of Telemedicine, *[2]* Minimal Technical Requirements, *[3]* Main Telemedicine Applications, *[4]* Basic Knowledge of Multimedia Communications, *[5]* Quality Control and Quality Assurance, *[6]* Internet in Telemedicine, *[7]* Distant Training Tele-Working and Tele-Teaching, *[8]* Data Security and Privacy, *[9]* Liability and Legal Aspects, *[10]* Economics and Management in Telemedicine, *[11]* Social Aspects and Technology Transfer, and *[12]* Emerging Issues.

The less common/significant entities are included in "other categories" and cover: *[i]* Standardization Bodies, *[ii]* Statistics, *[iii]* Colour Theory, *[iv]* Networking & TCP/IP[5], and *[v]* Informed Consent.

b) Coding schemes: The code-dependent hierarchy structures the major categories content into subheadings. For example, the Major Category-*[3]* entitled "Main Telemedicine Applications", has the following subheadings: *[3.1]* Tele-radiology, *[3.2]* Tele-pathology, *[3.3]* Tele-cardiology, *[3.4]* Tele-home Care, *[3.5]* Tele-oncology, *[3.6]* Tele-surgery, *[3.7]* Tele-psychiatry, *[3.8]* Tele-dermatology, *[3.9]* Primary Care, and *[3.10]* Phone medicine.

c) Medical sub-heading for indexing medical procedures: Considering that TM is a medical subject, Medical SubHeading (MeSH) qualifiers can be used to refer to headings when applied to specific medical delivery procedures.

3) Managing Attributes and Enhanced Data: The extended content package (**Fig 2**) selects vocabularies twice; once for the metadata fields and once again for the Data Model and Navigation Knowledge Map.

Metadata fields are associated with the provided vocabulary including TM (**Table 1**). In the case of the Data Model (**Table II**) and Navigation Knowledge Map, the *domain* is selected first, because it determines the specific vocabulary. In our case MeSH and TM domains were chosen, meaning: main categories II.B.2.a) supplemented with II.B.2.b.) and II.B.2.c.). Nevertheless, in some specific main categories different vocabularies are required (i.e. *[9]* Liability and legal aspects require a Legal domain vocabulary)

Table I- Metadata vocabularies Association

Language	English / Spanish
Key-words	TM-BoK ; MeSH
Version	No vocabulary
Status	Lifecycle.status
Format	Mpeg/doc/html/ppt/pdf/xls
Learning resource type	Exercise/figure/table/problem/questionnaire/index/exam/test/simulation/graph/narrativetect/selfassess/diagram/slide/experiment/URL

[5] TCP/IP= Transmission control protocol/ Internet Protocol

Interactivity level	
Semantic density	
IntendedEndUserRole	Doctor/nurse/managerial/technical
Difficulty	Veryeasy/easy/medium/difficult/verydifficult
Context	Univ1cycle/Univ2cycle/Univ3cycle/ContEd/CovT
Relation.kind	Ispartof/isversionof/isformat/isreferencedby/haspart/hasformat/isbasedon
CopyrightAndOtherRestrictions	Yes/no
Subject	MeSH; Telemedicine; Legal; etc…

In the TM-BoK ontology, the attribute values or qualifiers of the main categories (i.e. nodes) are capable of building the dependency maps specific to the TM learning system. This would not be possible if MeSH qualifiers were chosen since they are not adapted to telemedicine categories and subcategories. This is regardless of the fact that both TM-BoK and MeSH hierarchies allow broader (parents or ancestors and siblings) and narrower (children or successors) concept relationships; and, that within a given hierarchy, a single concept may appear either as a narrower one or as more-than-one broader concept, thus being capable of creating dependencies and Knowledge Maps.

Table II- KOD Data Model for learner profile characteristics according to LIP model

KOD Data Model	**Learners Characteristics**
Kod.learner.demopersonal.language	Language
Kod.learner.ld.learnstyle	Learning Style (ILS-Index learning stile Felder & Silberman)[6]
Kod.learner.objective.[]	Goal (MeSH vocabulary)
Kod.learner.objective.[goal].interest_level	Competency
Kod.learner.objective.[goal].classification	Interest

In **Fig. 3** the Tele-radiology knowledge map *[3.1]* is shown to contain: Basic parts *[3.1.1]*[7]as well as Fundamental nodes (The term fundamental refers to main categories in the TM-BoK).

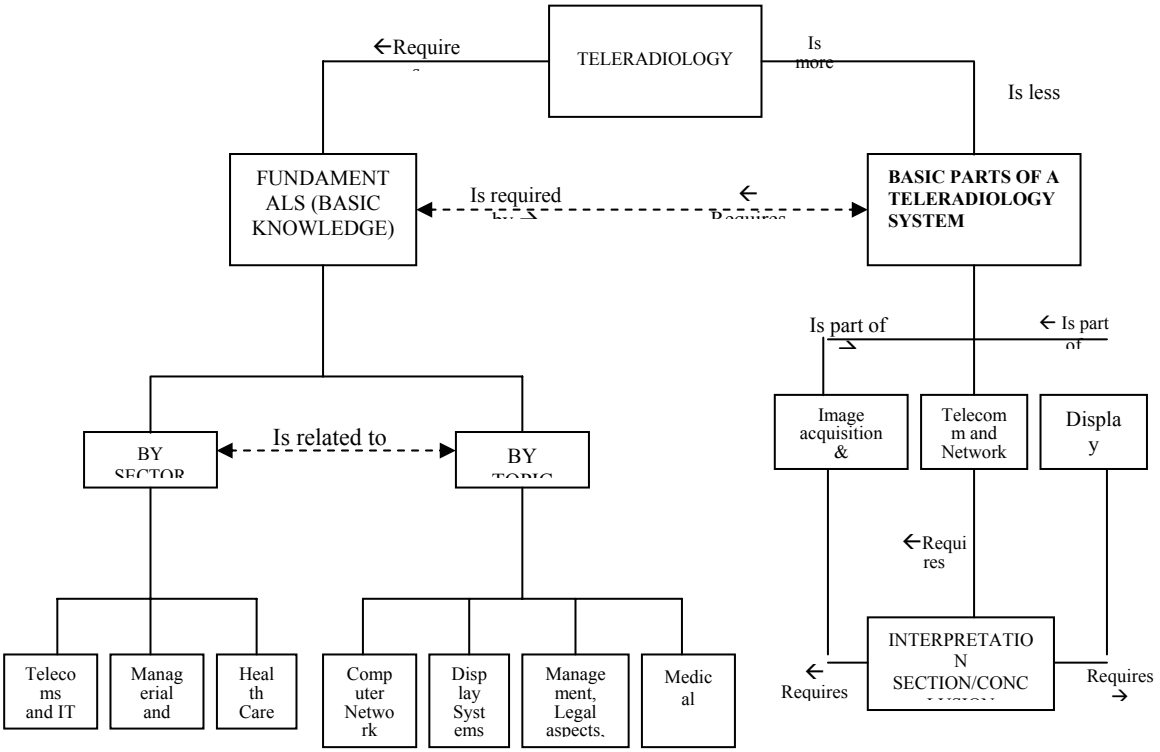


Figure 3. Knowledge map nodes of Tele-radiology

[6] Learning and Teaching Styles in College Science Education (http://www2.ncsu.edu/unity/lockers/users/f/felder/public/Papers/Secondtier.html)

[7] *[3.1.1.1.]* Communications, *[3.1.1.2.]* Display systems, *[3.1.1.3.]* Image acquisition & management, and *[3.1.1.4.]* Interpretation

Defined *custom ontologies* allow complex data structures to pass among agents within Agent Communication Language messages. Our ontology implementation for communication purposes was deliberately simplistic. It was basically a rule container since attributes were considered beyond its scope particularly because actions are not read in the ontology but implemented through the behaviour of agents.

Adaptive package delivery was under the control of the *modified*-LMS, which was capable of interacting with agents (delivery agents) and of executing the run-time rules encoded in the Content Packages. The modified-LMS was also responsible for storing and checking user profiles and complementary information such as: elements already visited, performed rules and updated user knowledge. All the above is essential for personalized adaptive delivery.

3. Application Deployment-TM demonstrator

Two innovations have been implemented: A) Personalization and B) Re-using.

I.C Personalization

This starts with the system *dimension* followed by prescriptive or adaptation rules to deliver customized contents.

The dimension definition enables the system to select *determinants* or parameters that help to decide whether the content (constituent) must be presented to a particular user. The *constituents* are divided into learning paths (collections of learning assets) and learning assets.

The dimension of the TM demonstrator takes into account user backgrounds, learning styles and goals:

- **Individuals**: The TM introductory courses reach a wide range of professionals, who were classified into three main groups: Medical Informatics Experts, Health Care Personnel, and Managerial people. The main determinant for the first group was technical issues, for the second medical items, and for managers economic and legal aspects.
- **Learning styles**: Regarding the format of available documents, half of the material had optional (textual or visual) presentation formats.
- **Topics**: Telemedicine being a new discipline, most material is *introductory* with a reduced number of *advanced* items. For that reason, we rejected so-called "dimension-levels", because no critical mass of contents is to be located in the advanced content set, in this initial demonstrator.

Table III- Pseudo-code of a prescriptive/ adaptation rule that controls content activation to deliver customized content.

Initialise an element as “disable”. IF determinant=true THEN constituent=enable, AND IF (kod.user.occupation=medical AND kod.user.learningstyle=visual) THEN (kod.behavior.seqnav(man1_ToC_234_(medicalvisualpresentation.ppt)=enable)”

Table III shows a pseudo-code example of an “*adaptation rule*”. When a particular user meets both conditions (being a doctor and interested in visual material), then the element addressed in the “table of contents” with the number 234 -corresponding to medicalvisualpresentation.ppt item- will be activated. In **Figure 4b** the final result is displayed.

I.D Re-Using

The authoring tool imports raw learning assets located anywhere (e.g. Internet) into the Resource-window (**Fig 4)**. Once in the Knowledge-Object-window of the application, they are re-packed, re-used or modified according to the adaptive Knowledge Rules. As a result of the number of permutations (n-objects per j-dimensions), learning data delivery become individually adapted and highly personalized.

The personalized packaged course (re-packable in each session or by the author) is just an aggregation, in the Knowledge Object window, of raw assets.

The system is permanently updated. For that purpose the "user-Agent" (who identifies each learner) of the Finding agent system (see **Fig 1**), looks into the available repositories for any material or complete package suiting user demands. Besides, it communicates with other agents in order to search in different repositories, including the Internet. The returned packages can be incorporated interactively into the course.

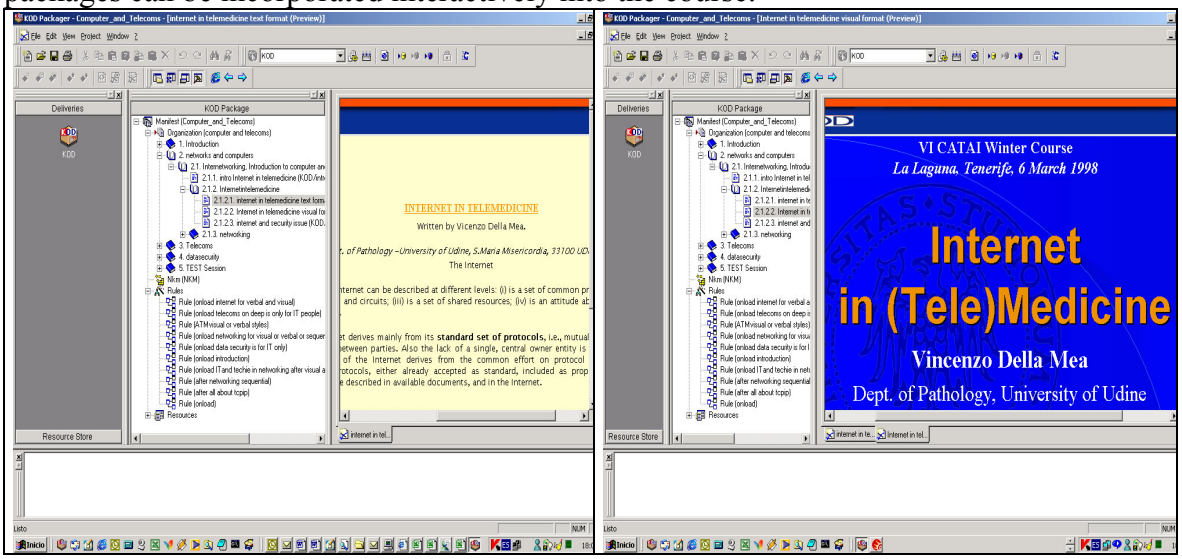


Figure. 4. – Knowledge personalization according to learning style. A. Textual B. Visual

4. System Validation

For system validation a TM course integrated by a number of packages (cluster of contents organized by subject, topic, taxonomy…) to teach tele-radiology were build

After checking student profile, the system decides which of those contents are suitable. The purpose is to deliver to each learner category exclusively the required information that fits his interest (doctors/ engineers/ managerial), learning style (visual or verbal) and goal (teleradiology/ quality control).

The adaptive course and package builder was provided to 96 students and 33 teachers. Their opinions were evaluated using questionnaires (**Table IV**).

Table IV. A.User interest-difficulties to Telemedicine students.	B. Teachers to analyse the authoring tool
1. How did you find the Tele-radiology learning usability? 2. In your opinion the Telemedicine application and learning assets are… 3. Has learning and time searching for contents improved?. 4. Was tool familiarization time short and adequate? 5. Were you able to track and bookmark your progress?. 6. How did you find the telemedicine demonstrator? 7. What is your opinion of the course building packager? 8. Was the package builder easy to use: 9. Is it easy to built a Telemedicine course? 10. Is the TM demonstrator effective for learning:	1. Opinion on Authoring usability? 2. Opinion on Functionality for learners 3. Opinion on the capability to create adaptive learning material. 4. Opinion of the terminology used in the author interface 5. Opinion on completeness of questions and test capabilities of the system? 6. Opinion on the use of resources. 7. Is the tool easier or as good as other e-learning tool? 8. Is the Telemedicine demonstrator an effective learning tool? 9. Opinion of the author user interface. 10. Are steps to build a Telemedicine course simple enough? 11. Is the terminology of the interface adequate? 12. Does it support all expected functionalities for authors, publishers and brokers? 13. Is the time spending to get acquainted with the use of the software reasonable? 14. Do you consider that the test capabilities in the current version are sufficient? 15. Opinion on the Rules interface?

Answers were weighted from 0 to 3, with 3 being the maximum positive evaluation and 0 a negative or "do not know" answer . The global score was obtained summing the weight of each

answer. The result was normalized dividing it by the maximum weighted score per answer (3x96=288 for students and 3x33=99 for the teachers).

Students of Telemedicine (University optional subject), evaluating the course, gave the results seen in **Figure 5a**. The average normalized weighted value was 57.65 per 100 with a standard deviation of 3.23. The best score was for time reduction in learning TM or searching for updated information. The lowest score (53) was the time spent in becoming familiar with the tool.

Teachers were professionals familiar or not with e-learning tools. Results (**Figure 5b**) showed a 53.3% average weighted score with 10.03 standard deviation. The best score was for the capability to build adaptive courses (score 68) followed by user interface, learner functionality (61) or rule-interface (60). The lower scores were for test-building facilities incorporated in the tool (33) and the time required to learn its use (38). A medium score (51) was given to the simplicity of building Telemedicine courses.

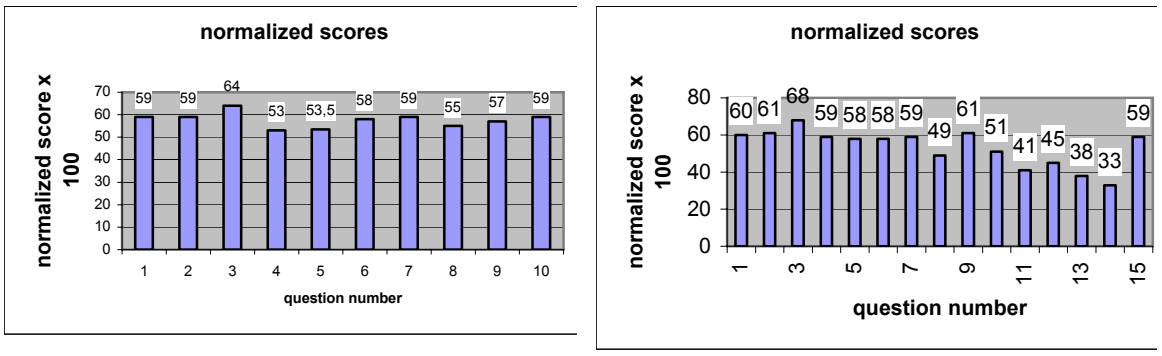


Figure 5. 5A-Students´ normalized score evaluation of the Tele-radiology course. 5B- Teachers´ normalized score evaluation of the Telemedicine authoring system.

5. Discussion

The present TM Open and Distance Learning structure based on agent technology proved capable of personalized data retrieval according to user profile and goals. For that purpose we developed local and remote XML repositories, tagged with TM metadata vocabulary following TM ontology capable of being used by metadata handling Agents.

As shown in the design and deployment section this results in a multi-role personalised learning platform with a modular architecture that uses collaborative software Agents capable of reading the information located in the *modified*-XML-Manifest. Other Agents are devoted to representing the various user categories and to gathering knowledge about a particular learner's profile, in order to adapt the delivery of knowledge to this profile.

The platform proved to be very efficient in personalization issues, because it created only one package able to handle *n* objects per *j* dimensions, displaying only the suitable material and allowing re-use/re-pack learning objects. One of the major drawbacks was that it required tedious metadata filling sessions. The reason was that the tool does not contain automatic generation of metadata derived from relevant ontologies and resource description formats [8]; nor does it contain editing tools to add structured metadata, such as the eXtensible Authoring and Publishing (XAP) Adobe metadata initiative for PDF formats.

Being a distributed platform for continuous learning it aims at using the Internet as an effective learning environment. Although medical researchers are often reluctant to trust Internet information mainly because it does not fulfil long-established verification criteria, the number of *on-line* Medical and Biology journals is increasing. Furthermore the availability of new techniques, lead us to consider the Internet as a future source of updated medical information. On the Internet, scientific papers can use handles [9] describing the physical location of the file (Universal Resource Names (URN) as part of the Universal Resource Identifiers) facilitating the search tasks. Moreover, the National Library of Medicine supported the Internet Engineering Task Force (IETF), in designing a quasi-permanent naming of web-based information objects. The Archival Resource Key draft

entitled *The ARK Persistent Identifier Scheme* [10] defines three ARK services to access: i) the object, ii) the description of the object (metadata), and iii) the commitment description, made by the Name Mapping Authority (NMA) regarding the persistence of the object (policy). On this context semantic webs use equally i) software Agents able to negotiate and collect information, ii) Markup Languages that can tag many types of information and iii) Knowledge Systems enabling machines to read web pages and determine their reliability [5].

In the medical field, Internet discovery tool innovations go from web Ontology Agents capable of retrieving information in an intelligent manner [11] to Medical Core Metadata (MCM) standardizing attributes and enhanced data to be used by agents [12]. Finally, to support free text queries, terms should be compared with established vocabularies; the free web resource HSTAT (Health Services/Technology Assessment Text) [13] accesses full-text document title, checking users spelling queries by means of software Agents based on Unified Medical Language System (UMLS) meta-thesaurus.

The success of any of these tools relies on the use of common ontologies. Medical terminologies are long-established foundational ontologies, allowing the retrieval of related and synonymous concepts, querying and cross-mapping multiple terminologies/ classifications at the same time. They culminate in the meta-thesaurus, which is a foundation product of the National Library of Medicine UMLS initiative[14], of which MeSH-2001 forms an essential part. It is a machine-readable knowledge source that represents multiple biomedical vocabularies organized as concepts in a standard format. Although it provides an immensely rich terminology in which terms and vocabularies become linked by a meaning, it does not include most telemedicine classifications, subheadings and qualifiers that require a new set of concepts. For that reason, the TM-BoK hierarchy [7] is essential for this application.

Until now the above mentioned tools, could assist users in the process of cataloguing hierarchic content relationships for a set of documents, but did not address personalization issues, which are vital for multidisciplinary topics such as telemedicine.

The present e-learning platform that retrieves personalized medical information according to users profile and goals is an example. Furthermore, since agent technology is fundamental for intelligent queries and data retrieval, it becomes necessary to build health care agents specialized in the various health services using specific taxonomies and adapted markup languages. In this respect, AgentCities started a Health Care Working Group [15] in 2002; actively working in 2003 [16] and 2004 [17], their work is not very apparent among agent technology specialists and particularly, in everyday medical applications[18].

In conclusion, the structure presented in this paper could create an Internet based distributed learning platform, with repositories placed anywhere. The system will keep the information on available learning objects/packages to access and retrieve information. Once loaded, the set of rules placed in the *modified*-XML-manifests stored anywhere, will be executed, presenting only data suited to users demands.

References

[1]Lowe H., Barnett G. "Understanding and using the medical Subject Headings (MeSH) vocabulary to perform literature searches". *The Journal of the American Medical Association,* 1994. vol. 271,: pp 1103-1108.

[2]IEEE Learning Technology Standards Comitee (2002). Learning Object Metadata. IEEE 1484.12.1-2002. Available: http://lttf.ieee.org/learn_tech/issues/january2003/index.html

[3]IMS Global Learning Consortium , INC. (2003, Oct 21) . *Open Specifications for Interoperable Learning Technology* . [Online] . Available: http://www.imsglobal.org/specifications.cfm

[4]Azambuja Silveira R., Vicari RM. Developing Distributed Intelligent Learning environment with JADE- Java Agents for Distance Education Framework. (2002, Jun 2-3) Intelligent Tutoring systems Cerri SA, Gouarderes G., Paraguacu F. Ed *Lecture Notes in Computer Science LNCS* 2002. vol 2363 pp 105-118, Springer-Verlag. Berlin. . [Online]. http://www.inf.ufrgs.br/~rsilv/publicacoes/SilveiraITS2002.pdf ; http://www.inf.ufrgs.br/~rsilv/Jade/jade.html

[5].Garro A. , Palopoli L. An XML Multi-Agent System for e-Learning and Skill Management. (2002). http://www.info.uqam.ca/~nkambou/DIC9340/an-xml-multi-agent.pdf

[6]Dublin Core Meta-Data Element Set, Version 1.1. (1999, Jul. 2) *Dublin Core Metadata Initiative*. [Online]. Available: http://purl.org/DC/about/element_set.htm

[7]Olga Ferrer-Roca, Marcelo Sosa-Iudicissa. "Handbook of Telemedicine". IOS-Press Publ., 1998.

[8]Thornely J. "The How of Metadata: Metadata creation and standards", (2003, Oct. 13). [Online]. Available: http://www.slq.qld.gov.au/pub/staff/catcon99.htm

[9]Chapman Ch., Brailsford D.F. (2001). "Navigating a corpus of journal papers using Handles". [Online]. Available: http://www.cs.nott.ac.uk/~clc/elpub 2001.pdf

[10] Diana Dack. "Persistent Identification Systems", (2001, May.). *National Library of Australia*. Available: http://www.nla.gov.au/initiatives/persistence/PIcontents.html

[11] HealthCyberMap.org, (2003, Jan. 10). [Online]. Available: http://healthcybermap.semanticweb.org/

[12] Malet G., Munoz F., Appleyard R., Hersh W. "A model for enhancing Internet Medical Document Retrieval with "Medical Core Metadata"". *Journal of the American Medical Informatics Association*, vol. 6, no. 2, pp, 163-172, 1999

[13] "Health Services/Technology Assessment Text", (2001, Nov. 30) *Computer Science Branch*. [Online]. Available: http://lhncbc.nlm.nih.gov/csb/CSBPages/HSTATproject.html

[14] Gu-H, Perl-Y, Geller-J, Halper-M, Liu-L-M, Cimino-J-J. "Representing the ULMS as an object oriented database: modelling issues and advantages". *Journal of the American Medical Informatics Association,* vol. 7, no. 1, pp. 66-80, 2000

[15] Moreno A., Nealon J. "Agentcities Working Group Proposal: Healt Care". (2002, March 12). Available: http://www.agentcities.org/in/00011/

[16] http://www.agentcities.org/Activities/WG/Healthcare ; First International workshop on the Application of Agent Technology to Health Care. Agentcities ID4 meeting. Barcelona (2003, August 22). http://wwwcms.brookes.ac.uk/hcwg/Programme.htm.

[17] Second workshop on Agents Applied in Health Care. ECAI-2004 . (2004, August 23) http://wwwcms.brookes.ac.uk/nealon/ecai/healthcare.html

[18] Moreno A., Garvey C. "Software Agents in Health Care" *Artificial Intelligence in Medicine* 2003. vol.27, no 3, pp, 229-232.

New Features of the OPScript Language

Vicente Arturo ROMERO ZALDIVAR
University of Cienfuegos
Carretera a Rodas Km 4, Cienfuegos, Cuba
Jon Ander ELORRIAGA ARANDIA
University of the Basque Country
Apdo. 649 P.K. - 20080 Donostia - San Sebastián, Spain
Mateo Jerónimo LEZCANO BRITO
University of Las Villas
Carretera a Camajuaní Km 5, Santa Clara, Cuba
Mikel LARRAÑAGA
University of the Basque Country
Apdo. 649 P.K. - 20080 Donostia - San Sebastián, Spain

Abstract. In this paper the authors make a brief introduction to the YADBrowser project which includes the educational browser YADBrowser and its script language, OPScript. The authors expose also some features added to it, with them the author of an educational application can achieve more functionality and adaptation with less code. Also the YADBrowser reduces the interactions with the server reducing the response time and keeps a record of the actions and preferences of the user. These new features were added to facilitate the creation of dynamical applications, adaptable to student skills. They include "verbal" communication between objects, XML object models and reusable methods.

1. Introduction

Internet has been highly used as an effective means for publishing information of any kind. The Web is also a natural field for educational hypermedia applications [1]. Nevertheless it is hard to construct context-based adaptive Web applications for many reasons: static links between documents, the stateless nature of the HTTP protocol, etc. In despite of these problems authors are always creating applications, models, languages etc., to offer adaptive educational applications to students according to theirs knowledge and skills [2, 3].

In order to create these educational applications it is very important to have languages which easies the work of the developers, it is also necessary to consider that sometimes authors of educational applications do not have a solid knowledge of programming languages. So every tool must be powerful and easy to use for a wide set of authors. Automatic generation of code and code reuse are also desirable characteristics for a new language so many features can be added to a given application with less work. There are some works related with the development of new browsers or script languages with educational purposes [4]. There are also reports of the development of browsers for general and specific domain applications. An interesting example is the Grendel browser [5], this browser allows the user to script the browser user interface, the browser's HTTP

interactions, and the browser's rendering of documents. To do so the authors define a language, CrossJam, similar to Lisp, using it the user can perform a great degree of customization. Other authors have notice that collaborative browsing is more valuable than lonely browsing for educational purposes, so there are reports of tools for collaborative browsing [6]. Another issue is browsers adaptable to computing resources, location of the user, etc, [7].

There are other issues between authors; one of the most important of them is the production of frameworks which facilitate the creation of educational applications using current technologies. In this field, for example, the Avante architecture [8], and the MTeach framework [9] can be found. Another important issue is semantic relationships and consistency in hypermedia. There is also great interest in lesson components and courseware reusing [10, 11].

2. Brief Introduction to the OPScript Language and the Educational Browser YADBrowser

The OPScript language [12, 13, 14] is the script language of the educational browser YADBrowser which main objective is to develop a browser suitable for educational applications with a language that allows a fast development of these applications and other facilities like persistence of selected information in browser memory, lesson components reuse, etc. Some authors have stated the advantages of saving information in the main memory of a given browser [15]. Persistent additions can be very useful in educational applications, because they reduce the response time, the code needed to create a given application, etc.; for this reason the YADBrowser and the OPScript language includes many features to persist information in browser memory. When information is saved in browser memory, bandwidth and time can be saved. There is a great interest in bandwidth saving and many techniques including sharing information between browsers have been reported [16, 17]. Also, the YADBrowser includes an object model with a wide collection of objects adapted to educational applications; there are reports of the development of object models for Web applications to save effort and time [18].

An important member of the object model of the YADBrowser is the *TUser* class, only one object of this class is created for session and it is available in every page using client side code. This object keeps user generated information such as pages visited by him or her, score obtained in the solution of exercises and other information related to the user profile. Also the *TUser* class defines methods to add application specific information related with the user and to later request that information. This information can be useful to decide what content must be shown to the user, and content-based adaptive applications can make use of it to display information according to user knowledge and skills. For example the target of a given link could be redefined according to the level of a given user, beginner, intermediate or expert, or a graph of actions necessary to operate a medical equipment can vary according to the level of the user and to a given pathology selected by him. This information can be present in the *User* object and used on demand, by the way interactions with the server can be reduced because the information is always present in the browser overcoming this way the stateless nature of the HTTP protocol.

Another important aspect is the markup language used by the YADBrowser which is a subset of HTML 4.01 [19]. Some additions have been made to this subset for achieving a better adaptation to educational applications requirements. Between these additions the more important is the pattern tag which allows defining some content that should be reused in several Web pages, this way it is possible to define a content once and use it several times, maybe adapting it when necessary. Patterns are another way of defining persistent

information. Some authors have created new markup languages for domain specific applications some of them are XML based languages [20], the language defined by these authors, named MeML, was created for publishing mathematical content on Internet.

Initially the OPScript language was defined as a language without types, like JavaScript [21], nevertheless at present it defines the following types: integer, string, boolean and object. With these types it is possible to check better the correctness of programs, which is desirable even when code complexity can increase a little. The language is similar to Object Pascal but has some elements of the C++ language, for example variables declaration. OPScript has constructions to define what classes, methods or fields has to persist to be used in a coming page, decreasing this way the bandwidth needed and also the time to respond to a user generated event.

The experience obtained with the development of this project could show what characteristics are desirable in a language and a browser created specially for educational applications or what should be included in a standard browser to make it more suitable for educational applications.

In this article some features included lately in the OPScript language will be exposed, which make the language suitable to develop dynamical educational applications, reduce the amount of code necessary to implement them and the programming skill needed to create a given project.

3. Metadata and Reusable Methods

Metadata have proven to be very useful when used in programming languages, even in aspect oriented programming (AOP), they have proven to be very useful, for references about AOP see [22, 23]. They allow annotating a field, a class, a method, etc., metadata can be obtained later by reflection and special treatment can be done to a given programming element according to the metadata applied to it. In OPScript metadata can be defined by placing it in brackets before the element to be annotated. This is shown in the following fragment of code:

```
TStudent =
  class
    string [level] fLevel;
  end;
```

Later if it is needed to retrieve a field annotated by a metadata it can be used a set of functions present in every object, because they belong to the *Object* class and in OPScript every class inherits by default from this class. In OPScript metadata are the main support to the reusable method concept. A reusable method is one that can be called by objects of different classes, no matter which is the original class that defines the method. Using the sample code above the field could be retrieved inside a reusable method like this:

```
var
  string s;
begin
  ...
  s := GetStringFieldVal('level');
  ...
end;
```

The function used above to retrieve the field is applied to the calling object, no matter its class. When a reusable method is called it is executed as if it were a method of the

calling object class. Actually what happens is that a class lends a method to another class and, no matter this fact, everything works properly. This can be confusing, but is very useful. To make a method reusable, it can not access the fields of the current object directly, but through metadata. Reusable methods decrease the amount of application code, making it simpler and they are an important component of the feature exposed below, “verbal” communication between objects.

4. Verbal Communication between Objects

This new feature makes possible the interaction of two objects without additional programming and with almost no knowledge between the objects. The relation between the objects is created automatically whenever it is necessary by the browser. So the creation of a Web application can be reduced, at least an important part of the process, to the inclusion of some objects in several pages and allowing the browser to link them automatically. With verbal communication, two objects can interact even if the interfaces of one or both objects change or even if any of them changes almost completely.

Verbal communication is important because it speeds up programming and makes the development of useful and real educational applications easier to people with less knowledge of programming languages. The following sample code shows how verbal communication works in OPScript:

```
TExercise = class
  needs Play;
  fields
    string [mediafile] fPath;
  methods
    procedure PlayMedia();
    begin
      ExecuteVerb('Play');
    end;
end;

TPlayer = class
  methods
    procedure DoPlay();
    var
      string Path;
    begin
      Path := GetStringFieldVal('mediafile');
      ...
    end;
  offers Play(DoPlay);
end;
```

In the sample code above, with the use of a hypothetical example, it is shown how two classes can communicate with each other by using verbal communication. See the use of the keywords *needs* and *offers*, the *TExercise* class needs to interact with a class offering the *Play* verb, as long as the *TPlayer* class offers it, the link between both classes is made automatically. In the *offers* part after the verb and in parentheses, it must appear a method name, this is the method that will be executed when the verb is required.

See also that this method has no parameters, to access the information contained in the calling object at execution time the metadata are used. This facilitates the communication with different objects of different classes, no matter their types or interfaces; of course at execution time there can be problems if an object does not have a

field annotated with the correct metadata, but the programmer can prevent multiple possibilities and in the worst case to do nothing appear to be the better option. Finally, see that the *ExecuteVerb* method present in every object is the trigger to make the link with an object that offers a given verb. This method can be called in response to a user generated event or when the application reaches a given state.

It is important noting that any class can be included dynamically in the object model in response to a user generated event, the download of a new page, etc., anytime a new class is added to the object model the browser automatically looks for matching between classes offering verbs and classes needing them.

Due to verbal communication a given object can interact easily with more than one object during the lifetime of the application, without noting it. This can facilitate the adaptation of an educational application to the skills and knowledge of a given student, because a given object without any change can interact with different objects which have different information or behavior depending on the answer to a question or the selection of the correct option in a given exercise or situation exposed to the student by an educational application.

Verbal communication between objects reduces development time and complexity. It is known that the development of software components can considerably reduce the development time of any application and its complexity, now with verbal communication the objects can define how the communicate with each other, without the need of the developer knowing exactly how they interact, the parameters that are necessary to pass and what to do with the resulting data. Course, verbal communication has to evolve, but it offers many facilities which make it valuable in the development of educational applications.

Suppose for example that a developer with not too much experience has a wide set of classes, developed by experienced programmers, these classes can communicate with each other by using verbs; his work is reduced a great deal because verbal communication reduces the time to tie the model and the amount of errors that can appear supposing that the original set of classes does not have errors. As a side effect, objects that communicate with other objects by verbal communication are simpler and can be more easily reused than objects that do not.

5. XML Object Models

The YADBrowser allows extracting object models represented in files using the XML language. XML has many properties which make it suitable to represent object models: it is a well known language, simple even for people with less knowledge about computing technologies; it has an intrinsic hierarchical representation, etc. So XML is a natural language to represent object models which can take advantage of XML hierarchical representation to reduce the amount of declarations needed to express the interrelations and properties of a given object model.

To extract an object model from an XML file the YADBrowser follows the following conventions:

- Every XML node will be converted into an object of the language, every attribute of the node will be a field of the object with the value and type of the attribute, this way is almost unnecessary to implement explicitly a constructor for a given object.
- If in the XML file a node named “node1” belong to another node named “node2” the object corresponding to “node2” will be the owner of the object corresponding to “node1”.
- For all the nodes with the same name it is created a class which represents them all. Every class declares a list of objects in case it is needed to contain any object,

especially objects corresponding with XML nodes. All lists have the same name, and methods to make operations with the list like adding, deleting, inserting, etc., are included in every class automatically, reducing the amount of code that should be included in the XML file to define the behavior of the model classes.

- Every node should include a property named “id”, which will be the name of the resulting object. If any XML node includes a property which value is the “id” of any object this will be interpreted as an existing relation between both nodes which will be represented in the resulting object model. So the topological structure of the model is constructed automatically.

XML files representing object models can be downloaded at any time according with application needs. To download an object model from a XML file dynamically can be done in OPScript with a piece of code as the following:

```
var
  Object Graph;
begin
  ...
  Graph := XMLModel.LoadModelFrom('XMLModels\BeginnerGraph.xml');
  ...
end;
```

In the sample code above “XMLModel” is an object of the OPScript language which has the method “LoadModelFrom”, it receives the path of a XML file and returns an object that corresponds with the root node of the XML file. To represent object models in XML files has many advantages:

- The dynamical addition of object models to the current application model. Initially the main components of a page can be downloaded and later a model which can vary according with applications needs, student skills or both, can be downloaded on demand.
- It is necessary to code less because of the advantages of the XML language mentioned above.
- It is a simpler way of defining an object model so persons with less knowledge of programming languages should find easier to define an object model using XML than using an imperative programming language.

Let us see how a model defined in a XML file looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<graph id='Root' CurrentNode='Start'>
  <methods>
    <!--
      procedure ProcessEvent();
      var
        integer i, event;
        Node Current;
        Link tmpLink;
        integer terminate;
      begin
        terminate := 0;
        event := GetIntegerFieldVal('eventdata');
        Current := this.CurrentNode;
        i := 0;
        while (i < Current.fItemsList.Count()) and (terminate = 0) do
          begin
            tmpLink := Current.fItemsList.Objects(i);
            if tmpLink.Event = event
              then
```

```
                begin
                  terminate := 1;
                  this.CurrentNode := tmpLink.Node;
                end
              else i := i + 1;
          end;
        if i = Current.fItemsList.Count() then Messages.Alert('Incorrect
Action');
      end;
      procedure Reset();
      begin
        this.CurrentNode := this.fItemsList.Objects(0);
      end;
    -->
  </methods>
  <verb name='ChangeEvent' offers='true' method='ProcessEvent' />
  <verb name='VReset' offers='true' method='Reset' />
  <node id='Start' name='StartNode'>
    <link id='LStartOn' node='On' event='1' />
  </node>
  <node id='On'>
    <link id='LOnPulsoElectrodo' node='PulsoElectrodo' event='10' />
    <link id='LOnOff' node='Off' event='5' />
  </node>
  <node id='PulsoElectrodo'>
    <link id='LPulsoElectrodoElectrodo' node='Electrodo' event='11' />
    <link id='LPulsoElectrodoOff' node='Off' event='5' />
  </node>
  <node id='Electrodo'>
    <link id='LElectrodoSincronismo' node='Sincronismo' event='12' />
    <link id='LElectrodoOff' node='Off' event='5' />
  </node>
  <node id='Sincronismo'>
    <link id='LSincronismoEnergia' node='Energia' event='3' />
    <link id='LSincronismoOff' node='Off' event='5' />
  </node>
  <node id='Energia'>
    <link id='LEnergiaEnergia' node='Energia' event='3' />
    <link id='LEnergiaPaleta1' node='Paleta1' event='6' />
    <link id='LEnergiaOff' node='Off' event='5' />
  </node>
  <node id='Paleta1'>
    <link id='LPaleta1Paleta2' node='Paleta2' event='7' />
    <link id='LPaleta1Off' node='Off' event='5' />
  </node>
  <node id='Paleta2'>
    <link id='LPaleta2Carga' node='Carga' event='8' />
    <link id='LPaleta2Off' node='Off' event='5' />
  </node>
  <node id='Carga'>
    <link id='LCargaDescarga' node='Descarga' event='13' />
    <link id='LCargaOff' node='Off' event='5' />
  </node>
  <node id='Descarga'>
    <link id='LDescargaSincronizar' node='Sincronismo' event='12' />
    <link id='LDescargaOff' node='Off' event='5' />
  </node>
  <node id='Off' />
</graph>
```

The fragment above shows a model created for a real application. It defines a graph, its nodes and the relations between them. The graph represents the steps a doctor must

follows to employ a medical equipment used in case of a heart attack or other heart pathology. The graph is added to the application dynamically and is used to know if the user is following the correct steps for the equipment to function properly. The application, under development at present, is a tutorial to teach how to use the medical equipment. In the sample above *fItemsList* is the list automatically added by the browser, see also the methods it has, *fParent* is a field present in every object to access its owner and *Current* is a field that points to a given child, all of them are conventions that follow the YADBrowser when it loads a model from a XML file done to reduce the amount of code needed to define the model. See also in the sample above how it is possible to define a method or a set of methods that belong to a XML node.

6. Educational Importance of the New Features added to the OPScript Language

The new features of the OPScript language exposed in this paper can help authors building adaptive-hypermedia-based systems, but before seeing how this is possible it will be explained how all the features commented in this paper interact between them. The sample code above is a XML file that represents an object model, so it is an XML object model. In this file it is possible to see two verb tags. They are verbs offered by the *Graph* class defined in the XML file. These verbs offer theirs behaviour through two methods: *ProcessEvent* and *Reset* the first one looks for a valid node in the graph after the generation of an event by the user, the second restarts *Start* as the current node, this can be necessary to start an exercise again, etc., these methods are reusable methods and are lend by the *Graph* class to a class, not represented here, which will interact with the *Graph* class. To do so the counter part does not have to know what methods are offered by the G*raph* class, in fact it not even knows that it is interacting with a class named *Graph*, at a given moment it could continue interacting with another class which represents the valid answers to an exercise in any way, in fact in the real application it is possible to interact with different *Graph* classes according to the level of the student and to a previously selected pathology. Considering that the XML files can be downloaded on demand it is possible using these features to obtain a great degree of adaptation and in a dynamic way.

The idea behind this is that using the features exposed in this paper it is possible to, in a Web application, send to the student a page with selected static information, static in the sense that it is since the moment it arrive, present in the browser, and later according to her preferences or to some other characteristics it is possible to download an object model, that will be added to the already downloaded model, and will interact with it transparently. The so downloaded object model can be obtained using the user model represented in the server side of the application. In the sample code shown above that XML file corresponds to a beginner student, so the object model downloaded dynamically is related with the user model of the given student. The rest of the client side application, the one that has been downloaded previously is able to interact with this XML model or with a model created to an expert student; its programming, definition, etc. can be the same. So the difference is in the XML model and what makes possible the interactions of the same objects with different models no matter the level or preferences of a given student are: “verbal” communication between objects and reusable methods.

The advantages for educational applications are the adaptation that can be obtained, the dynamism, because a XML model can be downloaded at any time without changing the current page, without a noticeable delay for the student and because the part that depend most on the student can be added to the application at any moment, it can be generated dynamically no matter what has been already send to the student through Internet because the communication using verbs is so flexible that it can adapt itself to almost any scenario.

7. Conclusions

In this paper authors have exposed the new features added to the OPScript language, these features have been added to achieve more adaptation using techniques in the client side of a Web application, this project could benefit a lot if it could be joined to models that could achieve the adaptation from the server side of a Web application, created to employ OPScript as the client side script language, like the LAOS model [24].

Authors have shown that applications could be more easily adapted to students' skills and knowledge using the features exposed, because by using verbal communication a given object can interact with different objects in different moments during the application lifetime without special coding. By downloading whole object models represented in XML files the application can be adapted easily to the necessities, responses, etc., of the student. Code added automatically to those object models reduce the effort needed to develop the final application. Finally the features mentioned work based upon the reusable method and metadata features, which make method sharing and reusing between classes possible.

In this paper it have been included some portions of code of a developing project. This project will show to doctors and paramedics how to operate a medical equipment, this project is been created using all the features of the OPScript language mentioned in this paper.

References

[1] Montessoro, P., Pierattoni, D., Cicuttini, R.: MTeach: A Simple Production Framework for Context-Based Educational Hypermedia. Journal of Educational Multimedia and Hypermedia 12(4) (2003) 335-359.

[2] Manouselis, N., Sampson, D.: Dynamic knowledge route selection for personalised learning environments using multiple criteria. In Proceedings of the IASTED International Conference on Applied Informatics, (2002) 351–605.

[3] Atif, Y., Benlamri, R., Berri, J.: Learning Objects Based Framework for Self-Adaptive Learning. Education and Information Technologies 8(4), (2003) 345–368.

[4] Huang, R., Ma, J.: A Java Technology Based Shared Browser for Tele-Lecturing in University 21. [Online]. Available: http://csdl.computer.org/comp/proceedings/iccima/2001/1312/00/13120298.pdf (2001).

[5] Dennis, B., M., Harrison, M., A.: Grendel: A Web Browser with End User Extensibility. [Online]. Available: http://csdl.computer.org/comp/proceedings/compcon/1997/7804/00/78040074.pdf (1997).

[6] Hoyos-Rivera, G., J., Lima-Gomes R., Courtiat, J., P., Benabbou R.: The Web as a Tool for Collaborative *e*-Learning: the Case of CoLab. [Online]. Available: http://csdl.computer.org/comp/proceedings/icalt/2003/1967/00/19670312.pdf (2003).

[7] Henricksen, K., Indulska, J.: Adapting the Web Interface: An Adaptive Web Browser. [Online]. Available: http://www.dstc.edu.au/m3/papers/AUIC2001.pdf (2001).

[8] Theoktisto, V., Bianchini, A., Ruckhaus, E., Lima, L.: AVANTE: A Web Based Instruction Architecture based on XML/XSL Standards, Free Software and Distributed CORBA Components. UPGRADE 4(5), (2003) 29-38.

[9] Montessoro, P.L., Pierattoni, D., Toppano, E.: MTEACH: A simple framework for didactic and context-based hypermedia. Proceedings of SSGRR 2002W, International Conference on Advances in Infrastructure for Electronic Business, Science and Education on the Internet (2002).

[10] Boyle, T., Cook, J.: Towards a pedagogically sound basis for learning object portability and re-use. In: G. Kennedy, M. Keppell, C. McNaught, T. Petrovic (eds.): Meeting at the Crossroads. Proceedings of the 18th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education. The University of Melbourne. (2001) 101-109.

[11] Boyle, T.: Design principles for authoring dynamic, reusable learning objects. Australian Journal of Educational Technology, 19(1), (2003) 46-58.

[12] Romero, V. A., Mateo, L., Elorriaga, J. A.: The educative browser YADBrowser and its script language OPScript. In: Actas del II Congreso Internacional de Tecnologías y Contenido Multimedia. Tecnología y aplicaciones Web. La Habana (2004).

[13] Romero, V. A., Elorriaga, J. A., Mateo, L.: The Language for the Educational Browser YADBrowser. In: Actas del 6º Simposio Internacional de Informática Educativa. Cáceres (2004).

[14] Romero, V. A., Elorriaga, J. A., Mateo, L.: YADBrowser: A browser for Web based educational applications. To be published by Journal of Educational Multimedia and Hypermedia.
[15] Uehara, S., Mizuno, O., Kikuno, T.: An Implementation of Electronic Shopping Cart on the Web System using Component-Object Technology. [Online]. Available: http://www-kiku.ics.es.osaka-u.ac.jp/paper/data/pdf/10.pdf (2001).
[16] Xiao, L., Zhang, X., Xu, Z.: On Reliable and Scalable Peer-to-Peer Web Document Sharing. [Online]. Available: http://csdl.computer.org/comp/proceedings/ipdps/2002/1573/00/15730023b.pdf (2002).
[17] Xiao, L., Zhang, X., Andrzejak, A., Chen, S.: Building a Large and Efficient Hybrid Peer-to-Peer Internet Caching System. [Online]. Available: http://csdl.computer.org/comp/trans/tk/2004/06/k0754.pdf (2004).
[18] Hennen, D., S., Ramachandran, S., Mamrak, S., A.: The Object-JavaScript Language. [Online]. Available: http://acuity.cis.ohio-state.edu/Nois/Tguide/ojs.ps. (2000).
[19] W3C HTML 4.01 Specification (n. d.). [Onbline]. Available: http://www.w3.org/TR/html4/.
[20] Wang, P., S., Kajler, N., Zhou, Y., Zou, X.: WME: Towards a Web for Mathematics Education. [Online]. Available: http://ox.cs.kent.edu/~pwang/47wang.pdf (2003).
[21] Netscape Corporation. JavaScript Central (n. d.). [Online] Available: http://devedge.netscape.com/central/javascript/.
[22] Shonle, M., Lieberherr, K., Shah, A.: XAspects: An Extensible System for Domain-Specific Aspect Languages. [Online]. Available: http://www.cs.ucsd.edu/users/mshonle/p28-shonle.pdf (2003).
[23] Lieberherr, K. J.: Connections between Demeter/Adaptive Programming and Aspect-Oriented Programming (AOP). [Online]. Available: http://www.ccs.neu.edu/home/lieber/connection-to-aop.html (2004).
[24] Cristea A. I.: Automatic Authoring in the LAOS AHS Authoring Model. [Online] Available: http://wwwis.win.tue.nl/ah2003/proceedings/ht-2/

Acknowledgements

This work is partially funded by the University of the Basque Country (UPV00141.226-T-15948/2004), the Spanish CICYT (TIC2002-03141) and the Gipuzkoa Council in a European Union program.