Formalising the Design Process of Web-based Adaptive Educational Applications using an Object Oriented Design Model

Simos Retalis[1], Andreas Papasalouros[2], Manolis Skordalakis[2]

[2]Computer Science Department
University of Cyprus
P.O. Box 20537 Nicosia, Cyprus
Tel: +357 22 892246
Fax: +357 22 339062
retal@cs.ucy.ac.cy

[2]Software Engineering Laboratory
National Technical University of Athens
15780 Zografou, Athens, Greece
Tel. +302 10 7722486
Fax. +302 10 7722519
{andpapas,skordala}@softlab.ntua.gr

Abstract

The work presented in this paper is comprised of a visual design model for adaptive hypermedia educational systems that follows the principles of the object-oriented paradigm. The model is primarily originated on the Object Oriented Hypermedia Design Method, it has influenced by the AHAM. The Unified Modeling Language (UML) serves the purpose of notation syntax and semantics for this model. The theoretical analysis of the model is accompanied by a case study application for the design of an adaptive web-based testing system.

Keywords: Adaptive Hypermedia, Educational systems, visual design modeling

1. Introduction

It is generally agreed that there is a need for modeling of hypertext and hypermedia applications [17]. In contrast to generic software engineering, where significant progress has been made the past twenty years, there is still a great deal of work to be done on formalizing process models, and defining methodologies or design methods for hypermedia applications. These models can be reference models, facilitating the common understanding of the structure and behavior of particular hypertext applications, or design models that facilitate the process of design and development of such applications.

In this paper we propose a design model for *Web-based Adaptive Educational Applications*. A *Web-based Adaptive Educational Application (WAEA*) is defined as a dynamic web-based hypertext application, i.e. a set of dynamically generated web content, which provides a learning environment to its users. This environment comprises electronic content for study as well as a set of tools that facilitate the study of a learner such as web-based questionnaires, glossaries, communication tools, etc. This model focuses on content, which is considered as hierarchically structured, usually dynamically created, personalized assembly of predefined learning resources, either created from scratch or reused. These resources can be available in any form such as files, database entries, etc.

Our model is based on the Unified Modeling Language [18, 19] that is a standard, extensible formalism for visual object-oriented modeling. A design model like this can be used as a framework [1, 10] for authors of hypertext applications to develop and apply methodologies in order to create adaptive educational applications (and not general purpose) in a disciplined and controlled fashion. It incorporates the principle of separation of concerns in the design of hypermedia applications, dividing the design of the application in three stages: conceptual, navigational and presentational. We also claim that this separation of concerns aligns with the three types of adaptation, navigation and presentation. Beyond a design model, if the development of open, portable, maintainable WAEA is to be facilitated, there is a need for a formally specified description of the WAEA. This description must be automatically generated from the aforementioned design model, at least to an extent, and must be easily ported to specific run-time environments that will deliver the specific WAEA.

This model has been built with the following requirements that in mind [16]:

1. *Formalisation*: its notation system must describe a WbEA and its constituents in a formal manner
2. *Completeness*: its notation system must be able to fully describe a WbEA, including all types of its constituents, the relationships among them and their behavior
3. *Reproducibility*: its notation system must describe a WbEA and its constituents in an abstract level so that repeated execution/adoption is possible for specific subject domains
4. *Compatibility*: its notation system must fit in with the available standards and specifications (IMS, IEEE LTSC, SCORM, etc.)
5. *Reusability:* its notation system must make it possible to identify, isolate, decontextualize, exchange and re-use constituents of a WbEA.

The rest of the paper is structured as following: In section 2, an overview of the three steps in the design process as well as the main components of the model are described. In section 3 follows an analysis of the components of the model giving more emphasis on conceptual design. In section 4, this model is compared to other models and some concluding remarks are given.

2. CADMOS-D: A Hypermedia Design Method for WEHA

CADMOS-D (design) is a method for the creation of the detailed design of a web-based educational application, which includes structural details of the learning resources, the navigational schema and templates for describing abstractly the graphical user interfaces. This method follows the principles of the object oriented hypermedia design method (OOHDM) [17]), which has provided systematic ways to design generic hypermedia applications and not especially educational ones.

CADMOS-D method proposes a stepwise design process, as shown in Figure 1: Conceptual Design, Navigational Design and Interface Design. The intermediate products of each step are validated according to guidelines for formative evaluation of the instructional design (checking structural, navigational, aesthetics and functional issues). The whole design process is considered to be iterative, where in each iteration loop the design artefacts are evaluated and the feedback from the evaluation is used for their improvement, until they reach the desirable level.

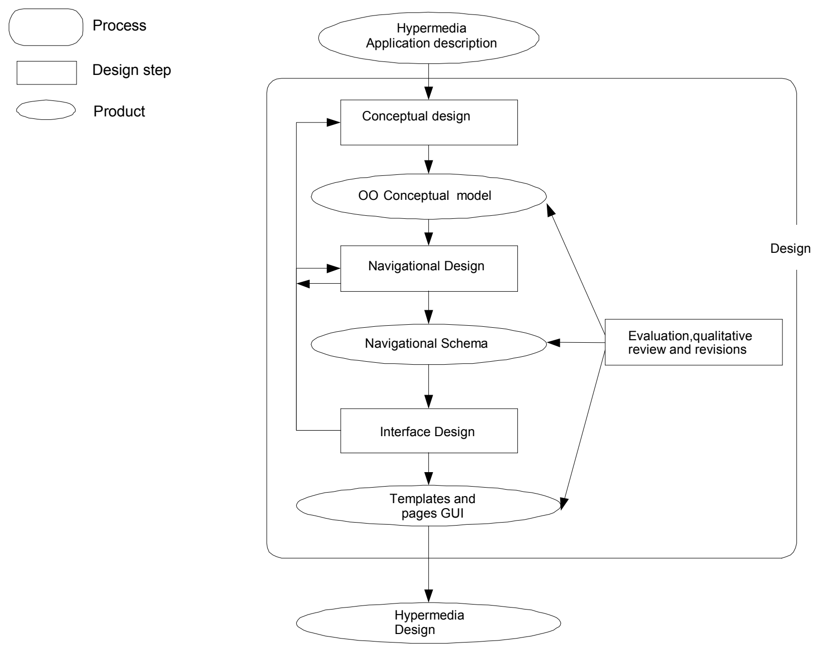

Figure 1. – The three design steps.

As facilitator and framework in the design process, a UML based model has been used. This model consists of the following components: Domain Model, User Model, and Teaching Model. Each one of these is a logical group of model elements and is represented as a package, the standard UML grouping mechanism, as shown in Figure 2.

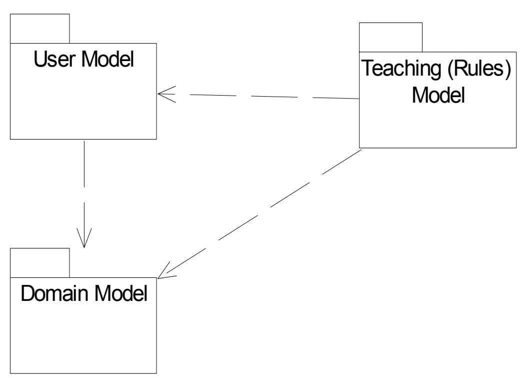

Figure 2. The models and their dependencies

The *Domain model* defines the concepts of the subject that is going to be taught with their semantic interrelationships. It can be considered as the Ontology [22] of the subject to be learned by the students. The Conceptual Model provides an objective definition of the knowledge subject. This definition is provided by the author of the educational application who is considered as a subject matter expert.

The *User Model* consists of two different parts, each one containing two types of elements: The *Overlay Model*, which is the domain specific part of the user model and defines the status of the learner's knowledge of the specific concepts covered by the learning material. The state of this model is frequently updated as a result of the learner's interaction with the learning content, for example the reading of learning material, the taking of an on-line test, the interaction with simulations, etc. The knowledge is defined as a structure of concepts (schema) and this structure is built during the user's learning activities. The Overlay Model depicts the system's awareness of the current status of the user's knowledge about the domain of the specific application as it is stated in the Conceptual Model. The elements of this sub-model are called *UserScheme* [2], and there can be one *UserScheme* element for each class of the Conceptual Model.

The second part of the User Model defines elements that are used to represent the usually predefined user knowledge profile either concerning the knowledge of the particular domain (novice, intermediate, expert, etc) or corresponding to the user's preferences or learning style. According to [3, 4] this constitutes the Stereotyped User Model. The elements of this submodel are called *User.*

The *Teaching Model* contains rules as Object Constraint Language [23] expressions applied to the appropriate UML elements, mainly classes. Constraints are conditions that must hold for the specific model they are applied. OCL is a formal language for applying constraints to UML models. OCL is a language for the specification and not the implementation of particular systems. The rules defined in our Teaching model are applied as two types of constraints:

- *Invariants*, that is conditions that must *always* be true in the context they are applied (concept components, concept relationships).
- *Postconditions*, that are conditions that must be met *after* the execution of a method or operation of a specific class.

The constraints are applied to specific model elements, defined by the keyword **context,** as will be shown in the following example.

Based on these three models, the designer is called upon to create the conceptual, navigational and presentation designs of the WEAH.

2.1 The Conceptual sub-Model

The *Conceptual sub-Model* defines the concepts of the subject that is going to be taught with their semantic interrelationships. The main entity of the conceptual model is the C*oncept,* which depicts a main idea or topic of interest into the educational application. Concepts are abstract entities that do not carry actual content by themselves. They can contain meta-data or other descriptions, but the actual content is defined in the associated Resources. The *Resources* are the actual fragments of content that compose the WAEA, text, images, sounds, videos, etc, which are static, reusable components or dynamic components. Two (or more) concepts can be associated with *Relationships,* which capture the semantic links between these concepts. Both concepts and relationships in the Conceptual Model are described as attribute-value pairs.

2.2 The Navigation sub-Model

The Navigation sub-model captures the decisions about how Concepts, Relationships and Resources of the Conceptual Model are mapped to actual hypertext elements Pages and Links, and how the conceptual relationships defined in the Conceptual Model are driving the structuring of the learning content. The *Navigation sub-model* is composed by two other sub-models:

2.2.1 The Navigation Structure sub-Model

This model defines the structure of the WAEA and defines the actual web pages and the resources contained in these pages.

This structure is composed of the following elements:

- *Content,* which is the top-level container in the hierarchy of an electronic content organization.
- C*omposite* entities that are used as containers, thus composing the hierarchical structure of learning content. The chapters and subtopics in which an electronic tutorial or book are organized are examples of composite entities.
- *Access structures* elements, namely *indexes* and *guided tours,* which are related to Content or Composite components
- *ContentNodes*, which are the actual pages of the learning content. Content, Composite and ContentNodes are associated with Concept elements, or directly with Resources, in the Conceptual Model.
- *Fragments* that are contained into the ContentNodes. Fragments correspond to Resource elements in the Conceptual Model.
- *Links* between ContentNodes as well as between Fragments. Note that these links are associative links [10, 20] implementing domain specific relationships of the conceptual model. They are not structural links denoting, for example, the transition from a page in the learning content to the next one.
- Composite, ContentNodes, Fragments and Links have a predefined attribute of Boolean type named *included*. This denotes whether or not a specific element (and all its descendants in the hierarchy) is included in the created hypertext or not, as a result of adaptation.

2.2.2 The Navigation Behavior sub-model

The *Navigation Behavior sub-model* defines the run-time behavior of the WAEA. Earlier research [7, 14, 24] has proposed the use of statecharts for the modeling of hypertext and web based applications. The Navigation Behavior model uses statecharts, as they are incorporated in the UML in order to specify the dynamic transitions of the hypertext structures as the user interacts with the WAEA. Every containing element of the Navigation Structure Model (Content, or Composite) is associated to a composite state in the Navigation Behavior Model, while every ContentNode corresponds to a simple state. Thus, the hierarchy of the navigational elements defined in the Navigation Structure Model corresponds to the hierarchy of nested states in the Navigation Behavior Model. The events that fire the transitions in the Navigation Behavior Model correspond to structure links into the ContentNodes: next, previous, up level, etc. In addition, guard conditions in these transitions can define alternative navigational transitions, which correspond to conditional behavior of the WAEA, thus implementing content sequencing and adaptive navigation.

2.3 The Presentation sub-model

The *Presentation sub-model* deals with the presentation aspects of the elements defined in the Navigation Model.

The presentation model is by itself separated in two additional sub-models: *Presentation Structure sub-model*, which defines the allocation of the navigational elements to actual user interface web elements: Web pages, frames, framesets, etc. Elements of this model, which is a variation of the synonymous model proposed in [14], are the following: *frameset*, *frame, window.* The aforementioned elements are associated with one or more elements of the Navigation Model.

User Interface sub-model, that captures the layout, colors, styles, etc of the entire web pages as well of atomic elements of the pages. This model consists of *Presentation* elements, which define the layout and style of associated elements of the navigation model.

4. A case study: An adaptive web-based testing system

We exemplify the design model by applying its concepts to an adaptive web-based testing system that conforms to the IMS QTI standard for question and test interoperability [http://www.imsproject.org]. The system enables either the editors to create/edit questions and tests (multiple choice, fill-in the blanks, etc.) or the simple users to be assessed answering to a series of questions of a test. The system supports adaptive exercise sequencing [7], customizing the succession according to which the questions are launched to the user. The answer to a particular question (right or wrong) changes the sequence of the test questions according to specific simple rules. The testing system was developed using the Java technology [http://java.sun.com], as applets, and utilizes the Extensible Markup Language [http://www.w3c.org/xml] for data storage. An example of the graphic user interface for a question is depicted in Figure 3.

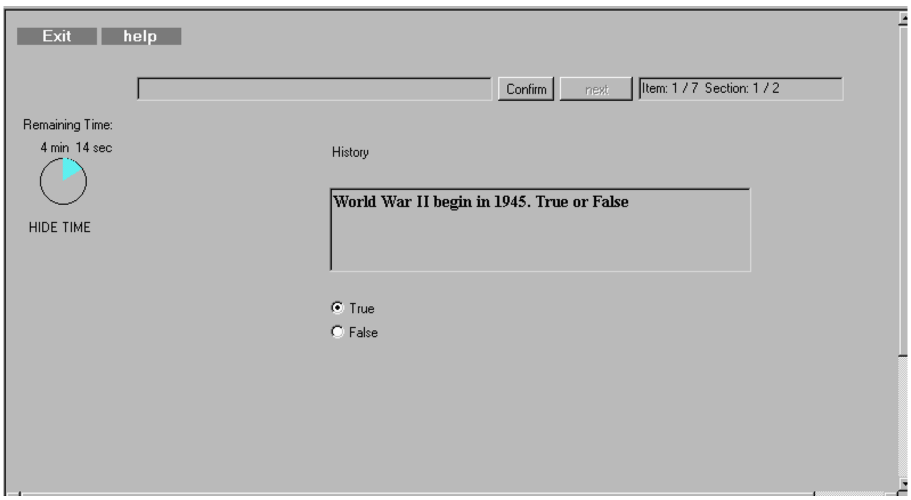

Figure 3. An example of the user interface

In the example of Figure 3, a section of the test on the History of the World War II contains seven questions. For the three first: question1, question2 and question3 we have applied the following simple rule: If a user answers correctly to the first question, the system skips question2 and immediately presents question3, else it continues with question2. As shown in Figure 3, the user can confirm his answer and then the button next is active, presenting a link to the next question.

In Figure 4, the conceptual design model of this part of the system is presented. In the same figure, we show some conventions that hold in the naming of the roles in the associations between the model elements. For example, when a concept component (e.g. the page entitled "question1") is associated with a UserModelScheme element, the role name of the UserModelScheme is, by convention, "user". An anchor is connected with its containing component with an association role named *uid*, which is the component's unique identifier.

We will show how the previous rule can be defined in the model with OCL expressions (constraints) applied to the appropriate contexts. For the first constraint we have:

context question1**::**confirmed()**: void**
  post: self.answer= self.correct_answer_answer **implies**
      self.user.answered=TRUE
    and self.answer <> serlf.correct **implies** self.user.answered=FALSE

This is a postcondition applied to the operation *confirmed* of the resource "question1". The operation named *confirmed* is the default operation that is called whenever the user presses the

Comfirm button while *accessed* is the default operation that is called whenever a component is accessed during the user's navigation. This kind of relationship between a domain model component and its corresponding user interface element is typically defined with state transition diagrams. It defines that if the user gives the right answer, then the corresponding user model attribute *answered* is set to TRUE, else it is set to FALSE.

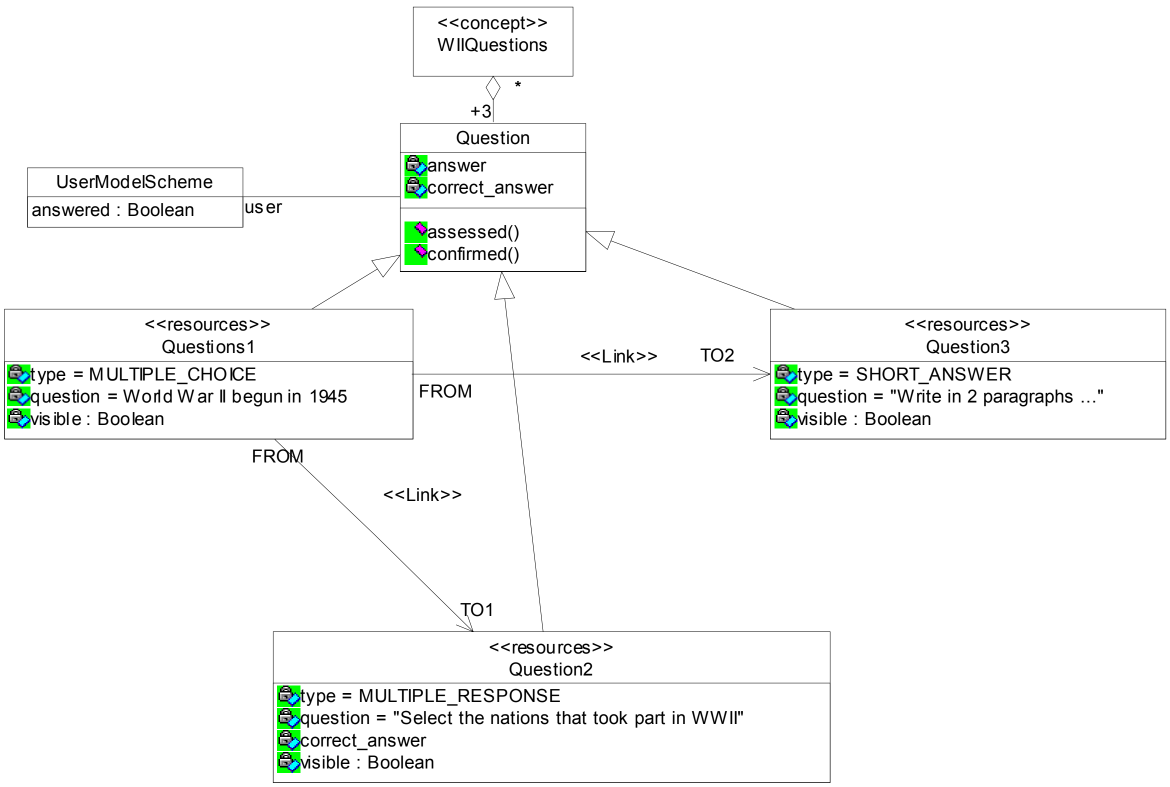

Figure 4. The adaptive question system example

The second concerns the links between the first question, named 'Question1' and the next question, i.e. 'Question2' and 'Question3'. If Question1 is correctly answered then Question2's presentation specification attribute *visible* would set to SHOW and this question would be presented to the user. Else if the answer to question1 is wrong for the specific user then the opposite would happen, that is the Question3 would be presented. This is defined with the following OCL expressions that are applied to relation1 as invariant conditions:

```
context question1 inv:
self.user.answered=true implies
    (self.TO1.visible=SHOW and
    self.TO2.visible=HIDE)

context relationship2 inv:
self.user.answered=false implies
   (self.TO1.visible=HIDE and
     self.TO2.visible=SHOW)
```

Note that the keyword **implies** means that when the expression is evaluated then if the condition left to the "implies" keyword is true then the condition to the right must also be true, in order for the whole expression to be true. In all the previous OCL examples the keyword **self** represents the UML classes that are the context in the specific OCL expression. It is evident that the OCL rules have been extracted-prescribed by the teaching model.

5. Discussion

The definition of a Design Model can facilitate the process of developing software projects, regardless of the domain of the application [9]. This facilitation is even more important in fields where the people involved in the development process come from different backgrounds so there is an increased need for a means of communicating design decisions. However, this improvement is often *confuted* by the lack of formalism in the definition of such models. This lack of formalism has certain negative aspects:

- Poorly defined models, which are based on the intuition of the designers rather than in predefined 'rules'.
- It is impossible to automate the authoring of models by means of specific Case Tools.
- It is impossible to automate the process of automatic code generation based on the models created (forward engineering).

Up to our knowledge only one similar attempt has been made for WAEA. In [7] a layer approach for the modeling of Adaptive Educational Applications is provided, together with a method for the design of such applications. This approach is similar to ours in the distinction of three views of Adaptive Educational Application depicted as layers: A conceptual Layer, a lesson layer and a student adaptation and presentation layers, which resemble our separation in three sub-models, i.e. conceptual, navigational and presentational. A second main similarity is that both approaches recognize that the authoring of WAEA is driven by an initial mapping of the available resources in a high level conceptual model. The main differences from this approach are in the way of mapping of the initially defined concepts into specific navigation and presentation elements, as well as the specific formalism used in our approach, namely UML.

The conceptual model we propose is based on or influenced by previous established models for web and hypermedia engineering. HDM [10] provides a model for high-level hypertext design. Like Dexter model [11] preserves the hierarchical structure of hypertext nodes but in addition supplies domain specific concepts, namely *Entities* and *Components,* so facilitating the definition of a conceptual *schema* of a hypertext application. The application itself is considered as an implementation of the previous schema and thus the model provides separation between conceptual design and implementation. In the same manner our model provides a conceptual organization of the educational through structuring of learning resources. The implementation of the previous organization in terms of navigation structure, user interface layout and actual content creation is taking place in later steps than the conceptual design, described in this paper.

OOHDM, described in [17] clearly proposes the separation between Conceptual, Navigational and User Interface Design steps in the development of hypermedia applications, suggesting certain types of models for each step. We follow this separation in the process of designing WbEA, though in the present paper we only deal with conceptual design.

Conallen's approach [5] introduces a UML extension for web application architecture modeling. While web-site modeling is more implementation oriented, introducing the web page as the core modeling element, a purely conceptual approach is adopted for the modeling of the business logic implemented in server side components. Unlike our model, it refers to generic web based applications and not particularly educational ones.

Hennicker and Koch [15] also extend UML for generic web application modeling. Unlike Conallen [5], they follow the distinction between conceptual, navigational and static presentation modeling.

Süß et al. [21] provide a UML extension by means of a meta-model for teachware management. Their meta-model can be separated into two sections, one concerning a conceptual model and one concerning a navigational model, implementing the previous one. They have also developed an XML based language, LM^2L that implements their meta-model. User Interface design as well as personalization of teachware is achieved by defining XSL stylesheets and applying them on LM^2L files.

EML [16] is not a strict educational hypermedia concept, but it defines a formal, XML based, language for modeling different aspects of the educational process e.g. activities, peoples' roles, content, etc, concerning specific lessons or courses called "units of study". It is a high level model aiming at facilitating instructional design in the context of e-learning.

This model has been tried out in small scale WAEA development projects. It still evolves but it seems that the notation used is valid and can be applied to every subject domain. The fact that this model uses a very specific and formal notation enables the development of a CASE tool that will support the design process. We do not expect the user to know OCL in order to describe the relationships and constraints applied. On the contrary, we are building a tool that will translate rules that the user will describe in “if then else” format (like the rules in the IMS Simple Sequencing [13]) into OCL that could be into understandable by an adaptive engine. Finally, we are in the process of modeling via reverse engineering a big WAEA in order to check the validity of this model with a “real” system.

References

1. P. De Bra and L. Calvi. AHA! An open Adaptive Hypermedia Architecture, *The New Review of Hypermedia and Multimedia*, vol. 4, pp 106-112 (2000).
2. P. De Bra, G. Houben and H. Wu. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *Proceedings of the tenth ACM Conference on Hypertext and Hypermedia,* Darmstadt, Germany, February 1999. ACM Press.
3. Peter Brusilovsky. Adaptive educational systems on the world-wide-web: A review of available technologies. In *Proceedings of Workshop WWW-Based Tutoring,* 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, 1998.
4. Brusilovsky. Methods and Techniqes of adaptive hypermedia. *User Modeling and User Adapted Interaction*, vol. 6, pp 87-129, 1996.
5. J. Conallen: Modeling Web Application Architectures with UML, *Communications of the ACM*, 42:10, Oct. 1999, pp. 63-70.
6. A. Cristea and L. Aroyo. Adaptive Authoring of Adaptive Educational Hypermedia. In *Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web System,* Malaga, Spain, May 2002.
7. M. C. Ferreira de Oliveira , M. A. Santos Turine and P. C. Masiero. A Statechart-Based Model for Hypermedia Applications, *ACM Transactions on Information Systems*, Vol. 19, Issue 1, 2001.
8. P. Fröhlich, N. Henze, and W. Nejdl. Meta Modeling for Hypermedia Design. In *Proceedings of IEEE Metadata Conference,* Maryland, USA, Sept. 1997.
9. D. Garlan and M. Shaw. Software Architecture: Perspectives of an Emerging Discipline. Prentice Hall, New Jersey, 1996.
10. F. Garzotto, D. Schwabe and P. Paolini. HDM- A Model Based Approach to Hypermedia Application Design. *ACM Transactions on Information Systems*, vol. 11, n. 1, pp. 1-26, 1993.

11. F. Halasz and M. Schwartz. The Dexter Reference Model. *Communications of the ACM*, vol. 37, n. 2, pp. 30-39, 1994.
12. T. Isakowitz, E. Stohr and P. Balasubramanian. RMM: A Methodology for Structured Hypermedia Design. *Communications of the ACM*, vol. 38, n. 4, 1995.
13. IMS Global Learning Consortium Inc. *IMS Simple Sequencing Information and Behavior Model*. Available at http://www.imsproject.org/simplesequencing/
14. N. Koch and A. Kraus. The Expressive Power of UML-based Web Engineering. In *Proceedings of the 2nd International Conference Workshop on Web Oriented Software Technologies,* Malaga, Spain, June 2002.
15. N. P. de Koch. Software Engineering for Adaptive Hypermedia Systems, PhD Thesis, Verlag Uni-Druck, Munich 2001.
16. Rod Koper. Modeling units of study from a Pedagogical Perspective: the pedagogical meta-model behind EML. Available at http://eml.ou.nl.
17. D. Lowe and W. Hall. *Hypermedia & the Web, an Engineering Approach*, John Wiley & Sons, 1999.
18. OMG. UML Profile for CORBA Specification. Available at http://www.omg.org/uml.
19. OMG. Unified Modeling Language Specification. Available at http://www.omg.org/uml. September 2001.
20. A. Papasalouros and S. Retalis. Ob-AHEM: A UML-enabled model for Adaptive Educational Hypermedia Applications. *Interactive educational Multimedia*, vol. 4, 2002.
21. C. Süß, B. Freitag and P. Brössler. Meta-Modeling for Web-Based Teachware Management. In *Advances in Conceptual Modeling*, ER'99 Workshop on the World-Wide Web and Conceptual Modeling , Paris, France 1999. Springer.
22. M. Uschold and M. Grunninger. Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review,* vol.1, n. 2, pp. 93-155, 1996.
23. J. Warmer and A. Kleppe. *The Object Constraint Language: Precise Modeling with UML.* Addison Wesley, Reading Mass., 1999.
24. Y. Zheng and M. Pong. Using statecharts to model hypertext. In *Proceedings of the ACM conference on Hypertext,* Milan, Italy, 1992, ACM Press.