Interactive Educational Multimedia, number 4 (April 2002), pp. 76-88
http://www.ub.es/multimedia/iem

Ob-AHEM: A UML-enabled model for Adaptive Educational Hypermedia Applications

Andreas Papasalouros

Department of Electrical and Computer Engineering
National Technical University of Athens - Greece

Symeon Retalis

Department of Computer Science
University of Cyprus - Cyprus

andpapas@softlab.ntua.gr

retal@ucy.ac.cy

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 Adaptive Hypermedia Applications Model (AHAM), and extends it by defining the presentation layer of a particular Hypermedia Application. 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 an case study application for the design of an adaptive web-based testing system.

Keywords

Adaptive Hypermedia, Educational systems, visual design modeling.

1. Introduction

Adaptivity is of paramount importance for educational applications on the WWW, as they are expected to be used by very different classes of users without any assistance by a real teacher. Educational applications delivered over traditional hypermedia systems suffer from certain problems that reduce the anticipated learning outcomes, such as:

- The 'lost in hyperspace' problem, where a user/learner looses his orientation while navigating into a complex hypertext structure.
- The lack of a teacher/mentor who would guide the user/learner during the learning process.
- The absence of concern about the individual characteristics of the users/learners, their previous knowledge about the subject they are studying, their history in navigating into the learning content, their learning style, their preferences, etc.

Adaptive Educational Hypermedia Systems (AEH) systems aspire to address these shortcomings and to provide individualized and personalized presentation of educational hypermedia content, easing the user's access to the content, facilitating the learning process and matching their own personal learning style. AEH systems are originated from the intersection of Adaptive Hypermedia Systems (AHS) and Educational Hypermedia Systems, a plethora of which has been developed recently; each one having its own adoption approach and implementation strategic [5]. As an attempt to capture the common elements, to provide a better understanding of their various aspects and to facilitate the interchange of applications created in the context of any of these systems, as well as any one that may be developed in the future, there is a need for a design model. In addition, such a model will be the basis for the development of a process for the development of adaptive educational hypermedia applications. Experience from traditional software engineering has shown that the adoption such a process in the quality of the product, namely adaptive educational hypermedia applications and the efficient management of the resources, time and effort is beneficial. This paper introduces an Object Oriented design model for AEH systems, which combines concepts from the Adaptive Hypermedia Application Model (AHAM) [8] and the Object Oriented Design Method (OOHDM) [12]. This design model adopts the Unified Modeling Language (UML) in order to formalize the syntax and semantics of the architecture of Adaptive Hypermedia Educational Applications.

UML is a visual modeling language, widely adopted in the software industry and pervasive in all kinds of modeling for software-intensive systems. The UML provides a consistent, human-readable way to specify a system's architecture, to capture requirements, to communicate design and implementation decisions between stakeholders in the software development industry. It proposes nine types of diagrams, each one containing certain types of model elements interconnected according to specific notation rules and semantics. It conforms to the object-oriented paradigm and is inherently component-based, at the implementation level. It is the de facto standard for visualizing, specifying, documenting and constructing the development products of software-intensive systems.

As far as the two aforementioned design models, we have chosen to extend them, as they are both offspring of the Dexter model. The latter is the first hypermedia design model ever and commonly serves as a reference model for "modern" hypermedia systems. Moreover, AHAM is an evolving design model specifically developed for AH systems [8] and OOHDM is a well known and widely used design method which allows the formulation of complex designs with a concise yet expressive notation thus simplifying the construction of AEH systems [12].

AHAM (like Dexter Model) deals with the storage layer, the presentation specifications and anchoring, as they are defined in the Dexter Model. In order to address the adoption issues of an AH system, AHAM alters the storage layer in the Dexter Model inserting into this layer three additional models: The Domain Model, the User Model and the Teaching Model. In addition, the model defines an Adaptive Engine that creates the user's perspective of the system, based on all the previous models.

OOHDM has also the design of a Conceptual Model that purports to capture precisely the important classes of objects (i.e., kinds of information) in the problem domain. Furthermore, the types of objects that the user will navigate (Navigational Classes) are defined as views over the Conceptual classes, which already provide a great amount of flexibility in accommodating variations.

The adoption of the principles of the object-oriented paradigm and the utilization of the UML for creating high-level models of AEHs and adaptive hypermedia applications in general, have certain benefits:

- The object-oriented paradigm has proven to be a successful software design and programming framework. Through its basic principles of encapsulation, inheritance and polymorphism, it provides a basis for well designed, easy to maintain, reusable software modules and systems. In addition, a number of the state of the art design and implementation environments (integrated development environments, application frameworks, case tools) conform to the object-oriented paradigm.
- Certain approaches like OOHDM have shown how object orientation can successfully address problems specific to hypermedia design. The use of object-oriented techniques in designing AEH applications seems even more appropriate since AEH are more sophisticated than conventional hypermedia applications. More specifically, an object-oriented approach to user modeling in AEH can give formal descriptions of user behavior, as regard to navigation and preferences, that can be reusable and able to be further refined and adjusted to specific technical and pedagogical contexts. Furthermore, object oriented design is best suited for systems undergoing complex state transitions over time.
- The UML is a standard, widely-adopted, formally-defined language for modeling software-intensive systems. Therefore, there exists strong motivation for using this language in the specification of models of AH applications.
- Models created with the UML are easy to understand, maintain and extend. A number of case tools are available facilitating the task of creation and maintenance of these models. In addition, these case tools can ensure the consistency of a model. For example, they can enforce an element in one model of an application to be properly associated with a corresponding element in a different model of the same application, according to specific semantics. Thus, adopting UML can benefit the design of AH applications in creating consistent and formal models.
- The UML can provide a common means of communication between AEH systems. In particular, through the OMG XMI standard, an XML-based standard for UML models interchange, models of AH applications can be shared between different AH systems.
- The adoption of a specific development process, combined with an appropriate CASE tool, can ease the development of AEH applications and result in high quality adaptive applications that meet their requirements and goals in terms of cost and time efficiency, pedagogical effectiveness, usability and technical competency. However such a process or tool is out of the scope of our current research.

The structure of the paper is as follows:

The analysis of the proposed model is given first. This model takes into account the following fundamentals of the AHAM: Domain Model, User Model, Teaching Model, Presentation Specifications, Run-time and Anchoring. Then, we associate the three aforementioned models of the AHAM to corresponding UML models. Each model contains a set of UML diagrams containing elements that correspond to AHAM fundamental elements.

The Teaching Model of the AHAM contains rules that cannot easily be specified using a visual notation like UML. For this purpose we propose the use of the Object Constraint Language (OCL), a formal language for expressing constraints related to UML specified models.

We also consider the Presentation Specifications in our model as the output of the Adaptive Engine, which is one of the innovative aspects of our model, which has not been addressed by other researchers, as yet. For this purpose we adopt the object-oriented concept of Abstract Data Views (ADV) [11], as used in the context of OOHDM. The modified ADVs not only provide dynamic, user-specific views of the Domain Model, but also specify the run time behavior of these views, corresponding directly to the run-time layer of the Dexter Model. The set of ADVs and ADV charts constitutes a new model itself.

Finally, we demonstrate the way in which the previous set of models constitutes the different views of an AEH application. The combination of these views defines a complete, consistent and expressive specification of the application.

The presentation of the design model will be accompanied by appropriate examples taken from a web-based testing system based on the IMS QTI standard (Instructional Management Systems Question and Test Interoperability, [http://imsproject.org] that has been developed in the Software Engineering laboratory of the NTUA.

2. Theoretical Underpinnings

2.1 Relation with other models

The proposed object-oriented model for the abstract representation of adaptive educational hypermedia applications is named Ob-AHEM (Object-oriented Adaptive Hypermedia Educational Model). Ob-AHEM is based on the Dexter Hypertext Reference Model, the Adaptive Hypermedia Application Model and the ADV Charts. In fact it is an object-oriented transformation of the AHAM with the introduction of the ADVCharts as a means of specifying the Presentation Specifications and the Run-time layer.

The Dexter Model suggests that any hypermedia system can be divided into three layers: The Storage Layer, that contains the actual data of a hypermedia application structured as nodes interconnected by links. This layer captures the main focus of the Model and is described in detail and formally specified. The Within-Component Layer, that represents the actual content and its structure inside the nodes of a hypertext, where the actual information is contained in the form of text, graphics, and multimedia. Due to its diversity in different existing hypermedia systems this layer is not further elaborated in the Dexter Model. Finally the Runtime Layer, that captures the dynamic aspects of hypermedia in the sense of instantiation of a component to the user interface environment and tracking of the sequence of the components presented to the user as the result of his/her navigation into the content (session management). Certain mechanisms provide the interfaces between the aforementioned layers: Anchoring permits the mapping of anchors, elements that belong to the within component layer, to components of the storage layer, intervening between the two layers. Presentation Specifications handle the presentation aspects of specific components in different contexts, acting as an intermediary between storage and runtime layer.

As mentioned before, the AHAM is a Dexter-based model that suggests three (sub)models for the definition of an Adaptive Hypermedia Application:

The Domain Model that provides a conceptual view of the AH application defining the concepts and their relationships that form the hypermedia content as they are considered by the author. The User Model that defines a user's personal characteristics as well as his/her history of interaction with the content that is conceptually described by the Domain Model. Thus, the User Model is closely tight with the previous model.

The Teaching Model that contains rules that transform the previous two models into hypermedia content presented to the user and control the evolution of the user model during the user's navigation into the hypermedia content.

All three models are located in the storage layer of the Dexter Model, as they ultimately deal with the nodes and links of the application. Through the use of the concept of Abstract Data Views we also consider the Presentation Specifications of the components defined in the Domain Model. In particular, defining the Presentation Specifications of concepts and relationships we achieve content and link adoption, respectively [4].

The Abstract Data View (ADV) Design Model is an object-oriented, based on state-chart diagrams, model for the design of User Interfaces for hypermedia applications, as used in the Object Oriented Hypermedia Design Method. The two fundamental elements in ADVs are the Abstract Data Objects (ADOs) and the Abstract Data Views themselves. ADVs are objects, in the sense that they have state, namely attributes that store specific values, and behavior, as they expose a specific interface responding to external events. ADOs are considered to only have state. Their dynamic behavior is controlled by one ore more ADVs that monitor the corresponding ADO. Two types of diagrams constitute an ADV model: Configuration diagrams and ADV Charts.

2.2 The UML as an extensible modeling language

The Ob-AHEM being an object-oriented model utilizes the UML, a standard visual modeling language that conforms to the object-oriented paradigm. Despite its expressiveness, UML cannot be successful in the definition of all types of models. In order for the UML to be effective in the definition of the model elements, the description of the notation and the capture of the semantics of models in fields different than typical software engineering, it provides certain, built-in extension mechanisms [2],[10]. These are stereotypes, tagged values and constraints. Stereotypes, the most important extension mechanism, are model elements that extend and refine existing UML elements, thus extending the vocabulary of the language. Stereotypes can be derived from existing model elements leading to more refined concepts. Tagged values are pairs of name-value strings that provide additional information about an element's specification. Constraints alter the semantics of existing UML blocks providing additional rules. Usually these rules are defined using the Object Constraint Language, a part of the standard UML specification, for applying constraints in UML models.

3. The model

3.1 The components of the Ob-AHEM

An Ob-AHEM model of an Adaptive Hypermedia Educational Application consists of the following components: Domain Model, User Model, Teaching Model, Presentation Specifications, Anchors and Run-time 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 1.

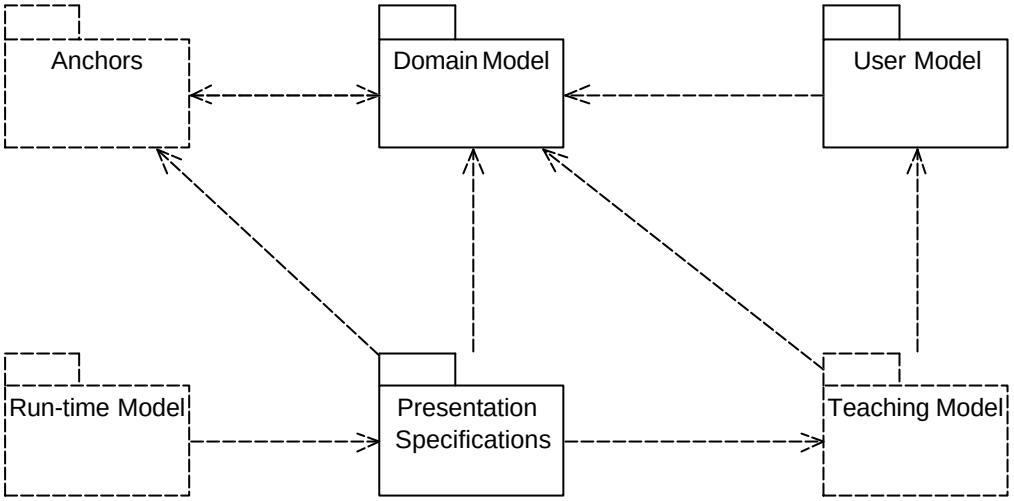

Figure 1. The sub-models that constitute Ob-AHEM and their dependencies

In Figure 1 the dashed arrows denote a '*depends on*' relationship in UML

In the following sub-sections we will describe the nature of each one of the previous models.

3.2 The Domain Model

The Domain model contains the actual nodes and links that compose the AH Application, both described by the term concept component. The model contains atomic concept components, that correspond to actual information fragments, composite concept components, that contain other concept components, and concept relationship components that establish the associations between concept components through the use of anchors and specifiers. Composite concept components can be abstract, if they contain only other composites, or pages, if they contain only atomic components. Each of the above concept components has a unique identifier and is described by a set of attribute-value pairs. It is noted here that the Domain Model is not explicitly a conceptual model or a lower-level navigation model. Its concepts and relationships can be either abstract, forming thus a semantic network, or a network of concrete pages and links. In both cases the adoption rules may be applied defining the adaptive hypermedia application. All the components previously described are defined as stereotyped classes in our model.

3.3 Anchors

As in the Dexter and AHAM models, a concept relationship contains one or more specifiers. A specifier of a specific anchor actually defines the direction (FROM, TO, BIDIRECT, NONE) of the anchor in the context of the specific relationship-link, as well as a presentation specification. Anchors are contained in the Anchors package, while specifiers are considered to belong to the Domain Model in the Ob-AHEM. Anchors and specifiers are described in Ob-AHEM as stereotyped classes.

3.4 User Model

The User Model consists of two types of elements, represented as stereotyped classes: The User Model Scheme and the User element. In the AHAM model only the User Model Instance element, corresponding to our User Model Scheme element exists. As will be shown later, each UserModelScheme element corresponds to a specific component of the Domain Model, defining the navigational state (read, ready to read, etc) of the component, as well as the level of acquired knowledge corresponding to the specific component from a conceptual point of view.

Thus, the set of UserModelSchemes constitutes the overlay model [4] of the user's knowledge about the studied subject, which is of temporal nature as it is rapidly changing as the user navigates into the adaptive hypermedia. Further, elements of User type in our model are used to represent the more stable, 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 [4] this constitutes the Stereotype user model. This is an extension to AHAM that our model suggests.

3.5 Teaching Model

The Teaching Model contains rules as Object Constraint Language [10] 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.

3.6 Presentation Specifications

In our model the presentation specifications are considered the output of the adaptive engine of the hypermedia system. The Presentation Specifications sub-model contains a set of elements that specify how a component of the Domain Model (including anchors) is projected to the user interface as a result of the user's navigation into the system. This specification is considered to be abstract, in that it does not take into account the layout, the color, particular fonts or images of a specified Domain Model component given that this visual presentation has no importance as the result of the adaptation itself.

We use the notion of Abstract Data Views, in particular Active Data Objects [11] for presentation specifications. Every component and anchor is associated to a particular ADO. ADOs and ADVs can be nested. In addition, composite components in the data model can contain other components, as well all types of components can contain anchors. This way a hierarchy of nested ADOs must correspond to a hierarchy of aggregated components and anchors. An ADO is an object that has state but not behavior. Its state is expressed by the values of its attributes, which show whether the corresponding component of anchor is visible or not, its color or visibility (as a result of navigation), whether its contents are sorted in a particular order in an index of links, etc. As will be shown later, in Ob-AHEM ADOs and ADVs are represented as stereotyped classes, each class having certain attributes. These attributes denote the visual state of the corresponding model element as the result of adaption. This state is by not means associated with the style, layout of navigation issues of the element, but defines the visual aspects of the element (and the components that it models) as a result of the adaptivity of

the system. For example, an attribute “color” in an ADV associated with an anchored text does not describe the color of the text defining its default style but its color defined by the Adaptive Engine as a result of the user’s navigation (exactly as the change of the color of a visited link in a document in the World Wide Web). An attribute “visible” is often used to denote whether a particular component can be presented or not in the current state of the adaptive system.

3.7 Runtime Model

The runtime model describes the runtime behavior of the hypermedia application through a set of ADVCharts. ADVCharts are statechart-like diagrams of interconnected ADVs. Each ADV must correspond to an Abstract Data Object of the Presentation Specifications. The transitions specified in the ADVCharts usually have as a result the application of a specific rule of the Teaching model. The rule is applied and the presentation specification (ADO) of a component is changed. It is important to point that the results of transitions at the runtime layer, as specified by the runtime model, is simple navigation, that is the hiding of specific ADVs and the appearance of one or more others. This kind of (simple) transition is explicitly separated in our model from the results of navigation from the adaptive perspective. In the second type of transitions (that usually happen simultaneously with the first) more elements of the model, as well as rules, participate than just Abstract Data Views.

3.8 The model elements

Table 1 presents the elements of our model, their mapping into UML elements and the sub-models where they are allocated. We have used the extension mechanisms of the UML, in particular the stereotypes, in order to extend the standard UML elements so as to match our model domain. The way these elements are associated according to the principles of the Ob-AHEM is shown in Figure 2. We could easily consider it as a meta-model from which we could derive instances, i.e. design models for each adaptive hypermedia educational system. The following section describes such a case study.

Table 1. Model elements

Model element	UML standard element	Stereotype name	Ob-AHEM component
Component	Abstract Class	<<component>>	Domain
Concept component	Class	<<concept>>	Domain
Abstract composite concept component	Abstract class	<<concept-composite>>	Domain
Page concept component	Class	<<page>>	Domain
Atomic concept component	Class	<<concept-atomic>>	Domain
Specifier	Class	<<specifier>>	Domain
Concept relationship	Class	<<relationship>>	Domain
Anchor	Class	<<anchor>>	Anchor
User Model Scheme	Class	<<userModelScheme>>	User
User	Class	<<user>>	User
Abstract Data Object	Class	<<ADObject>>	Presentation Specifications
Abstract Data View	Class	<ADView>>	Run-time

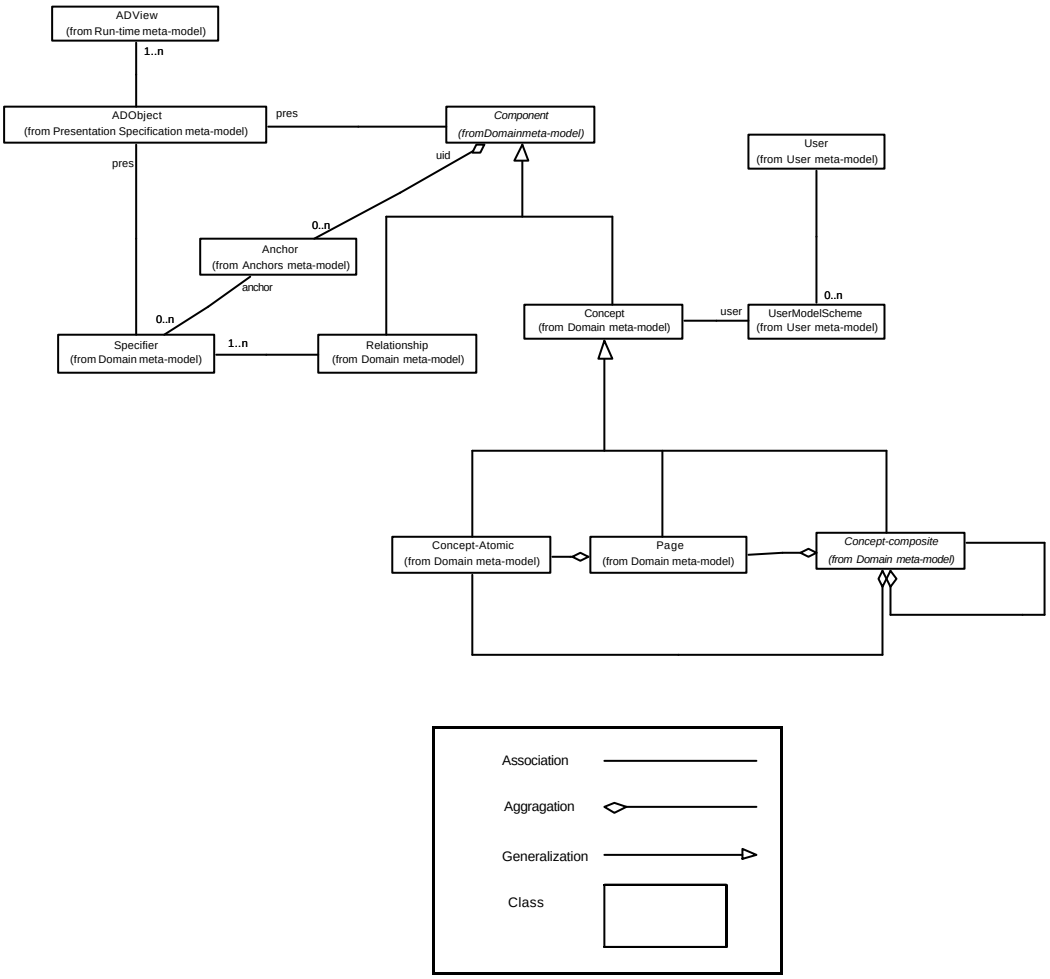

Figure 2. The Ob-AHEM meta-model

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

We exemplify the Ob-AHEM 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 [6], customizing the succession according to which the 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.

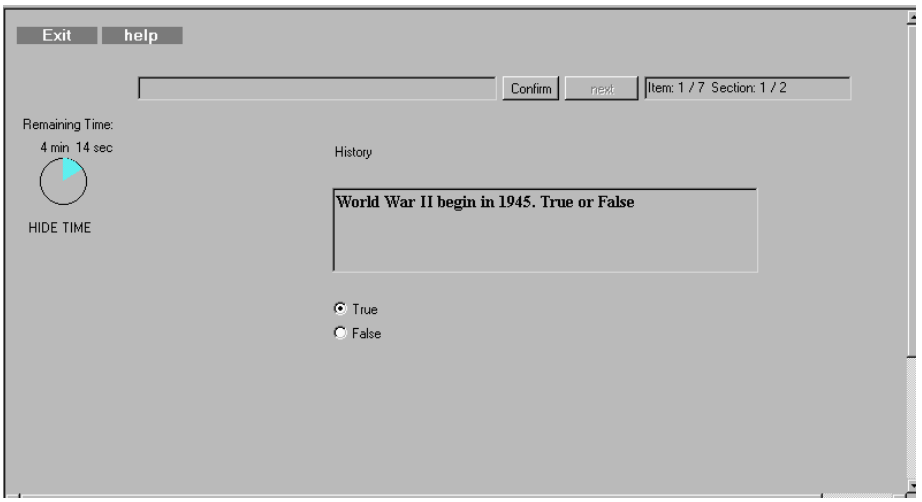

Figure 3. An example of the user interface

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.

As shown in Figure 4, the design model of this part of the system is presented according the the Ob-AHEM principles. In this 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. The role names of the specifiers related to a specific relationship are ss1, ss2, etc. In this example the presentation specifiers (ADO objects) are not shown in Figure 4.

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: if self.answer=self.correct_answer then
              self.user.answered=TRUE
        else self.user.answered=FALSE
```

This is a postcondition applied to the operation confirmed of the context component "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 accesed during the user's navigation. This kind of relationship between a domain model component and its corresponding user interface element is typically defined with a configuration diagram [11], as shown in Figure 5. 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.

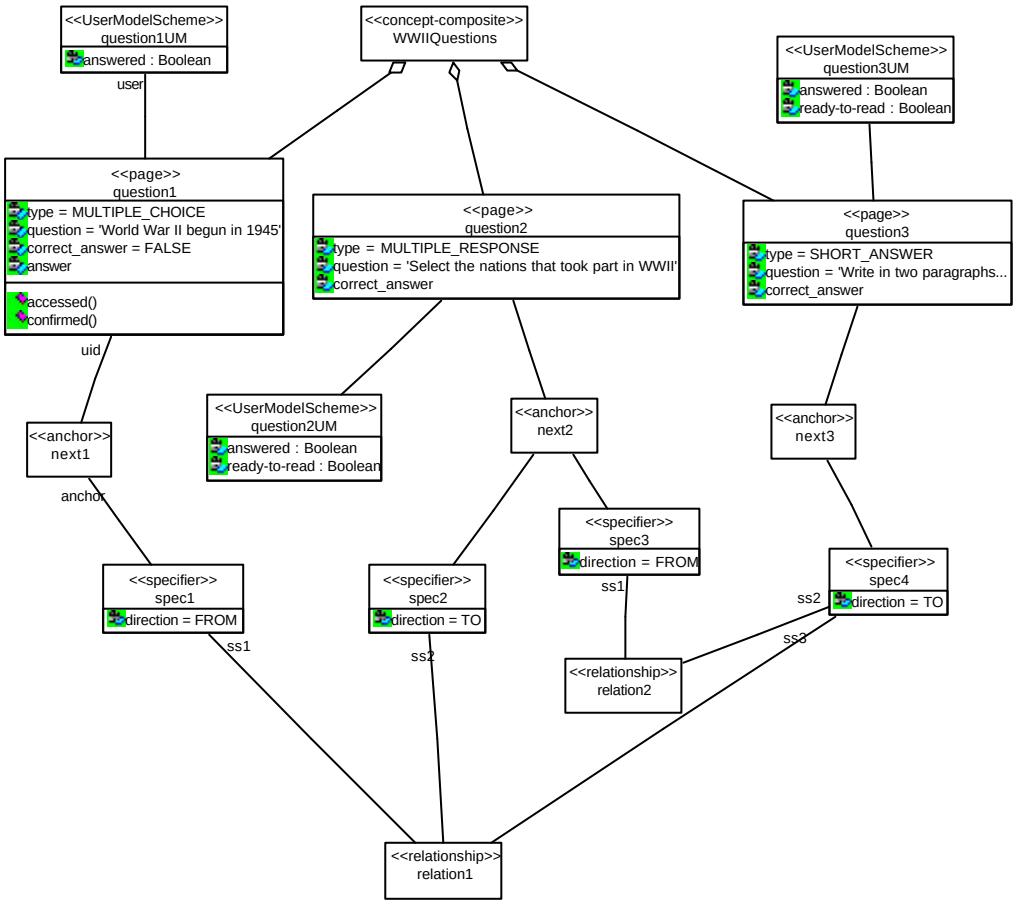

Figure 4. The adaptive question system example

The second constraint is applied to the relationship named "relation1". It connects three anchors having as corresponding specifiers: the "ss1" with direction FROM, the "ss2" and the "ss3" with direction TO. 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 relation1 inv:
self.ss1.anchor.uid.user.answered=true implies
    (self.ss2.uid.pres.visible=SHOW and
    self.ss3.uid.pres.visible=HIDE)

context relation1 inv:
self.ss1.anchor.uid.user.answered=false implies
   (self.ss2.uid.pres.visible=HIDE and
   self.ss3.uid.pres.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.

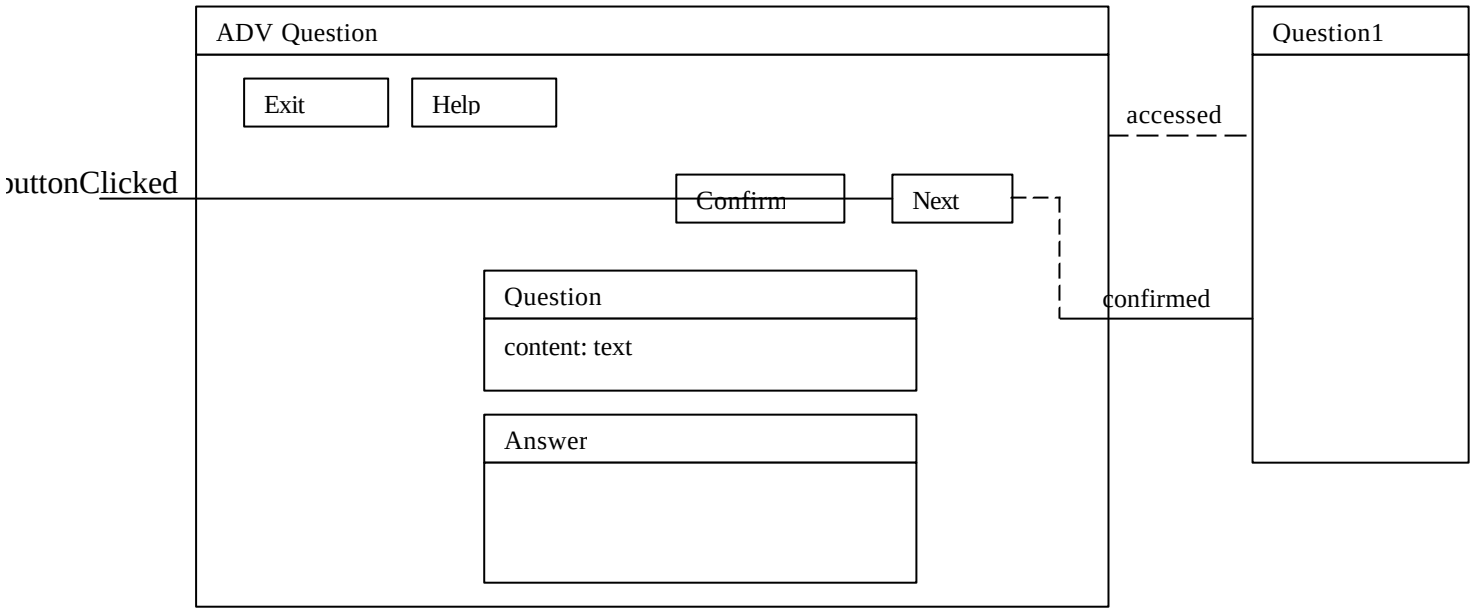

Figure 5. A configuration diagram

5. Conclusions and future work

In this paper we showed that the object-oriented paradigm and the UML can be used as the primitive tools to formally model Adaptive Hypermedia Educational Systems. We have taken AHAM, an existing model as a basis and mapped its primitive elements to graphic UML elements. We have extended the aforementioned model taking into consideration the presentation specifications of an adaptive hypermedia application using the notion of ADV Charts and have shown an example of how the model can be applied to an existing adaptive hypermedia application.

This model inherits many of the advantages of the object-oriented paradigm. It can be extensible, especially in the user modeling aspect of adaptive hypermedia. It promotes reusability of solutions in navigation, user modeling, definition and application of adaption rules at an abstract and technologically independent level. These solutions can be easily understood and communicated by means of an easy to comprehend and intuitive fashion that a graphical notation provides, providing a collection of design patterns [7]. In addition, the rules definition

with OCL, which is a language formally specified in [10] will facilitate automatic generation of (interchangeable) rules for systems that comply with the Ob-AHEM.
We are currently working on applying Ob-AHEM in order to introduce a new user model that is based on the level of achievement of learning objectives in the cognitive domain [1], in a per-concept basis with in an adaptive hypermedia educational system.

OB-AHEM is a high level model that can be used in order to produce more concrete models for specific applications. We intend to develop a more concrete design model for various web-based adaptive hypermedia applications, such as e-commerce. Finally, are in the progress of developing a system for the design and the creation of Adaptive Educational Hypermedia Applications with an Adaptive Engine based on the principles of the Ob-AHEM.

6. References

[1] Bloom, B.S (Ed). Taxonomy of Educational Objectives, Book 1: Cognitive Domain. New York: Longman, 1965.
[2] Grady Booch, James Rumbaugh, Ivar Jacobson. The Unified Modeling Language User Guide, Addison Wesley, 1999.
[3] Paul De Bra, Geert-Jan Houben, and Hongjing Wu. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. Proc. of ACM Hypertext '99, Darmstadt, Germany, 147-156, February 1999
[4] Peter Brusilovsky. Methods and techniques of adaptive hypermedia. User Modeling and User Adapted Interaction, Vol. 6, pp 87-129, 1996.
[5] Peter Brusilovsky. Adaptive educational systems on the world-wide-web: A review of available technologies. Proc. of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, 1998.
[6] Stephan Fischer. Course and Exercise Sequencing Using Metadata in Adaptive Hypermedia Learning Systems. ACM Journal of Educational Resources in Computing, Vol. 1, No. 1, Spring 2001.
[7] E. Gamma, R. Helm, R. Johnson and J. Vlissides. Design Patterns: Elements of reusable object-oriented software. Addison Wesley 1994.
[8] F. Halasz and M. Schwartz. The Dexter Reference Model. Communications of the ACM, Vol. 37, nr. 2, pp. 30-39, 1994.
[9] Maria Cristina Ferreira de Oliveira , Marcelo Augusto Santos Turine , Paulo Cesar Masiero. A Statechart-Based Model for Hypermedia Applications, ACM Transactions on Information Systems, Vol. 19, Issue 1 (January 2001).
[10] [OMG]. UML Version 1.4 Specification. Available at http://www.omg.org/technology/documents/formal/uml.htm
[11] G. Rossi, D. Schwabe, C.J.P. Lucena and D.D. Cowan. An Object-Oriented Model for Designing the Human-Computer Interface of Hypermedia Applications, Proceedings of the International Workshop on Hypermedia Design (IWHD'95), Montpellier, 5, Springer Verlag, 1995.
[12] D. Schwabe, G. Rossi. The Object-Oriented Hypermedia Design Model. Communications of the ACM, Vol. 38, nr 8, pp. 45-46, Aug. 1995.