

DESIGN PATTERNS IN ADAPTIVE WEB-BASED EDUCATIONAL SYSTEMS: AN OVERVIEW

Paris Avgeriou¹, Dimitrios Vogiatzis¹, Aimilia Tzanavari¹, Symeon Retalis²

¹University of Cyprus

Department of Computer Science
75 Kallipoleos Str., P.O. Box 20537,
1678 Nicosia, Cyprus

E-mail: {pavger, dimitrv, aimilia}@cs.ucy.ac.cy

²University of Piraeus,

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

Abstract

Adaptive Web-based Educational Systems represent an emerging technology that provides a unique advantage over traditional Web-based Educational Systems; that is the ability to adapt to the user's needs, goals, preferences etc. Adaptive Web-based Educational Systems are increasingly becoming part of the mainstream education, yet there does not exist a disciplined way of designing them - most of the development is *ad-hoc*. This paper aims to fill this void, which is the absence of disciplined design, by recording the expertise of existing Adaptive Web-based Educational Systems in the form of design patterns. We present a categorization of the patterns according to an established paradigm in Adaptive Hypermedia, we examine the patterns in one of these categories and provide an exemplary pattern.

Keywords

Design Patterns, Pattern Language, Adaptive Hypermedia Systems, User Modeling, e-Learning.

1. Introduction

An Adaptive Web-based Educational System (AWES) [11] is a dynamic web-based application, which provides a tailored learning environment to its users, by adapting both the presentation and the navigation through the content. Such a system is comprised of learning resources, as well as a set of tools that facilitate the process of studying, such as exams/questionnaires, glossaries, communication tools, etc. The learning content is dynamically generated based on some pedagogical rules that combine the (domain) model of the content with the model of the user. AWES are currently a 'hot' topic of research in the broader field of adaptive hypermedia applications [10],[14] and several AWES systems have been built during the past years [7],[8],[9],[12],[13],[15],[16],[28],[36],[37].

Several educational institutions are nowadays designing and developing their own AWES due to the fact that AWES leverage the shortcomings of non-adaptive web-

based educational systems, also known as Learning Management Systems or Virtual Learning Environments. In contrast to the latter, AWES can provide tailored, one-to-one tutoring, according to the specific characteristics of each individual learner rather than serve the same content massively to all the learners. Furthermore, some educational or research institutions tend to develop their own AWES in order to implement and test their own learning theories or instructional design methods.

However, the design and implementation of Adaptive Web-based Educational Systems (AWES) is a complex, if not overwhelming, task. This is due to the fact that it involves people from diverse backgrounds, such as software developers, web application experts, content developers, domain experts, instructional designers, user modeling experts and pedagogues, to name just a few. Moreover, these systems have presentational, behavioral, pedagogical and architectural aspects that need to be taken into account. To make matters worse, most AWES are designed and developed from scratch, without taking advantage of the experience from previously developed Adaptive Web-based Educational Systems, because the latter's design is not codified or documented. As a result, development teams are forced to 're-invent the wheel'.

Therefore, systematic and disciplined approaches must be devised in order to overcome the complexity and assortment of AWES and achieve overall product quality within specific time and budget limits. One such approach is the use of **design patterns** [2], so that these systems are not designed and implemented from scratch, but based on reusable design experience gained over several years of trial-and-error attempts. Therefore good design can be made explicit, and available to the whole community of designers, so that it becomes common practice. In this way, designers of new or existing AWES, especially inexperienced designers, can take advantage of previous design expertise and save precious time and resources.

The idea of design patterns began in the field of building architecture, when Christopher Alexander invented the idea of capturing design guidelines in the form of design patterns [2]. According to him, "each pattern describes a problem, which occurs over and over again in our environment, and then describes the core of the solution to that problem in such a way that you can use this solution a million times over". Patterns are not conceived in a 'big

bang’ but rather discovered or mined after numerous implementations of the same solution in a given problem, usually by different people. Alexander also proposed the notion of a **pattern language**, which is a collection of related patterns that captures the whole of the design process and can guide the designer through step-by-step design guidelines.

The ‘Alexandrian’ patterns found many followers in the computer science discipline, especially after the so-called ‘GOF’ book for object-oriented design [22]. Some of the fields that have adopted patterns are: software architecture [17],[34], hypermedia engineering [33], object-oriented analysis [21], business modeling [20], etc. Design patterns have also been developed for the field of Human Computer Interaction [26], including patterns for Interaction Design [6], [19], User Interfaces [38], Web Applications [40], Cooperative Interaction [29], Usability [24] and for teaching Human-Centered Design [35]. There is also a repository of patterns in the conventional learning and pedagogical discipline, mainly focused on teacher-based learning [31]. Recently the e-learning community has established a design pattern repository [3],[18], and it is this domain where the added value of this paper aims at. A relevant initiative that attempts to define reusable design techniques in user-modeling based adaptivity and adaptability can be found at [1].

This paper aims to initiate a pattern language for the domain of AWES, based on a well-established reference model for adaptive hypermedia, the AHAM model [10]. In particular, it uses the various layers, components and interfaces of AHAM to categorize and organize the patterns. It also portrays an overview of the entire pattern language at three levels of abstraction:

- The seven categories of patterns according to AHAM
- The organization of the patterns in a specific category
- A sample exemplary pattern of a specific category

The structure of the rest of this paper is as follows: the second section categorizes the pattern in the domain of AWES into seven thematic groups by utilizing the AHAM reference model. The third section presents an overview of the patterns in one category, namely the user model patterns, while the fourth section portrays an exemplary pattern. Finally, the fifth section wraps up with conclusions and ideas for future work.

2. Design Patterns for AWES

The most widely accepted reference model for adaptive hypermedia applications is AHAM [10], a model based on the well-established Dexter reference model [25] for hypermedia applications. AHAM extends the Dexter model and proposes that an adaptive hypermedia application is comprised of the following parts:

- **Within component layer**, which deals with the content and structure inside the hypermedia nodes.
- **Storage layer**, which is comprised of three parts:

- The **user model** which defines the knowledge acquired by the learner and the links she has visited.
- The **domain model**, which is a “database” with all the nodes and links of the application.
- The **teaching model**, which is a set of pedagogical rules that combines the user model and the domain model to perform the actual adaptation.

- **Run-time layer**, which deals with the presentation of the hypermedia to the user and takes care of the dynamic aspects of the system.

AHAM also defines two interfaces between the above layers:

- An **anchoring interface** that connects the storage layer and in specific the domain model, with the within component layer
- A **presentation specification interface** between the storage layer and in specific the domain model with the run-time layer.

Figure 1 depicts the AHAM reference model, using the object-oriented notation of the Unified Modeling Language [5],[32].

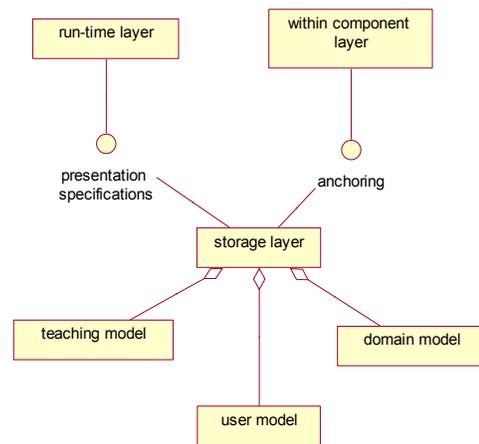


Figure 1. The AHAM Reference Model

According to the software engineering discipline [4] a reference model is a division of a system’s functionality into parts, together with the flow of information between the pieces. The AHAM reference model accomplishes exactly that:

- It separates the functionality of an adaptive hypermedia application into three parts, namely the three layers, storage, within component and run-time.
- It defines the flow of information between the above parts through the two interfaces proposed by the model, anchoring and presentation specifications.

The AHAM reference model is adopted in this paper because it provides a clear and precise way to categorize the patterns in the domain of adaptive web-based

educational applications. These categories define different thematic groups that solve similar problems, and they assist in managing the patterns, especially when their number increases to a large extent. Therefore, for the domain of AWES, we define seven categories of patterns:

- **Run-time layer patterns** that deal with the User Interface and dynamic aspects of AWES.
- **User model patterns** that concern the ways of creating and manipulating user models.
- **Domain model patterns** that concern the ways that the content is structured into nodes connected by links.
- **Teaching model patterns** that deal with the pedagogical rules that combine the user model and the domain model to perform the adaptation of the content.
- **Within component layer patterns** that concern the structuring of the content and the content per se within the nodes.
- **Anchoring patterns** that deal with the ways to link domain model elements with the within-component layer elements.
- **Presentation specification patterns** that concern the presentation of the domain model elements in the run-time layer.

The whole pattern language that is comprised of the aforementioned categories is not possible to be presented here due to its size. Instead the next section will provide an overview of the user model patterns, by briefly outlining what the patterns are and what problems they solve.

3. The map of the user model patterns

This section presents an overview of the design patterns that belong to the user model patterns category, as described in the previous section. The organization of these patterns can be achieved according to their interdependencies [22]. In the most fundamental repositories of patterns such as [17],[22], graphs or maps are designed that show how the distinct patterns refer to each other and what the nature of their relationship is. Figure 2 depicts the relationships between the proposed AWES user model design patterns. The semantics of the arrows between the patterns is that the pattern at the beginning of the arrows defines a general solution that in sequence entails more specific problems that are resolved by the patterns at the end of the arrows.

The user model component of an AWES is responsible for describing, representing, acquiring and maintaining the user model [39]. It is of paramount importance to define a complete and accurate user model so that the system can better adapt to the user's individual needs and characteristics. This is subsequently used in the adaptation phase, where primarily navigation and presentation are tailored to the user's needs. The patterns presented here attempt to cover the entire user modeling process. They originate from studying [39] a significant number of existing AWES, along with codifying expertise in user modeling. It is noted that the pattern names are in uppercase letters, so as to distinguish them inside the text. The USER MODEL COMPONENT is the umbrella pattern addressing the problem of how we design the user model component of an Adaptive Web-based Educational System (AWES) from scratch. It entails the following patterns:

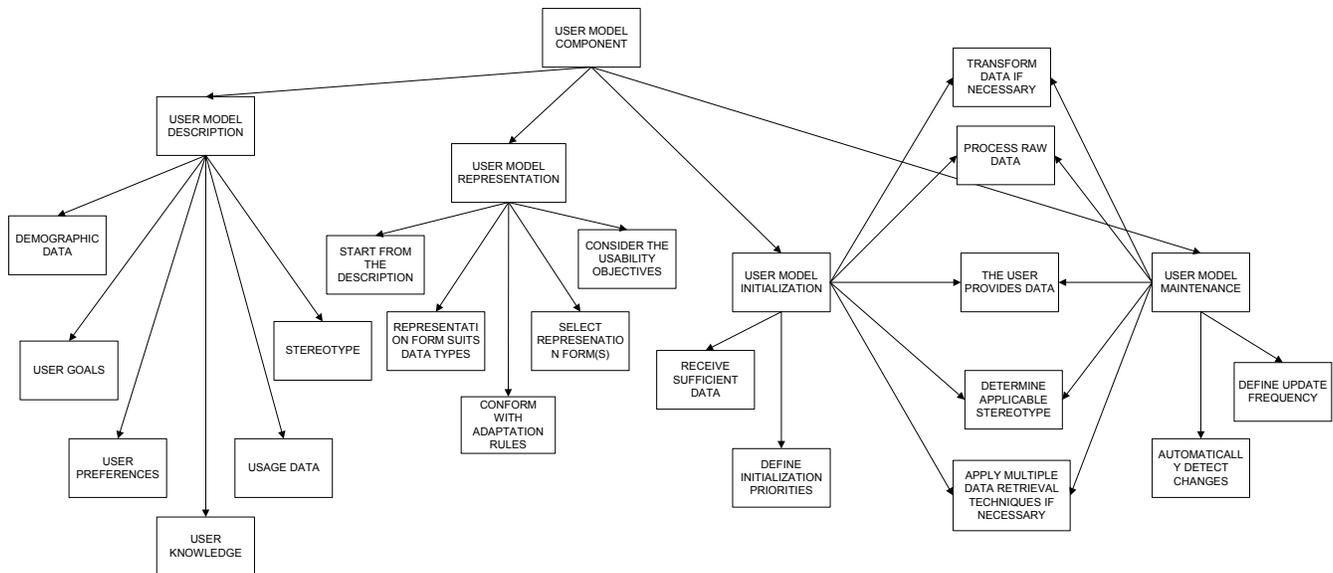


Figure 2. The patterns for user modeling

1. USER MODEL DESCRIPTION, shows what information should a user model that is to be used in an Adaptive Web-based Educational System (AWES) include, namely:
 - i. DEMOGRAPHIC DATA, which refers to the demographic information that has to be kept about the user.
 - ii. USER GOALS, which refers to the user's educational goals when using the AWES.
 - iii. USER PREFERENCES, which describes the user's preferences with respect to the various dimensions of the learning opportunity e.g. learning style or mode of assessment.
 - iv. USER KNOWLEDGE, which describes what should be considered as user knowledge and where this information may be found.
 - v. USAGE DATA, which portrays how usage data, can be exploited to learn things about the user.
 - vi. STEREOTYPE, which explains how stereotypes can be employed in a user model.
2. USER MODEL REPRESENTATION solves the problem of representing the user model. It is supported by the following patterns:
 - i. START FROM THE DESCRIPTION, which summarizes that the user model's description individual elements are the ones to be represented.
 - ii. REPRESENTATION FORM SUITS DATA TYPES, which explains why the data's characteristics can play an important role in the representation.
 - iii. CONFORM WITH ADAPTATION RULES, which gives evidence why the form of representation also depends on the adaptation rules format.
 - iv. CONSIDER THE USABILITY OBJECTIVES, which describes the reason why some of the AWES usability objectives may affect the decision regarding the representation form.
 - v. SELECT REPRESENTATION FORM(S), which demonstrates that one form of representation, may not be suitable to represent all the elements of the user model's description, but several forms may be selected.
3. USER MODEL INITIALIZATION addresses the problem of acquiring the user model initially. It is supported by the following patterns:
 - i. DEFINE DESIRED SUBSET OF ELEMENTS, which describes the subset of the user model elements that the designer considers sufficient to form the initial user model.
 - ii. DEFINE INITIALIZATION PRIORITIES, which analyzes the fact that some elements of the user model description, need to be initialized before some others.
 - iii. THE USER PROVIDES DATA, which describes the process of the user providing some elements of information directly.
 - iv. PROCESS RAW DATA, which describes the process of the system deriving some elements of information based on the user's interaction with the AWES.
 - v. TRANSFORM DATA IF NECESSARY, which explains the probability that the data derived after processing the raw data are not in the required representation form and as a result need to be transformed.
 - vi. APPLY MULTIPLE DATA RETRIEVAL TECHNIQUES IF NECESSARY, which gives evidence that different elements of the user model description, may not be suitable to be acquired with the same data retrieval techniques.
 - vii. DETERMINE APPLICABLE STEREOTYPE, which describes the process of determining the stereotype that applies to the particular user model description instance.
4. USER MODEL MAINTENANCE addresses the problem of maintaining an accurate user model.
 - i. AUTOMATICALLY DETECT CHANGES, which describes the process of the system monitoring the users' interaction to determine changes in their user model.
 - ii. DEFINE UPDATE FREQUENCY, which explains that the task of updating the user model should take place with a certain frequency that depends on a number of factors.
 - iii. THE USER PROVIDES DATA (same as above)
 - iv. PROCESS RAW DATA (same as above)
 - v. TRANSFORM DATA IF NECESSARY (same as above)
 - vi. APPLY MULTIPLE DATA RETRIEVAL TECHNIQUES IF NECESSARY (same as above).
 - vii. DETERMINE APPLICABLE STEREOTYPE (same as above)

4. A sample pattern

A proposed pattern gets accepted by the corresponding pattern community only if there have been two or three examples of its use by someone other than the one who suggested the pattern [17]. Therefore we have searched for the implementation of the proposed patterns in existing AWES, such as AHA! [7], I-Help [16], ISIS-Tutor [12], ELM-ART II [13], BGP-MS (with KN-AHS) [28], Interbook [15], AST [37] and ALE [36].

The next step was to describe them in a suitable format in a similar way to patterns of other domains. As eloquently stated in [23], it is more difficult to describe patterns than to actually find them. Almost all of the approaches that have proposed patterns in a subject field have also suggested a novel way of describing and cataloging them. Our suggestion for a pattern description format is largely based on the format proposed in [30] and contains the following fields:

1. **Name** – a unique name to distinguish the pattern and uniquely refer to it.
2. **Problem** – a brief description of the design problem at hand.
3. **Context** – the situation in which the problem is solved by the solution.

4. **Forces** – the often contradictory factors that need to be accounted for when choosing a solution to the problem.
5. **Solution** – a description of the solution proposed by this pattern that addresses the problem and motivation stated earlier.
6. **Related Patterns** – other patterns that are related to this one in some way, e.g. they are involved in the solution.
7. **Known uses** – examples of real AWES where this pattern has been applied.

A significant number of patterns of this language have been presented in [39]. For illustrative purposes, in this paper we include the USER GOALS pattern, which refers to the user's educational goals when using the AWES.

USER GOALS

Problem: What information should be included as user goals in a user model that is to be used in an Adaptive Web-based Educational System (AWES)?

Context: You have already designed the USER MODEL COMPONENT. You have also designed the USER MODEL DESCRIPTION and have decided to include user goals in the user model. You are currently reflecting about what to include as user goals.

Forces:

- Being able to model the user/learner's educational goal(s) can facilitate adaptation. An AWES (via its author) can "deliver" the same course differently to learners with different educational goals, by setting the appropriate conditions to meet those goals. For example a learner with a goal "to master subject X" will receive more in-depth tutoring than a learner with a goal "to familiarize themselves with subject X".
- It is false to assume that all users/learners aim at learning all of the material offered by an AWES.
- Educational goals can vary in scope. For example they may refer to the whole duration of the course, or to only a part of it.
- There are some types of user goals that can be determined by the users, but some others cannot. For example, the users can determine the initial educational goal before starting a course using the AWES, but probably cannot be in a position to set the best (short-term) goal(s) for themselves while the course is in progress.
- The user goals determined may be in a form that is not suitable to be directly included in a user model.

Solution:

Include specific user goals in the user model in order to facilitate adaptation, and capture the real intent of the learner with respect to the learning material. The information that has to be kept as user goals in order for the system to better adapt to its user, is divided in two categories:

- Long-term goals - educational goals that are valid for a longer period of time and require significant effort

to be met.

- Short-term goals – educational goals that are valid for a shorter period of time and require relatively moderate effort to be met.

Long-term goals are usually determined by the users, whereas short-term goals by the AWES that plays the role of a tutor and is driven by the course author. In the second case, a goal modeling component may be required: a component that will take relevant data (frequently USER KNOWLEDGE), process it and derive goals.

In order for particular user goals to be included in a user's model, a pre-processing operation may be necessary to bring them to the required format that was defined when the USER MODEL REPRESENTATION was designed.

Related patterns: USER MODEL DESCRIPTION, USER MODEL REPRESENTATION, USER KNOWLEDGE.

Known uses:

Interbook [15] initially modeled an educational goal as a sequence of sets of concepts, while later on as a stack of sets, allowing the user to move a selected goal to the top of the stack. BGP-MS [28] models the user's goals in multiple ways. Firstly, the developer of BGP-MS applications can specify groups of users that share common goals. In addition, he is able to specify the user goals that correspond to specific answers to a questionnaire, as well as to specify the user goals that correspond to specific user actions as they are observed by the system.

5. Conclusions and future work

We have introduced a pattern language for the design of Adaptive Web-based Educational Applications. The patterns have been categorized according to the AHAM paradigm, and one of these categories, which concerns the USER MODEL COMPONENT has been presented. Finally we demonstrated one of the patterns of the aforementioned category. It is noted that the hierarchical, tree-like categorization of patterns presented in the third section can be continued by decomposing the tree further and defining even lower-level patterns. For example, the USER GOALS pattern can be 'broken down' into smaller patterns that define types of user goals.

Our purpose was the introduction of a disciplined approach for designing AWES, which will be beneficial to inexperienced designers of AWES. The proposed pattern language structured as a hierarchy will help an AWES designer to have a broad view of the system, to know the components that should be incorporated, as well as their interdependencies. However, it should be understood that the pattern language is not a blue print design thus customization should be undertaken.

In the future, we aim to improve the pattern language by exposing it to the pattern community, as well as to incorporate it in a newly established relevant repository [18]. The final goal is to develop the rest of the patterns that are stated in the pattern language and have not been written yet.

References

- [1] ADAPT project web site <http://www.wis.win.tue.nl/~acristea/HTML/Minerva/index.html>
- [2] Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. and Angel, S. *A Pattern Language* (Oxford University Press, 1977).
- [3] Avgeriou, P., Papasalouros, A., Retalis, S. and Skordalakis, E., Towards a Pattern Language for Learning Management Systems, *IEEE Educational Technology and Society Journal*, 6(2), 2003, 11-24.
- [4] Bass, L., Clements, P. and Kazman, R. *Software Architecture in Practice* (Addison-Wesley, 1998).
- [5] Booch, G., Rumbaugh, J. and Jacobson, I. *The UML User Guide* (Addison-Wesley, 1999).
- [6] Borchers, J. *A Pattern Approach to Interaction Design* (John Wiley, 2001).
- [7] De Bra, P., Aerts, A., Smits, D. and Stash, N. AHA! Version 2.0: More Adaptive Flexibility for Authors. *Proc. AACE ELearn 2002*.
- [8] De Bra, P. and Calvi, L. Creating adaptive hyperdocuments for and on the Web. *Proc. WebNet'1997*, 149-155.
- [9] De Bra, P. and Calvi, L. Towards a Generic Adaptive Hypermedia System. *Proc. Workshop on Adaptive Hypertext and Hypermedia 1998*, 5-11.
- [10] De Bra, P., Houben, G. and Wu, H. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. *Proc. ACM Conference on Hypertext and Hypermedia 99*.
- [11] Brusilovsky, P. Adaptive and Intelligent Technologies for Web-based Education. *KI – Kunstliche Intelligenz*, 13(4), 1999, 19-25.
- [12] Brusilovsky, P. and Pesin, L. ISIS-Tutor: An adaptive hypertext learning environment. *Proc. JCKBSE 1994*, 83-87.
- [13] Brusilovsky, P., Schwarz, E. and Weber, G. ELM-ART: An intelligent tutoring system on World Wide Web. *Proc. ITS 1996, LNCS 1086*, 261-269.
- [14] Brusilovsky, P. Adaptive educational systems on the world-wide-web: A review of available technologies. *Proc. of Workshop WWW-Based Tutoring - ITS 1998*.
- [15] Brusilovsky, P., Eklund, J. and Schwarz, E. (1998). Web-based education for all: A tool for developing adaptive courseware. *Computer Networks and ISDN Systems*, 30(1-7), 1998, 291-300.
- [16] Bull, S., Greer, J., McCalla, G., Kettel, L. and Bowes, J. User Modelling in I-Help: What, Why, When and How. *Proc. UM 2001. LNAI 2109*, 117-126.
- [17] Buschmann F., Meunier, R., Rohnert, H., Sommerland P. and Stal, M. *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns* (John Wiley & Sons, 1996).
- [18] E-LEN project. <http://www.tisip.no/E-LEN/>
- [19] Erickson, T. Towards a Pattern Language for Interaction Design. In Luff, P., Hindmarsh, J. and Heath, C. (eds) *Workplace Studies: Recovering Work Practice and Informing Systems Design* (Cambridge University Press, 2000).
- [20] Eriksson, H. and Penker, M. *Business Modelling with UML-Business Patterns at work* (John Wiley & Sons, 2000).
- [21] Fowler, M. *Analysis Patterns: Reusable Object Models* (Addison-Wesley, 1996).
- [22] Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software* (Addison-Wesley, 1994).
- [23] Garrido, A., Rossi, G. and Schwabe, D. Pattern Systems for Hypermedia. *Proc. PLOP 1997*.
- [24] Graham, I. *A Pattern Language for Web Usability* (Pearson Education, 2002).
- [25] F. Halasz and M. Schwartz. The Dexter Reference Model. *Communications of the ACM*, 37(2), 1994, 30-39.
- [26] HCI design patterns web site <http://www.hcipatterns.org/>
- [27] IMS Learning Information Package Specification <http://www.imsproject.org/>
- [28] Kobsa, A., Müller, D. and Nill, A. KN-AHS: An adaptive hypertext client of the user modeling system BGP-MS. *Proc. UM 1994*, 31-36
- [29] Martin, D., Rouncefield, M. and Sommerville, I. Applying patterns of cooperative interaction to work (re)design: e-government and planning. *Proc. CHI 2002*, 235-242.
- [30] Mezaros, G. and Doble, J. A Pattern Language for Pattern Writing. *Proc. PLOP 1998*, 529-574.
- [31] Pedagogical Patterns Project site, <http://www.pedagogicalpatterns.org/>.
- [32] Rumbaugh, J., Jacobson, I. and Booch, G. *The UML Reference Manual* (Addison-Wesley, 1999).
- [33] Rossi, G., Garrido, A. and Carvalho, S. Design Patterns for Object-Oriented Hypermedia Applications. In Vlissides, J., Coplien, J., Kerth, N. (eds.) *Pattern Languages of Program Design*, 2 (Addison-Wesley, 1996) 177-191.
- [34] Schmidt, D., Stal, M., Rohnert, H. and Buschmann, F. *Pattern-Oriented Software Architecture, Volume 2: Patterns for Concurrent and Networked Objects* (John Wiley & Sons, 2000).
- [35] Seffah, A. Learning the ropes: human-centered design skills and patterns for software engineers' education. *Interactions*, 10(5), 2003, 36-45.
- [36] Specht, M., Kravcik, M., Klemke, R., Pesin, L. and Hüttenhain, R. Adaptive Learning Environment for Teaching and Learning in WINDS. *Proc. 2002 Conference on Adaptive Hypermedia and Adaptive Web Based Systems*.
- [37] Specht, M., Weber, G., Heitmeyer, S. and Schöch, V. AST: Adaptive WWW-Courseware for Statistics. *Proc UM 1997*.
- [38] Tidwell, J. Common Ground: A Pattern Language for Human-Computer Interface Design. http://www.mit.edu/~jtidwell/common_ground.html
- [39] Tzanavari, A., Avgeriou, P. and Vogiatzis, D. Patterns for Adaptive Web-based Educational Systems, *Technical Report*, Department of Computer Science, University of Cyprus, December 2003.
- [40] Weiss, M. Patterns for Web Applications. *Proc. PLOP 2003*.