ADAPTIVE HYPERMEDIA SYSTEM DESIGN: A METHOD FROM PRACTICE

Jacopo ARMANI
Institute of Communication Technology
Faculty of Communication Sciences
Università della Svizzera italiana (USI)
Jacopo.Armani@lu.unisi.ch

Luca BOTTURI
Institute of Communication and Education
Faculty of Communication Sciences
Università della Svizzera italiana (USI)
Luca.Botturi@lu.unisi.ch

ABSTRACT

Adaptive Hypermedia Systems represent a great potential for e-learning; nevertheless instructors and designers find it difficult to develop adaptive application within their framework. This paper presents a map-based visual design method, developed at the AHS atelier in the University of Lugano and tailored for non-technical people, for designing adaptive courseware according to different instructional strategies.

KEYWORDS

Educational Adaptive Hypermedia Systems, design, maps, interfaces, instructional strategy.

1. INTRODUCTION

Along the last years, Adaptive Hypermedia Systems (AHS) have been presented as a promising innovation first in CAI, then CBT and finally e-learning. Their potentiality of personalization and adaptation to the learner's cognitive style make them indeed a powerful tool. Nevertheless, while conceptual models have been refined and technical implementations improved, one deciding point has hindered the widespread application of AHS: design.

The study presented in this paper moves from the consideration that potential instructors and instructional designers (generally, *adopters*) willing to exploit any AHS simply do not know *how* to do it. On the one side, AHS still require high *technical skills*, and this imposes high production and testing costs. On the other side, each AHS requires the use of a specific conceptual model: each model is unique, and works in its own way. Moreover, no *design method* comes with the system, so that potential adopters do not know how to design their application. Despite these issues have been recognized in the last years and current research efforts are focused on developing authoring tools that enable adopters to use adaptive systems (Wu *et al.* 2000), very few works (Cristea & Aroyo, 2002, Carro *et al*., 2002) are aimed specifically at defining methodologies that enable adopters to effectively exploit AHS. Our contribution is focused on the development of a practical methodology for designing adaptive courseware.

2. THE ADAPTIVE HYPERMEDIA SYSTEM ATELIER

This issue was considered in the Adaptive Hypermedia Systems Atelier, held at the University of Lugano in 2002/2003. The Atelier involved 18 students from the School of Communication Science who worked under the authors' guidance at the development of online adaptive self-learning modules. The atelier's goal was to

Originally published as Armani, J. & Botturi, L. (2003). Adaptive Hypermedia System Design: a Method from Practice. WWW/Internet 2003, Algarve, Portugal.

develop a methodology for adaptive courseware design. Students were divided in four groups and the following conditions were set:

1. *Dimension:* the final outcome for each group was an online adaptive module for about a 4 hours' self-learning activity.
2. *Target:* the intended targets for the instruction developed in the atelier were university students and young managers.
3. *Subject matter:* the topics of the instruction were *effective email use* (chosen by 3 groups) and *Rhetorics* (chosen by 1 group).
4. *Technical environment:* the adaptive modules were to be exploited online, and the adaptive engine selected was AHA! version 1.0. We chose AHA! among other general purpose adaptive systems because of its reasonable ease of configuration and because of its availability for both UNIX and Microsoft platforms.
5. *Reality check:* groups were asked to develop real courseware for the selected topics, so not only a demo, but actual courseware potentially ready for real instruction. The only exception made was for time-consuming multimedia development (i.e. audio files, animations, videos).

3. WORKING HYPOTHESES

As pointed out by (Carro, 2002) "It can be desirable to specify the educational strategies that should be used when presenting information to the users". Thus our instructional working hypothesis was that the AHA! system (as any general purpose AHS) could be better exploited through implementing a specific instructional strategy instead of concentrating on a generic "adaptive exposition" of content. Each group was therefore assigned a definite strategy to be matched with the subject matter. The selected strategies were the following: *Learning Styles* (Kolb 1999), *Scenario-based learning*, *Problem solving* (Smith & Ragan 1999) and *Increasing complexity.*

The leading hypothesis was therefore that the instructional strategy would provide the main guidelines for the developing of the conceptual model of the AHS, as well as indications about the kind of interaction expected between the learner and the system.

From the technical point of view, other hypotheses were necessary. According to De Bra *et al.* (1999) an Adaptive Hypermedia System architecture need at least two models to function properly: the *domain model* representing the subject matter or the content to be learnt; and the *user model* representing the learner's status of knowledge or his/her general profile, as well as a description of how to update it during the interaction with the system. AHA! is an implementation of this general architecture. (De Bra & Calvi 1998a, De Bra & Calvi 1998b, De Bra *et al.* 2000).

AHA!'s domain model consists of a network of *concepts*, each coupled with one single HTML page. The arcs of the network are of two types:

1. *Pre-requirement*, i.e. the relationship existing between a concepts A and B when the learner must know B prior to visiting A in order to get a complete understanding of A.
2. *Propagation*, i.e. the relationship existing between a concepts A and B meaning that, when the learner understands A, he also understands "something" of B. Propagation can be expressed in terms of percentage or absolute value.

The user model is actually a replication of the domain model for each learner (an *overlay model*, see De Bra *et al.* 1999) tagged with further information recording the learner's interactions, namely indicating if the learner visited a concept (i.e. the page coupled with that concept) and recording a number quantifying the learner's knowledge of the concept. Generally speaking, if a learner has all the necessary pre-requirements for a concept and he/she visits it, he will get a score of 100; else a lower score, e.g. 35.

With these models, AHA! supports two kinds of adaptation. The *link annotation feature means that* a link may change color for a specific learner according to the pre-requirements the learner has achieved. A link can be recommended, to-be-avoided, visited or neutral. The *selective release* feature means that pages may contain conditional blocks that are visualized to a single user according to his/her level of knowledge, e.g. if he/she visited a particular page, or got at least 50 for concept X. Because this kind of adaptation is at the code

level of the page, namely HTML, it's up to the designer to decide if the selective release is used only for content (e.g. showing a text or not) or if to use it for altering the navigation interface (e.g. building an adaptive menu or a dynamic course map)

AHA! therefore offers great flexibility, in terms of content structuring and navigation behavior design, as well as interface design. Our working hypothesis was to exploit AHA! as a low-level adaptive engine, on the top of which a more complex model can be built, with the method presented in the following section.

4. THE DESIGN METHOD

The design method includes five steps. Them main tools used are maps. In order to clarify the process we will present as example the design documentation produced by the group dealing with learning styles on the topic of effective email use (*email 3)*.

4.1 Concept Map

The first step is drawing a map of concepts. Each concept will store a knowledge value for each user, representing how much the student *knows* the associated page. A first innovation, not represented by AHA! but extremely useful for designers, is the introduction of *islands* of concepts (Botturi 2000), i.e. groupings of concepts into semantic areas.

Moreover, we used *abstract concepts* in order to support the more demanding interactions that the instructional strategies required. Abstract concepts are variables that are not coupled with any physical HTML or XML page or other file; they are specific elements suitable for maintaining additional information about the user, such as preferences, early choices, answers, etc.

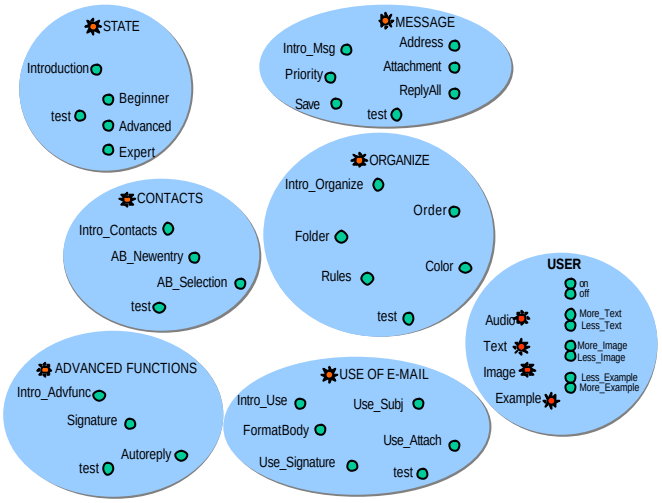

Figure 1. Concept map from the Learning Style application

Figure 1 shows seven islands with the concepts developed by the *email 3* group. Five islands gather the actual domain model (concepts are represented by dots). The *USER* island gathers the abstract concepts used for recording the learner's learning preferences, i.e. quantity of text, use of pictures, use of audio, quantity of examples. The *STATE* island records the initial knowledge level of the learner. Moreover, each content island contains one abstract concept (e.g. *MESSAGE, ORGANIZE,* etc., represented by stars), used for recording the overall knowledge degree for all the island's concepts. This knowledge level can be used in the navigation map to set conditions to show or inhibit the access to other islands. Because of this use, we call abstract concepts *launching pads.*

We suggest that this map should be designed without thinking explicitly how the application should interact with the user, rather just trying to represent the domain and the user as precisely as possible.

4.2 Pre-requirement Map

The concept map is then developed introducing pre-requirement arcs between concepts. Pre-requirement arcs can ideally connect page-related concepts and abstract concepts, but due to the particular way AHA! uses them, they will map differently when translated into the internal models of the AHA! system.
Each arc has a tag indicating the threshold of the requirement that must be fulfilled to understand the destination. As for the concept map, the pre-requirement map should as well be designed thinking of the subject matter "as is", without much concern for the user interface.

4.3 Propagation Map

Similarly to the previous map, propagation arcs can be introduced on the concept map. They also connect both page-related concepts and abstract concepts. The tag reports the weight of the propagation, which, as said, for AHA! can be relative (e.g. +30, which means 30%) or absolute (e.g. 30).

Figure 3 clarifies one possible use of abstract concepts in the content islands: thanks to the propagation rules they "collect" information about the learner's general knowledge in that area. This information may be used to disclose new areas to the student (see *Navigation and Interface Map* below). E.g., before enabling the ADVANCED FUNCTIONS area we may require that the MESSAGE area (collected by the *launching pad* MESSAGE) have a knowledge value of at least 75.

In other cases abstract concepts store information on the learner's preferences and attitudes, e.g. *Audio, Text, Image, Example* are Boolean variables that represents the learner's learning preferences as they have been stated during the interaction (thus representing his/her learning style).

As for the two previously presented maps, the propagation rules should be designed thinking of the subject matter "as is". Additional guidelines should be specified concerning abstract concepts and pads, which should be updated according to the desired tracking of the user profile. E.g., *audio* is updated according to the learner's request of having audio explanations.

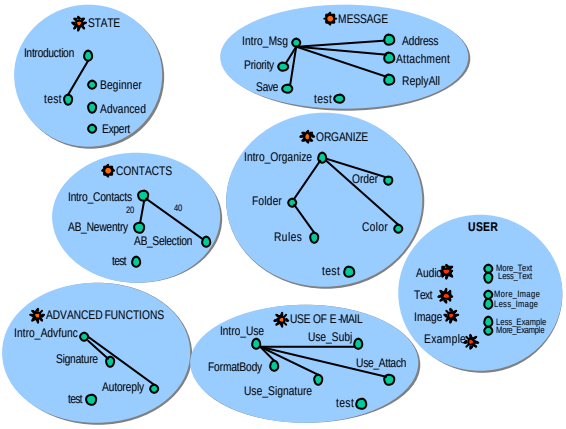

Figure 2. Pre-requirement map from the Learning Style application

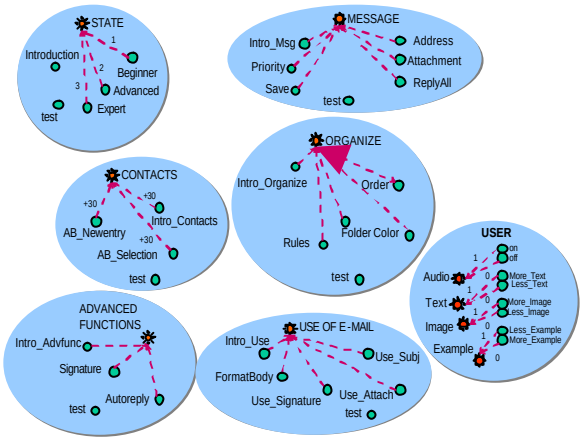

Figure 3. Propagation map from the Learning Style application

4.4 Navigation map and User Interface model

The fourth step introduces the major concern of interface design. In our opinion, in an educational context an adaptive application is useful inasmuch it offers an adaptive support through an intelligible and clear interface. Changes in the application may disorient or indispose the learner. We assumed as a design principle that the learner should be able to understand the rationale of the adaptation. The design of predictable interfaces can be conducted transforming the pre-requirement map into a navigation map.
We suggest below a procedure useful as a guideline:

1. Design the interface layout, defining functional screen areas and menus
2. Define what concepts/pages are linked directly from the menus and for each of them determine
   a. If anchor in the menu is always present (then do nothing)
   b. If the anchor in the menu appears only under certain conditions, then write them near the concept. These will be later implemented as conditional blocks.
3. For each pre-requirement arrow on concepts that are not in menus, decide if it will
   a. Be used for link annotation, then mark the arrow with A. These will be later implemented as annotated links.
   b. Be used for making a link appear or disappear according to the pre-requirement, then mark the arrow with D. These will be later implemented as conditional blocks.
   c. Be used only for tracking the user knowledge, but not for adaptive behavior (then do nothing).
4. Finally, introduce all other links between concepts that are not bound to pre-requirements or other adaptive behaviors.

Figure 4 shows the example navigation map. The interface layout presents a menu area (conditions and links are reported only for the MESSAGE and USE OF EMAIL islands) and a user preferences area. All pre-requirements within each island were used for conditional link annotation (tagging is not reported).

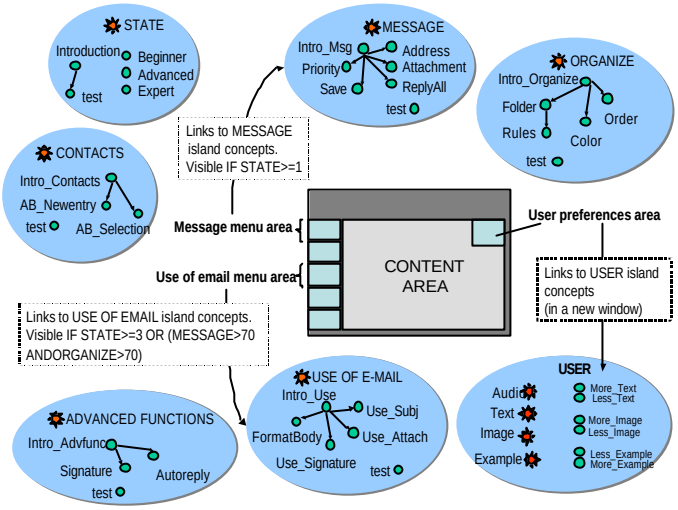

Figure 4. Navigation map from the Learning Style application

4.5 Content Implementation, System Set-up and Test

Content implementation means producing all the elements for the application. Their production strictly depends on the design and on the instructional strategy: for learning styles, the same concept should be presented with different media; for scenario-based learning great effort should be spent on creating "real" situations, etc. Moreover, a number of local decisions have to be taken about the structure of each page. One particularly relevant decision is the selection of alternate blocks with different media.

A practical issue is how to merge content within the general layout designed during the previous step. The whole implementation encountered some difficulties due to a few technical requirements of the AHA! version we used. For example AHA! does not support application-specific tags (e.g. by Macromedia Dreamweaver); moreover several characters commonly used by text and HTML editors are not recognized by the XML parser. These difficulties lead to increasing costs for page production and testing. While identifying the issues at stake, the Atelier did not develop guidelines for content implementation, leaving this an open issue.

Finally, the domain and user models should be translated into AHA! specific XML files, namely a file defining the pre-requirements and another defining the propagation rules. Moreover, a list of the files connected with concepts should be defined. Using both the prerequisite and propagation maps as a basis, this step proceeds in a straightforward way.

5. CONCLUSION

In this paper we presented a method for designing adaptive hypermedia courseware, as well as some specific issues concerning the design and implementation process. The method is based on maps that capture the core aspects of an adaptive hypermedia application: domain, user profile and interaction. The method was developed with real content and exploiting the AHA! system. It was proofed through implementing four different instructional strategies. This research aims at providing non-technical authors conceptual tools that support them in the design process for exploiting AHS in education.

Some open issues have been identified. The first and more important is the refinement of the model and its proof with different instructional strategies and different content. In particular, guidelines for the implementation of adaptive content should be developed. Moreover, it should be assessed if all the instructional strategies that may benefit from AHS can be developed following this method. Another issue is the portability of the proposed method to other AHS different from AHA!. At what level our methodology can be placed? It is more oriented to conceptual and functional design or to the implementation and logical one? Finally, the methodology lacks of a page model that represents conditional contents, and their showing/hiding conditions as well.

ACKNOWLEDGEMENTS

This research was conducted with the collaboration of Alberzoni D., Aus der Beck J., Crosta L., Di Nardo F., Franceschini P., Ghidoli B., Giuliani F., Henry W., Inglese T., Maino F., Pingitore C., Ricci E., Schoch M., Silvaroli M, Smacchia L., Spanò M., Turriziani L., and Vanucci F. Thanks also to Prof. Colombetti M. and Mazza R..

REFERENCES

Botturi L., 2000. *Seaway Tracker: An Adaptive Navigation Engine for Educational Applications*, BUL, Lugano.

Carro R.M., 2002. Adaptive Hypermedia in Education: New Considerations and Trends. In *6th World Multiconference on Systemics, Cybernetics and Informatics*. Orlando, Florida, July 2002.

Carro R.M., Breda A.M., Castillo G., Bajuelos A.L., 2002: A Methodology for Developing Adaptive Educational-Game Environments. *In Adaptive Hypermedia and Adaptive Web-Based Systems*. Lecture Notes in Computer Science 2347, Eds. De Bra, P., Brusilovsky, P. and Conejo, R., Berlin, Springer-Verlag, pp. 90-99.

Cristea A., Aroyo L., 2002. Adaptive Authoring of Adaptive Educational Hypermedia, In *AH 2002, Adaptive Hypermedia and Adaptive Web-Based Systems*, Lecture Notes in Computer Science 2347, Springer, pp. 122-132.

De Bra P., Aerts A., Houben G.J., Wu H., 2000. Making General-Purpose Adaptive Hypermedia Work. In *WebNet Conference*. San Antonio, USA, pp. 117-123.

De Bra P., Calvi L., 1998 [a]. AHA! An open Adaptive Hypermedia Architecture. *The New Review of Hypermedia and Multimedia*, Vol. 4, pp. 115-139.

De Bra P., Calvi L., 1998 [b]. AHA: a Generic Adaptive Hypermedia System. In *2nd Workshop on Adaptive Hypertext and Hypermedia. Pittsburgh*, USA, pp. 5-12

De Bra P., Houben G.-J., Wu H., 1999. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *10th ACM conference on Hypertext and Hypermedia*. Darmstadt, Germany, pp. 147-156.

Kolb D.A., 1999. Learning Style Inventory. Hay/McBer Training Resources Group, Boston.

Smith P., Ragan T., 1999. Instructional Design (2nd edition). Wiley & Sons Publisher, New York, USA.

Wu H., Houben G.J., De Bra P., 1999. Authoring Support for Adaptive Hypermedia Applications. In *Ed-Media 99 Conference*. Seattle, Washington. USA, pp. 364-469.