Cross-References in Web-Based Adaptive Hypermedia

Hongjing Wu, Erik de Kort

Department of Mathematics and Computing Science
Eindhoven University of Technology
PO Box 513, 5600 MB Eindhoven
the Netherlands

email: h.wu@tue.nl, erik.de.kort@asml.nl

Abstract: Many websites offer their users a lot of freedom to navigate through a large hyperspace. Adaptive hypermedia systems (or AHS for short) aim at overcoming possible *navigation and comprehension problems* by providing adaptive navigation support and adaptive content. The adaptation is based on a *user model* that represents relevant aspects about the user [2]. In most systems navigation support is tied to the existing link structure. To provide users with better understandable navigation, we discuss adaptive cross-references based on concept relationships in AHAM. We define an authoring tool to generate a cross-reference page for each page based on already defined relationships. With these added cross-reference pages, we can easily provide adaptive cross-references according to user features by AHAM.

Keywords: adaptive hypermedia, user modeling, navigation support, hypermedia reference model, adaptation rules.

1. INTRODUCTION

The introduction of World Wide Web has made hypermedia the preferred paradigm for user-driven access to information. Websites typically offer users a lot of freedom to navigate through a large hyperspace. Unfortunately, this rich link structure of hypermedia applications may cause some usability problems:

- A typical hypermedia system presents the same links on a page, regardless of the path a user has followed to reach this page. Even when providing navigational help, e.g. through a map (or some fish-eye view) the system does not know which part of the link structure is most important for the user. The map can thus not be simplified by filtering (or graying out) links that are less relevant for the user. Having personalized links or maps would eliminate some *navigation problems* that users have with hypermedia applications.

- Navigation in ways the author did not anticipate also causes *comprehension problems* for the user: for every page the author must take into account what foreknowledge the user has when accessing that page. In order to do so the author must at least consider all possible paths that lead to the current page. This is clearly an impossible authoring task because there are more ways to reach a page than any (human) author can foresee. In a traditional hypermedia application a page is always presented in the same way. This may result in users visiting pages

containing redundant information and pages that they cannot fully understand because they lack some expected foreknowledge. The website should really select the pieces of information that are shown on a page, based on a history of which pages the user has seen before, or provide easy access to extra information on the concepts presented in the page.

Adaptive hypermedia systems (AHS) in general, and adaptive websites in particular aim at overcoming the navigation and comprehension problems by providing *adaptive navigation support* and *adaptive content*. The adaptation (or personalization) is based on a user model that represents relevant aspects of the user such as preferences, knowledge and interests. We focus on simple Web-based systems that can only gather information about the user by observing the *browsing* behavior of that user. Each time the user "clicks" on a link the system uses the selected link to update the user model and to adapt the presentation accordingly.

In previous research we have shown that our AHAM model [2] can be used to describe content adaptation and link adaptation of AHS. Link adaptation as used in many AHS guides users towards interesting information by changing the presentation of link anchors. In general however, adaptation of the existing link structure alone is not enough to solve all users' navigation and orientation problems:

- It may not be possible to select a guided tour consisting of existing links, and of only pages that are interested for a given user, because the interesting pages may not form a connected sub graph in the whole link structure.
- It may take too many steps (possibly going through uninteresting pages) to guide the user to the page(s) that deal with the topic the user is interested in.

To improve adaptive navigation support based on basic link structure in AHS, we propose *cross-reference navigation support* to supplement the above cases. Cross-reference navigation support (or CRNS for short) aims to help the user when s/he navigates through hyperspace following cross-references, i.e. conceptually related information. A cross-reference page provides meta-information for a "normal" page in the form of links, and can thus be adapted according to user features.

While cross-references could be investigated at a more general level, we concentrate on its use in AHS that can be described in the AHAM model. This paper is structured as follows: in Section 2 we briefly review the AHAM reference model, thereby concentrating on the parts that are needed to describe adaptation functionality at an abstract level. Section 3 discusses how to generate cross-reference pages and how to provide cross-reference navigation support based on these pages. Section 4 draws conclusions and describes our future work.

2. AHAM, A DEXTER-BASED REFERENCE MODEL

Many adaptive hypermedia systems share parts of their architecture. Just like the Dexter model [4][5], tried to capture the facilities offered by hypermedia systems of its time (and of potential future systems), AHAM [2], (for Adaptive Hypermedia Application Model) describes the common architecture of adaptive hypermedia systems. Part of this common architecture is typical for Web applications: their event-driven nature, where each page access results in a user-model update and an adaptive

nature, where each page access results in a user-model update and an adaptive presentation. AHAM's overall structure is an extension of the Dexter model. According to AHAM each adaptive hypermedia application is based on three main parts:

- The application must be based on a *domain model*, describing how the information content of the application or "hyper-document" is structured (using concepts and pages).
- The system must construct and maintain a fine-grained *user model* that represents a user's preferences, knowledge, goals, navigation history and other relevant aspects.
- The system must be able to adapt the presentation (of both content and link structure) to the reading and navigation style the user prefers and to the user's knowledge level. In order to do so the author must provide an *adaptation model* consisting of *adaptation rules*. An AHS itself may offer built-in rules for common adaptation aspects. This reduces the author's task of providing such rules. In fact, many AHS do not offer an adaptation rule language; the way in which the user model is updated and the presentation adapted is then completely predefined.

The division into a *domain model* (DM), *user model* (UM) and *adaptation model* (AM) provides a clear separation of concerns when developing an adaptive hypermedia application. The main shortcoming in many current AHS is that these three factors or components are not clearly separated. Modeling an existing AHS in AHAM may not be straightforward because AHAM requires these parts to be made explicit, and the adaptive behavior to be described using adaptation rules. However, using AHAM enables us to clearly describe how an AHS works, how different AHS compare, and also how to design new and more powerful AHS.

The AHS consists not only of the three sub-models (mentioned above) but also of an *adaptation engine* (AE). The AE describes implementation dependent aspects of the AHS. In previous work [7] we described design issues for a general-purpose AE, and defined AHAM CA-rules to illustrate how sets of rules work together. We will use these rules to describe the generation of adaptive cross-reference in Section 3.

Figure 1 shows the overall structure of an adaptive hypermedia application in the AHAM model. The figure has been made to resemble the architecture of a hypermedia application as expressed in the Dexter Model [4][5].

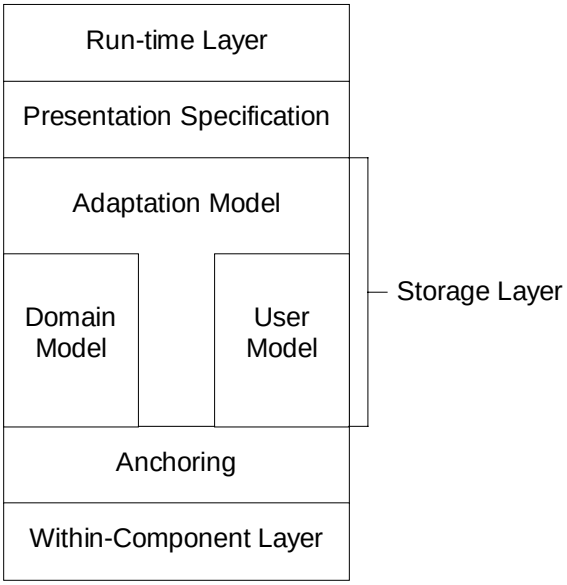

Figure 1 : Structure of adaptive hypermedia applications

In this section we only present the elements of AHAM that we will need to discuss how cross-reference navigation support (CRNS) can be expressed in AHAM.

2.1 The Domain Model

The domain model of an adaptive hypermedia application consists of *concepts* and *concept relationships*. Concepts are objects with a unique object identifier, and a structure that includes attribute-value pairs and a sequence of anchors.

A concept represents an abstract information item from the application domain. It can be either an atomic concept or a composite concept.

- An *atomic concept* corresponds to a fragment of information. It is primitive in the model (and can thus not be adapted). Its attribute and anchor values belong to the "Within-Component Layer" and are thus implementation dependent and not described in the model.
- A *composite concept* has a sequence of children (sub-concepts) and a constructor function that describes how the children belong together. The children of a composite concept are either all atomic concepts or all composite concepts. A composite concept with (only) atomic children is called a *page*. The other (higher-level) concepts are called *abstract concepts*.

The composite concept hierarchy must form a DAG (directed acyclic graph). Also, every atomic concept must be included in one or more composite concepts. **Figure 2** illustrates a part of a concept hierarchy.

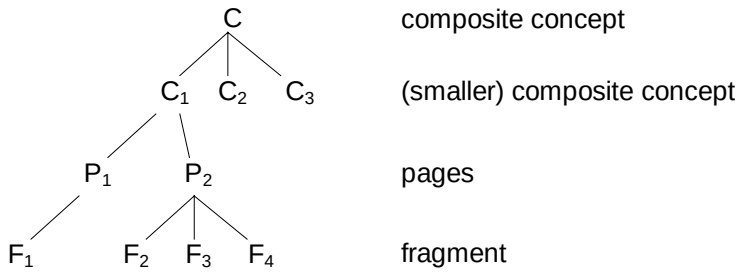

Figure 2 : Part of a concept hierarchy.

A *concept relationship* is an object (with a unique identifier and attribute-value pairs) that relates a sequence of two or more concepts. Each concept relationship has a *type*. The most common type is the hypertext **link**. In AHAM we consider other types of relationships (abstract relationships) as well, which play a role in the adaptation, e.g. the type **prerequisite**. When a concept C_1 is a prerequisite for C_2 it means that the user "should" read C_1 before C_2. This does not imply that there must be a link from C_1 to C_2. It only means that the system somehow takes into account that reading about C_2 is not desired before some (enough) knowledge about C_1 has been acquired. Through link adaptation the "desirability" of a link will be made clear to the user. **Figure 3** shows a small set of (only binary) concepts associated to one another by three types of concept relationships: prerequisite, inhibit, and link.

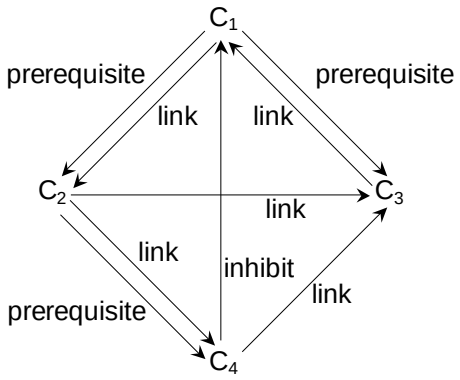

Figure 3 : Example concept relationship structure**.**

Apart from the "implicit" relationship type and set of relationships that form the concept hierarchy an AHS need not contain or support any other relationships. AHAM can thus also represent applications without traditional hypertext links (like e.g. in *spatial hypertext* [6]). A relationship graph defines a certain connection among a set of concepts. In Section 3 we will use relationship graphs to generate cross-reference pages; these can be used to supplement the basic adaptive navigation support in AHS.

The atomic concepts, composite concepts and concept relationships together form the *domain model* (DM) of an adaptive hypermedia application.

2.2 The User Model

A user model consists of named entities for which we store a number of attribute-value pairs. For each user the AHS maintains a *table-like structure*, in which for each concept in the DM the attribute values for that concept are stored. Because there can be many relationships between *abstract concepts* and *concrete* content elements like fragments and pages, a user model may contain many attributes per concept to indicate how the user relates to the concept. Typical attributes would be *knowledge level* (e.g. in educational applications) and *interest* (e.g. in encyclopedia, museum sites, or online mail-order catalogs). The user model may also store information about what a user has read about a concept, and for how long or how long ago, etc. Concepts can furthermore be used (some might say abused) to represent "global" user aspects such as preferences, goals, background, hyperspace experience, or a (stereotypical) classification like student, employee, visitor, etc. For the AHS or the AHAM model the true meaning of concepts is irrelevant.

In the sequel we will always consider UM as being the user model for a single user; we do not discuss adaptation to group behavior.

2.3 The Adaptation Model

The AHAM model targets adaptive hypermedia applications that follow the *request-response* paradigm that is typical for the Web. The interaction with the system is described through *events* generated by the user (or by an external source). Each event triggers user model updates and results in an adaptive presentation. In [7] we introduced a database-like language to express the effects of user actions as *condition-action rules* (AHAM CA-rules). This implies that we do not explicitly model *events* as events, but as updates to attributes that trigger rules. Accessing a web-page for instance will result in a Boolean "access" attribute of the page (in the user model) to become *true*. The small example below illustrates the structure of these rules. (A syntax description can be found in [7].)

C: **select** P.access
A: **update** F.pres := "show"
where Fragment(P, F) **and** F.relevance = "recommended"

In this example we first see that the *condition* for this rule is that P.access has become true for some page P. When this happens (and because there is no additional **where** clause in the condition) the *action* is executed. In the action we look at fragments F of page P. If a fragment is marked as "recommended" then that fragment will be shown. This is indicated as a *presentation specification* and represented as a "pres" attribute of the fragment.

Note that the AHAM CA-rule language is just a vehicle for describing how an AHS should perform user model updates and adaptation. It does not imply that we require AHS to use such a language. Even when an AHS has only a built-in behavior, we can still describe this using the AHAM CA-rule language. Also, we partition adaptation rules into *phases* to indicate that certain rules must always be executed before certain other rules. The phases include IU, the initialization of the user model, UU-pre, the user model updates that are performed before generating the presentation, GA, the

generation of the adaptation, and UU-post, the user model updates that come after the presentation. The phases are a convenience for ensuring that the execution of the rules has desirable properties such as *termination* and *confluence*, as discussed in [7].

2.4 The Adaptation Engine

An AHS does not only have a domain model, user model and adaptation model, but also an adaptation engine, which is a software environment that performs the following functions:

- It offers generic page selectors and constructors. For each composite concept the corresponding selector is used to determine which page(s) to display when the user follows a link to that composite concept. For each page a constructor is used for building the adaptive presentation of that page (out of its fragments). Page constructors allow for dynamic content like a ranked list of links.
- It optionally offers a (very simple programming) language for describing new page selectors and constructors. For instance, in AHA [1] a page constructor consists of simple commands for the conditional inclusion of fragments.
- It performs adaptation by executing the page selectors and constructors. This means selecting a page, selecting fragments, organizing and presenting them in a specific way, etc. Adaptation also involves manipulating link anchors depending on the state of the link (like enabled, disabled and hidden).

It updates the user model (instance) each time the user visits a page. The engine will change some attribute values for each atomic concept of displayed fragments in a page, of the page as a whole and possibly of some other (composite) concepts as well (all depending on the adaptation rules).

The adaptation engine thus provides the implementation-dependent aspects, while DM, UM, and AM describe the information and adaptation at the conceptual, implementation independent level. Note that DM, UM and AM together thus do not describe the complete behavior of an AHS. The same set of *adaptation rules* may result in a different presentation depending on the *execution model* of the adaptation engine.

3. CROSS-REFERENCES IN AHS

Sometimes users do not want to follow the suggested navigation order, but rather skip across to related information. It would therefore be helpful to have access to meta-information for the concepts the user is reading about, especially through cross-references that make related information readily reachable. These cross-references are a collection of links to all concepts relevant to a (page)concept. Relevance is determined by examining the concept relationship graphs: there must be an (in)direct connection between the concepts. Section 3.1 explains how to generate cross-reference pages for each page in the DM. Section 3.2 explains how to update the user model for generating the CRNS. Section 3.3 explains how to personalize the cross-references.

3.1 Generate Cross-Reference Pages

An authoring tool can generate a set of cross-references for each page to connected or related concepts. It depends on the author and system what pages or abstract concepts are included in this set. Also depending on the system, cross-references can be show

as a part of the original page or in a separate page. For brevity and ease of reference, we choose to organize a cross-reference set as an independent page called a *cross-reference page* (CRP). A (page)concept may be connected with several abstract concepts through one relationship graph or with several abstract concepts through different relationship graphs. A CRP should contain a list of links that can be grouped according to the type of relationship so that the user can easily find related concepts and go directly to the related concepts. The CRP should be adaptable to a user's knowledge state for every referenced concept. Before generating CRNS, we first need to add some requirements on the general definition of relationships in the DM of AHAM. We then define a function to generate CRPs from the relationship graphs defined in the DM.

AHAM provides a platform to define all kinds of relationships. For a clear understanding we distinguish between the hypertext link relationship, and other more abstract relationships such as *prerequisite* or *consists_of*. To allow a user to go directly from one concept to a related concept, we have to create a hypertext link. If an abstract concept has no direct representation (e.g. C_1 in Figure 2) a choice has to be made as to which hypertext links need to be created. For the sake of argument we will assume that all related pages in the concept hierarchy are included, but this is really an implementation decision and could possibly be a configurable option in the authoring tool.

Once all required hypertext links are created for a (page)concept, the corresponding CRP can be generated. We can define a function to generate CRP for each page by using relationship graphs defined in the DM. There are different ways to generate cross-references according to relationship type and connection distance (the length of the shortest path in the relationship graph). For example, a CRP may contain links to relevant concepts for just one specific relationship type *R*, or for all types of relationships. Connection distance can be used to exclude certain links, and may be used as tool to determine relevance.

Let us assume that *G* is a relationship graph with type *R*, *P* is a page, and *P.cross-reference* is a set consisting of relevant concept links to *P*. Relationship *R(C, P)* represents that *C* is directly related to *P* through relationship *R,* while $R^+(C, P)$ represents that *C* is directly or indirectly related to *P* through relationship *R*. Function *Generate* generates the CRPs from the *P.cross-reference* set. Function *Generate* is part of the authoring tool, so details are not relevant here. We illustrate four possible ways to the function to generate CRPs:

Algorithm 1 (only directly related concepts of relationship type R):

1. Select a relationship graph G with type R;
2. **for** all P in G **do**
   P.cross-reference := ∅;
   for all C in G **do**

if R(C, P) **then** P.cross-reference := P.cross-reference ∪ {C};

3. *Generate* the CRP based on *P.cross-reference*.

Algorithm 2 (directly related concepts of all relationship types):

1. **for** all G in Relationship_Graphs **do**
   for all P in G **do**
   P.cross-reference := ∅;
2. **for** all G in Relationship_Graphs **do**
   for all P in G **do**
   for all C in G **do**
   if R(C, P) **then** P.cross-reference := P.cross-reference ∪ {C};
3. *Generate* the CRP based on *P.cross-reference*.

Algorithm 3 (all related concepts of relationship type R):

1. Select a relationship graph G with type R;
2. **for** all P in G **do**
   P.cross-reference := ∅;
   for all C in G **do**
   if R^+(C, P) **then** P.cross-reference := P.cross-reference ∪ {C};
3. *Generate* the CRP based on *P.cross-reference*.

Algorithm 4 (all related concepts of all relationship types):

1. **for** all G in Relationship_Graphs **do**
   for all P in G **do**
   P.cross-reference := ∅;
2. **for** all G in Relationship_Graphs **do**
   for all P in G **do**
   for all C in G **do**
   if R^+(C, P) **then** P.cross-reference := P.cross-reference ∪ {C};
3. *Generate* the CRP based on *P.cross-reference*.

3.2 Updating the User Model

We briefly illustrate how the AHAM CA-rules are used to perform user model updates, in which the concept relationships play a role. Adaptation is normally based on *relevance* of or *recommendations* for concepts or pages. This is in turn deduced from aspects like *interest* or *knowledge* (which must exceed some *threshold* to be considered sufficient to be taken into account). Below we give a possible rule that decides whether a concept should be recommended based on whether the user has enough knowledge about all prerequisite concepts.

C: **select** C_2.knowledge
where C_2.knowledge ≥ C_2.threshold
A: **update** C_1.relevance := "recommended"
where Prerequisite(C_2, C_1)
and not exists (**select** C_3

where Prerequisite(C_3, C_1)
and C_3.knowledge < C_3.threshold)

The rule is triggered when the knowledge of concept C_2 is changed and when that knowledge then matches or exceeds the required knowledge threshold for C_2. The action of the rule sets the relevance value for C_1 to "recommended" if there are no unsatisfied prerequisites left for C_1. (For efficiency reasons only concepts C_1 are considered for which C_2 is a prerequisite. Other concepts cannot be influenced by the knowledge change for C_2).

In order to describe the user model updates, changes to relevance of pages, and also the resulting adaptation one needs many rules. We don't describe them here, not just because of the size restrictions for this paper, but also because every AHS has different behavior and thus also a different description using AHAM CA-rules.

3.3 Personalize the Cross-References

In this section we discuss how to present a CRP. The relevant links presentation in the reference page should be adapted to the user's features. For example, we use color metaphors as used for adaptive annotation to show the knowledge state for all links in the cross-reference page.

The following rule describes that when a CRP is displayed, the system shows the relevant concept links of this page by using the color metaphor for adaptation annotation.

C: **select** CR.access
A: **update** F.pres := "green"
where Fragment(CR, F) **and** F.relevance = "recommended"

Here F can be a page or an abstract concept. *F.relevance* has been calculated in UU-phase. The cross-reference page consists of a list of related page links or abstract concept links depending on constructors of the cross-reference page. From the cross-reference page users can go directly to related concepts. In this way we add additional navigation paths that we call link-independent navigation support.

4. CONCLUSIONS AND FUTURE WORK

With AHAM we are can define any relationship within AHS. We can then automatically generate cross-references based upon these relationships. Using the basic functionality of AHAM we are able to apply adaptation and personalization to these cross-references to provide users with a better understandable navigation environment. This method could be very useful for educational web applications; teachers should provide the relationships among concepts for the course. It is less practical in large information systems without a clear notion of an author. But the idea to provide relationship based cross-references to assist the user while browsing through hyperspace is applicable in web-based information systems in general. All that is needed are meaningful relationships among pages, e.g. provided by semantic web techniques.

5. REFERENCES

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

[2] De Bra, P., Houben, G.J., Wu, H., “AHAM: A Dexter-based Reference Model for Adaptive Hypermedia”. Proceedings of ACM Hypertext’99, Darmstadt, pp. 147-156, 1999.

[3] EI-Beltagy S. R., Hall, W., De Roure, D. and Carr, L. “Linking in Context”. Proceedings of The 12th ACM Conference on Hypertext and Hypermedia, pp. 151-160, 2001.

[4] Halasz, F., Schwartz, M., “The Dexter Reference Model”. Proceedings of the NIST Hypertext Standardization Workshop, pp. 95-133, 1990.

[5] Halasz, F., Schwartz, M., “The Dexter Hypertext Reference Model”. Communications of the ACM, Vol. 37, nr. 2, pp. 30-39, 1994.

[6] Marshall, C.C., Shipman, F.M., “Spatial Hypertext: Designing for Change”. Communications of the ACM, Vol. 38, nr. 8, pp. 186-191, 1995.

[7] Wu, H., De Kort, E., De Bra, P., “Design Issues for General Purpose Adaptive Hypermedia Systems”. Proceedings of the 12th ACM Conference on Hypertext and Hypermedia, pp.141-150, 2001.