Authoring and Delivery of Adaptive Electronic Textbooks made Easy

Ewald Ramp[1,2], Paul De Bra[1], Peter Brusilovsky[2]
[1]Information Systems Group
Department of Computing Science
Eindhoven University of Technology
[2]Dept. of Information Science and Telecommunication
School of Information Sciences
University of Pittsburgh
e_ramp@hotmail.com, debra@win.tue.nl, peterb@mail.sis.pitt.edu

Abstract: The vast majority of textbooks (even when offered on-line) are still traditional book-like static documents with a fixed structure and content. Authoring a textbook in a "simple" environment like Microsoft Word is much easier than using special authoring environments for adaptive electronic textbooks. In this paper we show how to translate a Word-based textbook into an adaptive website that provides adaptive navigation support (through link annotation) as well as adaptive presentation (through conditionally included explanations). The generated adaptive textbook is used through the powerful and highly customizable AHA! adaptive delivery platform. The essence (and novelty) of the presented approach is that the translation from Word (using an extended set of Interbook annotations) to AHA! is performed at the conceptual level. The further translation into (low level rules defining the) adaptive behavior is completely independent from the conceptual structure and content of the textbook (and can also be customized).

1. Introduction

The advantages of adaptive electronic textbooks over traditional paper (or pdf) ones are clear to most authors. Just putting a textbook online, perhaps translated into html, with links added to quickly jump to different chapters and sections, isn't enough. But when learners have the freedom to jump around between different topics they quickly discover that the links they follow do not correspond to any reading order the author of the textbook has foreseen. This results in a frustrating experience of visiting pages in which not previously studied concepts are used, in which details and comparisons are given the learner cannot yet comprehend, etc. An *adaptive* textbook can guide and help the user by recommending some links (and advising against other links), by inserting prerequisite explanations when needed, and by providing additional details only to the learners who are ready for them (Brusilovsky et al., 1998). Adaptive textbooks belongs to the class of adaptive hypermedia systems (Brusilovsky, 1996) and (Brusilovsky, 2001). Several pioneer projects explored the use of adaptive hypermedia techniques for developing better electronic textbooks. However, despite the demonstrated advantages of using adaptive hypermedia techniques to produce electronic textbooks they are still relatively rare.

We think that the main problem is the lack of proper authoring tools to develop electronic textbooks that could be easily accessible and understood by potential authors. InterBook (Brusilovsky et al., 1998), an old authoring platform targeted to electronic textbook authors is not supported for a number of years. The modern general-purpose adaptive hypermedia authoring and delivery platforms like AHA! (De Bra et al, 2003b, 2004) are powerful but hard to use for the majority of the potential textbook authors. Even a number of more specialized authoring systems such as ALE (Specht et al., 2002) or MetaLinks (Murray, 2003) are used mainly by their developers. To resolve the authoring problem, we developed a solution that combines features of InterBook and AHA!. It provides a MS-Word-based special-purpose authoring platform oriented to the target users, yet it allows delivering electronic textbooks that are the product of that authoring process using the versatile AHA! platform.

This paper describes a tool to generate adaptive textbooks automatically from normal textbooks, written using an ordinary word-processor like Microsoft Word. This tool extends the Word-based authoring process that was originally intended to produce electronic textbooks for the InterBook platform (Brusilovsky et al., 1998). It accepts Word source files prepared according to a few special rules and performs a high-level translation to the concept structures of the AHA! system. The end-result is an adaptive electronic textbook with *adaptive navigation* (including automatically generated menus and also *adaptive link annotation*) and with *adaptive presentation* (including the use of *conditional fragments* that can be used to add prerequisite explanations and additional details).

2. Authoring for InterBook using Microsoft Word

InterBook (Brusilovsky et al, 1998) is designed to help a course-author to transfer a normal textbook existing in electronic form into an adaptive electronic textbook (ET). It works on textbooks in RTF format, as produced by Microsoft Word. In order to use InterBook the textbook must be structured hierarchically, using "header" tags to indicate the start of chapters and sections. (Simply creating a line of text with a large bold font does not produce a recognizable section or chapter title.)

2.1. Publishing Electronic Textbooks on the Web with InterBook

To publish an ET on the Internet with InterBook, the author needs to go through 5 steps which are described below (Figure 1). In brief, the author must prepare the ET as a specially structured RTF file and then convert this file into InterBook format. The result of this process is a file with the Textbook in InterBook format which can be published on the Internet by the InterBook system.

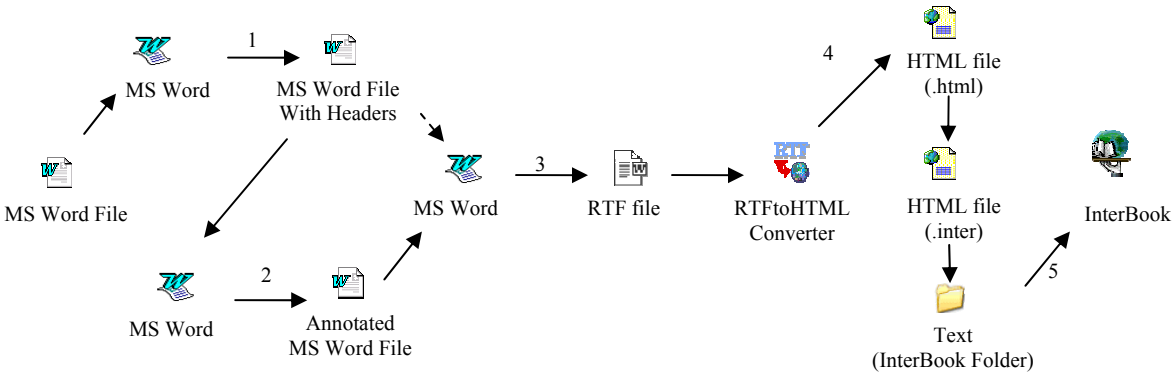

Figure 1. Serving an Electronic Textbook on the WWW with Interbook

- *Step 1. Structure the RTF file*. To let InterBook recognize the structure of the ET, the titles of the chapters should have the pre-defined paragraph style "Header 1", the titles of sections should be "Header 2", and so forth. The title of the textbook should have paragraph style "Title". This structure is necessary because InterBook's RTFtoHTML converter will use these section headers to decide how to divide the textbook into pages. The Word file can be converted to InterBook format straight away via step 3, but it won't be adaptive.
- *Step 2. Annotate the MS Word file*. The concept-based annotation of the ET is needed to let InterBook know which concepts correspond to which section. This knowledge lets InterBook help the reader of the ET through adaptive navigation support. Although it is not necessary to design the structure of concepts before writing the paragraphs and sections of text, most authors will start with an outline and structure of a textbook before writing anyway. An *annotation* is a piece of text in a special style and format, inserted at the beginning of each section (between the section header and the first paragraph). Annotations have the special character style (hidden[1] + shadowed). For each (sub)section, the author can provide an (optional) set of *outcome* and *prerequisite* (or *background*) concepts. The format for the outcome annotation is (out: concept-name1, concept- name2, etc.). The format for the prerequisite annotation is (pre: concept-name1, concept-name2, etc.).
- *Step 3. Save your file in RTF format*. An RTFtoHTML program is used to convert the ET into an HTML format, so Word must save the textbook in RTF format.
- *Step 4. Translate your RTF file to HTML*. The RTF file with the ET is translated into an HTML file by the RTFtoHTML program controlled by some specially designed settings. In particular, all annotations are translated into HTML comments with a special format. RTFtoHTML will extract the images and create a single HTML file with the same name as the RTF file, but with the *.html* extension. This extension needs to be renamed to *.inter* and the file needs to be transferred to a special "Text" folder on the InterBook server.
- *Step 5. Parsing into LISP structure and Serving on WWW*. When the InterBook server starts, it parses all interbook files in its "Texts" folder (i.e. all files with extension ".inter") and translates them into the list of

[1] Note that it may be necessary to turn-on the viewing of formatting marks for *hidden text*.

section frames. Each section frame contains the name and type of the unit, its spectrum, and its position in the original HTML file. The obtained LISP structure is used by InterBook to serve all the available textbooks on WWW along with providing all advanced navigation and adaptation features. All content which the user will see on the screen is generated on-the-fly with the knowledge about the textbook, the user model, and HTML fragments extracted from the original HTML file. These features of InterBook are based on the functionality of the Common Lisp Hypermedia Server CL-HTTP.

In Section 4 we describe how an ET developed for InterBook can be served by the AHA! system instead of the InterBook server, by replacing steps 4 and 5 by a newly developed translator R2Net. At the same time this new translator adds adaptive features not present in InterBook, like the *conditional inclusion of fragments*.

2.2. Advanced Authoring for InterBook

2.2.1. Creating Links to Internal Concepts and Section

InterBook can handle *links* to *concepts* and *sections*. They are not created as Microsoft Word links for historical reasons. A link to a *concept* or *section* consists of two parts. The first part is the name of a *concept* or *section*. It has to be formatted as a *hidden* and *double-underlined* text. The second part immediately following the URL is the *hot word* or *hot phrase* (the link anchor). This phrase has to be formatted with the character styles *single underlined* and *italic*. RTFtoHTML uses this formatting to generate a proper link. And because the link anchor is not hidden any word or phrase that appears in the ET can be turned into a link anchor.

2.2.2. The Glossary

The glossary is an important part for any (InterBook) ET. It should be considered as the visualized (and externalized) domain network. Each node of the domain network is represented by a glossary entry and each glossary entry corresponds to one of the domain concepts. The links between domain model concepts constitute navigation paths between glossary entries. Thus, the structure of the glossary resembles the pedagogic structure of the domain knowledge. In addition to concept description, in InterBook, each glossary entry provides links to all pages that introduce the concept. This means that an InterBook glossary integrates features of a traditional index and a glossary. In the Word file, the concept description should be added at the end of the file, with "Glossary" (Heading 1 paragraph style) indicating the start of the glossary, followed by the concepts with their concept description (with the concepts in Heading 2 paragraph style).

2.3. InterBook User Model Structure

Besides the *concept* structure and the *content* (paragraphs, sections, chapters) of an ET, the *adaptive* functionality of an ET depends mostly on the *User Model* (UM). Figure 3A shows different sizes of checkmarks next to the concepts on the right side of the screen. These correspond with *three knowledge levels* that a user can have for a concept. Each time the learner visits a page (a content element) the knowledge level of some *outcome concepts* is increased. The exact *amount* of knowledge contributed depends on whether the user already has knowledge of all *prerequisite* concepts and whether the page was already *visited* before. Figure 2 shows a graphical representation of this *UM Structure*. Note that the Knowledge attributes are omitted from the lower concepts for simplicity.

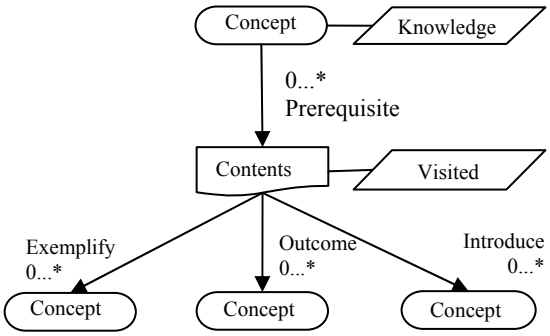

Figure 2. InterBook Concept Structure

3. Authoring for AHA!

AHA! (De Bra et. al., 2003, 2004) is a general purpose AHS, focused on *extensibility* and *expressivity*. It has strong support for *concept and concept relation authoring*, but little support for *content authoring*.

3.1. AHA! Concept Structure

Whereas InterBook explicitly distinguishes between *concepts* and *content*, AHA! considers everything as *concepts*, some of which have associated content and some of which do not. Figure 2 shows *concepts* as having a *knowledge* value in the UM and *contents* to have a *visited* attribute. In AHA! all concepts have both attributes by default. When the learner visits a page, the corresponding concept's *knowledge* and *visited* attributes are updated. Optionally, through the equivalent of the outcome *relationship*, the *knowledge* attribute of some other concepts can be updated as well. AHA! is very flexible. New attributes can be introduced, and new types of concept relationships. The new relationships can be associated with rules that define how the attributes of concepts are updated in the UM. An example of a newly added feature is discussed in Section 4.

3.2. Creating Applications for AHA!

Since AHA! version 2.0 a graphical tool has been available that lets authors draw the structure of *concepts* and *concept relationships* for an ET. This tool is described in (De Bra et. al., 2004). So unlike in InterBook where the concepts and concept relationships are defined as hidden constructs that are scattered throughout the textbook this structure is represented by a nice graph and visualized and edited using the *Graph Author* tool.

The content (pages) of an AHA! application is created using any HTML editor. Each page must have a corresponding concept in the concept graph. Since AHA! version 3.0 links can not only refer to other pages but also to concepts. The main reason for the translation described in this paper is that authors prefer to use a word processor to create a textbook over an HTML editor to create lots of individual pages.

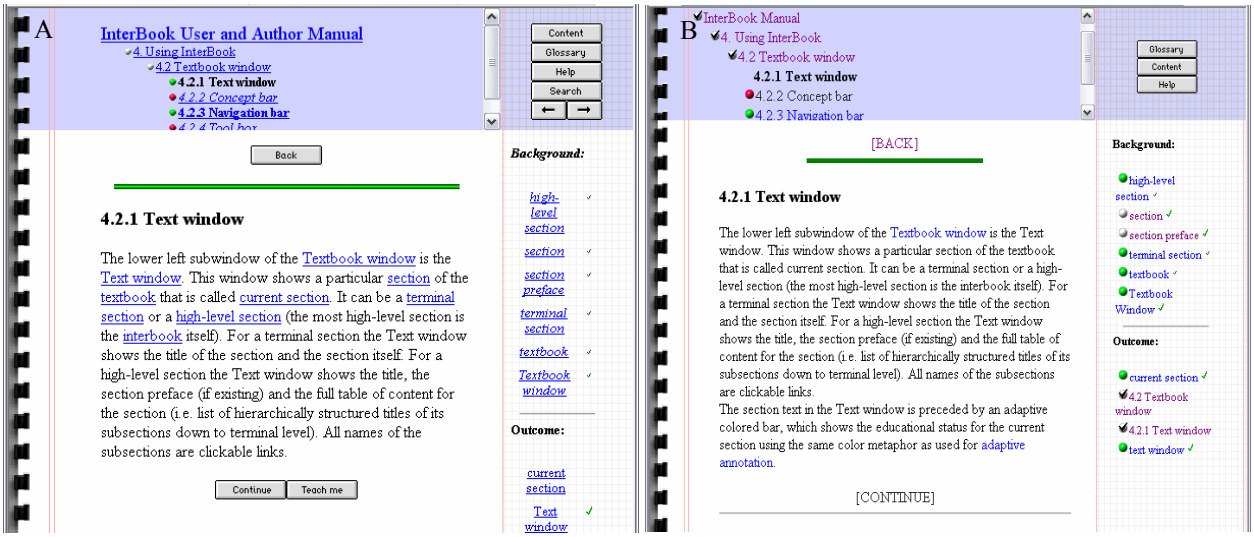

Figure 3A. InterBook User Interface; B. Translated InterBook User Interface on AHA!

AHA! has a very flexible presentation and layout module, first introduced in (Brusilovsky et al, 2003). It allows for a different layout for different types of concepts (like for pages and for glossary items), consisting of frames with menus giving access to the hierarchical structure of chapters, sections and paragraphs, prerequisite concepts, outcome concepts, etc. It allows an author to choose different link annotations through colors as well as icons, etc. Figure 3 below shows a page from the InterBook Manual, shown as InterBook presents it (part A) and as AHA! can approximate it (part B), using the automatic translation described in Section 4. It is possible to mimic the InterBook layout even more closely in AHA! but we have decided not to do this for reasons explained below. Every aspect of the presentation, like which frames exist, where which frames appear, what backgrounds and buttons to use, which

link colors, which link annotation items, is defined through a layout specification. The presentation style of many different adaptive hypermedia systems can be emulated in AHA!. The InterBook presentation style is just one example.

4. High-Level Translation from InterBook to AHA!

The goal for translating the InterBook authoring format was twofold: create the ability to serve existing and future InterBook applications through the Java-based AHA! server instead of the Lisp-based InterBook server, and also to make it possible to do the authoring of new AHA! applications in MS Word, by using that same translation.
In our first attempt to translate from InterBook to AHA! (De Bra et al, 2003a) our main concern was to create a convincing, near perfect emulation of InterBook's behavior through the AHA! server. This included not only the adaptive capabilities, but also the complete presentation and layout. The translation was completely one way, from the low level InterBook format (output of step 5 in Figure 1) to the lowest level AHA! constructs of concepts and adaptation rules.

The new InterBook to AHA! translation is more concerned with obtaining a rich functionality on the target system than with replicating InterBook exactly. Rather than creating exactly the same presentation we have decided to alter a few elements: In InterBook the "Back" and "Continue" buttons are not annotated, so the learner cannot see whether they lead to a recommended page or not. By using textual links in AHA! they automatically become annotated, the text color indicating whether the link is recommended. Also, the links in the lists of background and outcome concepts are annotated (with link color and colored balls) to indicate whether it is recommended to use these links or not. Some InterBook features are missing in AHA!, including the "Teach Me" facility to generate a list of links to pages explaining a concept, and the "Search" facility. We also added completely new features, described in Section 5.

4.1. Authoring Through Translation

The authoring-through-translation process we created is based on the InterBook authoring process, using the new version of RTFtoHTML: R2Net. Compared to our earlier translation attempt (De Bra et al, 2003a) the major change is that the input is now the RTF file (instead of the set of files used by the InterBook server) and that the output is the high-level concept graph used by the AHA! Graph Author tool, not the low level set of concepts and adaptation rules the AHA! engine uses. Figure 4 shows the translation process in detail. All annotations from InterBook (see Section 2) are supported, together with the extension described in section 5.

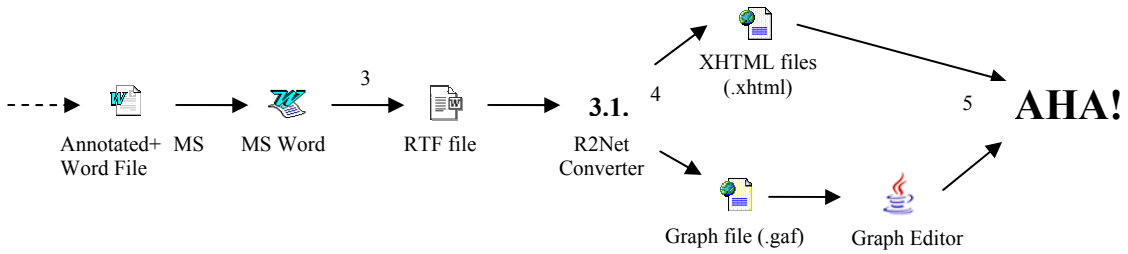

Figure 4. Serving an Electronic Textbook on the WWW via InterBook to AHA! translation

Translating input files is not enough. They are based upon a *User Model* without which they would loose a lot of meaning. In our translation the InterBook Concept Structure, shown in Figure 2 is mapped onto an AHA! Concept Structure shown, as shown in Figure 5. To allow this translation to be made, we added outcome relations to AHA! using its ability to define arbitrary *concept relation types* (in an xml-based format). These concept relation types are then bound to a generic translation into *adaptation rules* once (also in xml). The translation from RTF to the Graph Author format requires no knowledge of the internals of the AHA! system. And the translation from Graph Author format to the low level AHA! adaptation rules requires no knowledge of the (InterBook) source format from which the Graph Author file was generated. As such, it is very useful for authoring-through-translation, because after we have made the translation once, we can rely on the existence of the required concept structures and their low level translations on the AHA! system, and only consider translating the contents and the exact relations, but not the later

low level translation of the concept structure. Figure 5 shows schema of the obtained resulting Concept Structure, again with the attributes from the lower concepts omitted.

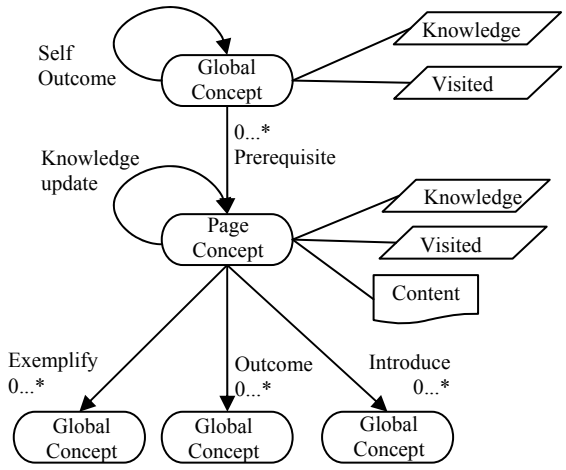

Figure 5. The Translated InterBook Concept Structure

5. Extended Annotations

AHA! has some capabilities, not available in InterBook. As a result, there was no RTF markup corresponding to that functionality. With the new translation we wanted to not only translate InterBook to AHA!, but also extend the content authoring process to support more of AHA!'s capabilities. Therefore we have extended the RTF markup and the conversion through R2Net for several new relations, reusing objects and conditional fragments.

5.1. New Relations

One of the last studies on InterBook involved the extension of the *outcome* and *prerequisite* relations to include *introduction*, *example*, *is-a* and *part-of* relations. We decided to also support these relations in our translation, even though the exact functionality is still under study. That poses no problem however, because their inclusion in the *conceptual structure* and the translation from the RTF file to the AHA! format is independent from the (low level rules defining the) *adaptive behavior* for these relations (which is expressed elsewhere in AHA!'s concept relationship templates). It allows us to support the relations in the translation, even though changes might still be made to their functionality in a later stage.

The *annotation format* is very similar to that of outcome and prerequisite. In *hidden* and *shadowed* character style, at the beginning of the section (same as for outcome and prerequisite), the author needs to include (<relation>: concept-name1, concept- name2, etc.), where <relation> is *intro* for the introduction of concepts, *exam* for concepts that are exemplified in the section, *isa* to indicate an is-a relation with other concepts, and *partof* to indicate a part-of relation with other concepts. The first two are generally used to annotate sections; the later two are usually used in the glossary to indicate relations between concepts

5.2. Conditional fragments in RTF

In AHA! it is possible to use conditional fragments to (not) include some text in one page. In InterBook, this was not possible, and therefore there was no RTF markup available to support this. To be able to use this functionality from AHA! through RTF files too, we extended the RTF markup, thereby trying to stay close to the original InterBook idea, that it should be easy to use, also for novices, but also tried to include a maximum of the expressive power of AHA!. The compromise consists of using near-natural language and allowing a shortcut for expert users. The basic form is:

```
(if: <condition> ;<text>)
```

and should be in *hidden* and *shadowed* font style. <text> defines the text fragment that is conditionally displayed, and it can either be a *single* text block (to use when <condition> evaluates to true) or have an entry for *both* the true

and false case of the condition, separated by a semicolon (;). Hence a semicolon can not be used within these text blocks.

<condition> is more complicated, it would be defined as follows, where *concept* can be any concept name:

```
<condition> = (not? {suitable|visited|(very? well)? known|#} concept)
     ({and|or} (not? {suitable|visited|(very? well)? known|#} concept))*
```

Or in plain English, it consist of at least one attribute/concept pair, optionally preceded by "not", optionally followed by a Boolean operator ("and" or "or") and another attribute/concept pair (again with optional "not"), and with arbitrary (valid) placing of parentheses. This can then again be followed by more Boolean operators and possibly negated attribute/concept pairs. The attributes translate to the AHA! concept attributes as follows:

- "suitable" and "visited" equal the "suitability" and "visited" attributes
- "very well known" translates to "knowledge>=95" (InterBook Knowledge Level 3)
- "well know" translates to "knowledge>=66" (InterBook Knowledge Level 2)
- "known" translates to "knowledge>=33" (InterBook Knowledge Level 1)
- "#" translates to "knowledge>= #"", with # being any integer number, allowing a shortcut to arbitrary values for more advanced users.

An example of a correctly formatted conditional fragment (except for the *hidden* and *shadowed* markup) would be:

```
(if: not known interbook system and suitable interbook system; some text
for people without (much) knowledge about InterBook)
```

which translates to AHA! format as :

```
<if expr="!(interbook_manual.interbook_system.knowledge&gt;=33)
&amp;&amp; (interbook_manual.interbook_system.suitability)">
<block>
some text for people without (much) knowledge about InterBook
</block>
```

5.3. Reusable Objects in RTF

One feature of AHA! is the conditional inclusion of reusable objects. Although it is a very powerful feature, it would require the transfer of a lot of information via the RTF file to the final application. An annotation which fully supports this feature would become a very complex annotation schema. However, having the conditional fragments, we decided to reduce conditionally included reusable objects to simple reusable objects, because with a thoughtful use of the conditional fragments in combination with reusable objects, it would still be possible to produce a similar effect as conditionally included reusable objects have.

A reusable object is a fragment of text that can easily be made to re-appear on several different pages in the final application. The main markup here is a *hidden* and *shadowed* text fragment, in the form of:

```
(reuse:concept1)
```

where *concept1* is any concept name (similar to "(pre:concept1)" for prerequisites). The text fragment of the object that will be displayed, is either a *glossary entry* (in which case there should be a glossary entry with description for the concept), or it can be defined *inline* by using (reuse:concept2, def) in *hidden* and *shadowed* markup *before a paragraph*[2]. That whole paragraph will then be used as text fragment for that object. Concept2 should be a *new* concept. Any occurrence of (reuse:concept, def) plus the following paragraph or (reuse: concept), will be replaced with the aha object tag;

```
<object name="interbook_manual.concept" type="aha/text">
```

which will be replaced with the relevant text fragment during the presentation of the final application.

6. Conclusion

The difficulty of authoring Adaptive Electronic Textbooks often depends on which Adaptive Hypermedia System it is meant for. Some powerful AHS like AHA! have a more cumbersome contents creation than a system like InterBook which uses an annotated MS Word file. We have advocated the use of authoring through translation to allow for the use of the much easier contents creation of one system, while still being able to use the adaptive power of another. We have shown the feasibility of this approach through an InterBook to AHA! translation. The strongest aspect of this translation is that it is performed at the high conceptual level. The translation starts from the RTF file generated by MS Word, and results in a conceptual structure that can be visualized in AHA's Graph Author tool. This translation is orthogonal of the translation of the conceptual structure to the low level adaptation rules used by

[2] Inline declared text fragments will not show up in the glossary in the presentation of the final application.

the AHA! engine. As a result, authors creating courses, and adaptation designers, defining the adaptive behavior that corresponds to the concept relationships, can work independent of each other. Also, the content authoring, the conceptual structure, and the presentation and layout specification are now all independent.

Acknowledgements

We thank the NLnet foundation for the financial support for realizing the AHA! system, and the PROLEARN Network of Excellence for providing support for networking and dissemination of adaptive educational systems, and the University of Pittsburgh for their hospitality.

7. References

Brusilovsky, P. (1996) *Methods and techniques of adaptive hypermedia.* In P. Brusilovsky and J. Vassileva (eds.), User Modeling and User-Adapted Interaction 6 (2-3), Special Issue on Adaptive Hypertext and Hypermedia, 87-129.

Brusilovsky, P. (2001) *Adaptive hypermedia.* User Modeling and User Adapted Interaction 11 (1/2), 87-110.

Brusilovsky, P., Eklund, J., and Schwarz, E. (1998) *Web-based education for all: A tool for developing adaptive courseware.* Computer Networks and ISDN Systems (Proceedings of Seventh International World Wide Web Conference, April 1998) 30 (1-7), 291-300.

Brusilovsky, P., Santic, T., De Bra, P. (2003) *A Flexible Layout Model for a Web-Based Adaptive Hypermedia Architecture*. Proceedings of the AH2003 Workshop, TU/e CSN 03/04, Budapest, Hungary, May 2003, pp. 77-86.

De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D., Stash, N. (2003a) *AHA! The Adaptive Hypermedia Architecture*. Proceedings of the ACM Hypertext Conference, Nottingham, UK, August 2003, 81-84.

De Bra, P., Santic, T., Brusilovsky, P. (2003b) *AHA! meets Interbook, and more...* Proceedings of the AACE ELearn 2003 Conference, Phoenix, Arizona, November 2003, 57-64.

De Bra, P., Stash, N., Smits, D. (2004) *Creating Adaptive Textbooks with AHA!*. Proceedings of the AACE ELearn'2004 Conference, Washington DC, November 1-5, 2588-2593.

Murray, T. (2003) *MetaLinks: Authoring and affordances for conceptual and narrative flow in adaptive hyperbooks*. In P. Brusilovsky and C. Peylo (eds.), International Journal of Artificial Intelligence in Education 13 (2-4), Special Issue on Special Issue on Adaptive and Intelligent Web-based Educational Systems, 199-233.

Specht, M., Kravcik, M., Klemke, R., Pesin, L., and Hüttenhain, R. (2002) *Adaptive Learning Environment (ALE) for Teaching and Learning in WINDS*. In: Lecture Notes in Computer Science, Vol. 2347, (Proceedings of Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002), Málaga, Spain, May 29-31, 2002) Berlin: Springer-Verlag, pp. 572-581.