Original citation:
Cristea, Alexandra I., Smits, D. and De Bra, P. (2005) Writing MOT, Reading AHA!
Converting between an authoring and a delivery system for adaptive educational
hypermedia. In: International Workshop on Authoring of Adaptive & Adaptable
Educational Hypermedia (A3EH 2005), Amsterdam, The Netherlands, 19 July 2005

Permanent WRAP url:
http://wrap.warwick.ac.uk/61414

A note on versions:
The version presented in WRAP is the published version or, version of record, and may
be cited as it appears here.For more information, please contact the WRAP Team at:
publications@warwick.ac.uk

http://wrap.warwick.ac.uk/

Writing MOT, Reading AHA!
- converting between an authoring and a delivery system for adaptive educational hypermedia -

Alexandra Cristea, David Smits and Paul de Bra

Faculty of Mathematics and Computer Science, Eindhoven University of Technology
PB 513, 5600MB, Eindhoven, The Netherlands

Abstract. This paper reports about the recent advances towards establishing a common platform for adaptive educational hypermedia (AEH) authoring. We present the conversion from MOT, a dedicated authoring system, to AHA! used in this context as delivery system for AEH. Moreover, we describe two new representation languages that emerged in the process: a common format for defining the static material, CAF, and an extended adaptation language for the description of the dynamic behaviour, LAG. Finally, some evaluations are shown and conclusions are drawn.

1. Introduction

Adaptive Educational Hypermedia (AEH) [3],[4] is dedicated to bringing personalization in the closed and open corpus hypermedia, e.g., the WWW. The AEH community is increasing in size and volume of research, indicating the importance of this topic at the global scale.

Within AEH, *authoring* is a serious problem, which has only recently started being given more attention by the research community. As authoring of static educational content is quite different from authoring of adaptive content, there is a growing understanding that a great deal of help is necessary for authors. In order to make the authoring process easier, in the sense of 'authoring once, delivering many' [18], there are two major possible approaches:

1. a common language, a *lingua franca*, used by all authors of AEH, and secondly
2. the use of *converters* between AEHs.

Very few such attempts have been made so far, mainly clustering along the second possible approach. In the following we shall discuss the current converters between AEHs shortly.

A pioneer work in the direction of converters was that from Interbook [6] to AHA! [14]. The approach in Interbook to AHA![13] is similar to the one presented in this paper, in the sense that Interbook is used as an authoring tool, and AHA! as a delivery tool for AEH. Moreover, the delivery end is the same. However, here the similarity ends. In Interbook, authors can create AEH applications in simple Word documents [13]. Adaptive behaviour is introduced via annotations in these documents. The main purpose in this conversion was to recreate the layout, look and feel of Interbook presentations, i.e., adaptive multiframe views. The conversion is done via a *compiler*, which doesn't use a semantically relevant intermediate format of representation. Finally, adaptivity in Interbook is mainly based on prerequisite relations and linking to glossary. No higher-level, reusable pedagogical strategies can be represented.

Another similar converter was built between MOT [9] and WHURLE [16]. This approach resembles the one presented in this paper, from the point of view of the authoring end, MOT. However, similar to the Interbook to AHA! converter, the approach in MOT to WHURLE [18] also uses a compiler. Moreover, as WHURLE cannot represent adaptive, but only adaptable educational hypermedia, the MOT system dynamics (adaptation maps) cannot be part of the conversion. Similarly to the current approach, the MOT to WHURLE system convertor has also been tested in practice with students.

Another interfacing exercise has been done between Claroline [8] and AHA! [14]. The approach in Claroline and AHA! [2] is quite different from the one presented in this paper, as, first of all, there is no conversion as such: Claroline and AHA! function as two separate modules of the same adaptive collaborative delivery system, ASCIL. The adaptive support for collaborative learning is integrated with the information contained in the User Model (student), which is kept in operation by AHA!. That is to say that the adaptive tasks are the result of individual learning in AHA! as a starting point [2].

Finally, an interesting conversion from the point of connecting the academia prototypes with extant commercial applications is the conversion between MOT and Blackboard [18]. Here, a regular educational environment, the Blackboard^{TM} Learning Management System [4], is added adaptivity and personalization by means of authoring the goal-oriented material in an Adaptive Hypermedia authoring system, MOT [11], and delivering it in Blackboard via a conversion to the SCORM specification. This represents the first attempt to connect Adaptive Hypermedia and Learning Management Systems [18].

2. Authoring with the MOT system

MOT [1] is a generic authoring system for adaptive educational hypermedia. Information about MOT (and the software) can be found at [16]. MOT is based on the LAOS model [11] for authoring of adaptive hypermedia, and implements some of the LAOS layers, as shown in the following subsections.

In order for material authored in MOT to be converted and represented in another adaptive hypermedia system, at least some components of the MOT system have to be converted and interpreted. Naturally, the more components will be integrated in the conversion, the more complete and accurate the end-result will be.

2.1 Domain Maps
Domain maps structure and organize the actual resources of the learning environment, as well as their intrinsic characteristics. These resources represent the content that is traditionally authored in any learning environment. Domain maps in MOT follow the LAOS specifications, by consisting of hierarchical domain *concept maps*, that are in turn built of typed attributes. Moreover, MOT also allows, according to the LAOS model, another type of relations between concepts, beside the *hierarchical* ones, called *relatedness* relations. Attributes in MOT represent, same as in LAOS, different content variants (or aspects) of the domain concept they belong to. For instance, the *title* attribute is the most concentrated way of representing a concept, which can, however, also be represented via the *text* attribute, via the *introduction* attribute, a.s.o.

A sample screenshot of how a domain map can be authored in MOT, and thus what has to be converted into another system is represented in Figure 1.

2.2 Goal and Constraints Maps

The goal and constraints model contains *all* information (resources and links between them) of the adaptive hypermedia system relevant for a delivery purpose. According to LAOS, within the educational domain, this model filters, regroups and restructures the domain model, with respect to an instructional goal. Moreover, these maps add the necessary information (content) that is goal-dependent. In MOT, this *goal* is a pedagogic learning goal, therefore these maps are called *lesson maps*. Thus, these maps add all the necessary pedagogic material and linking for the student. These maps are represented also as concept maps, according to the LAOS model. In MOT, they are implemented as a simplified *overlay* over the domain maps. Only in these lesson maps will the concepts be ordered, as the *order* in which the information is presented to the learner is dependant on pedagogic constraints and teaching strategies. Moreover, other pedagogic information can be authored via these maps, as for instance, *labels* and *weights* can be added to concepts, to semantically annotate the intended use of the respective concepts. Figure 2 shows ordered concepts (see numbers on the left side in the left frame), as well as semantically labeled concepts (e.g., 'beg' for beginners, 'beg_vis' for visual beginners and 'beg_ver' for verbal beginners; the visual-verbal differentiation is taken according to the ILS questionnaire [13]).
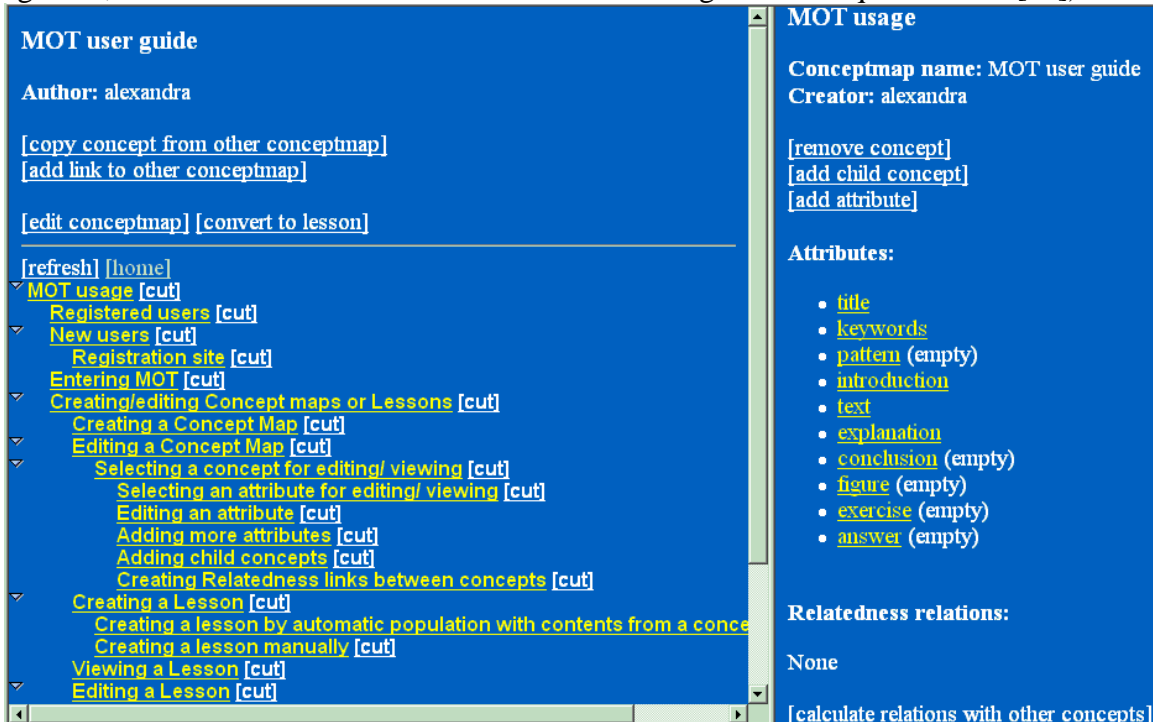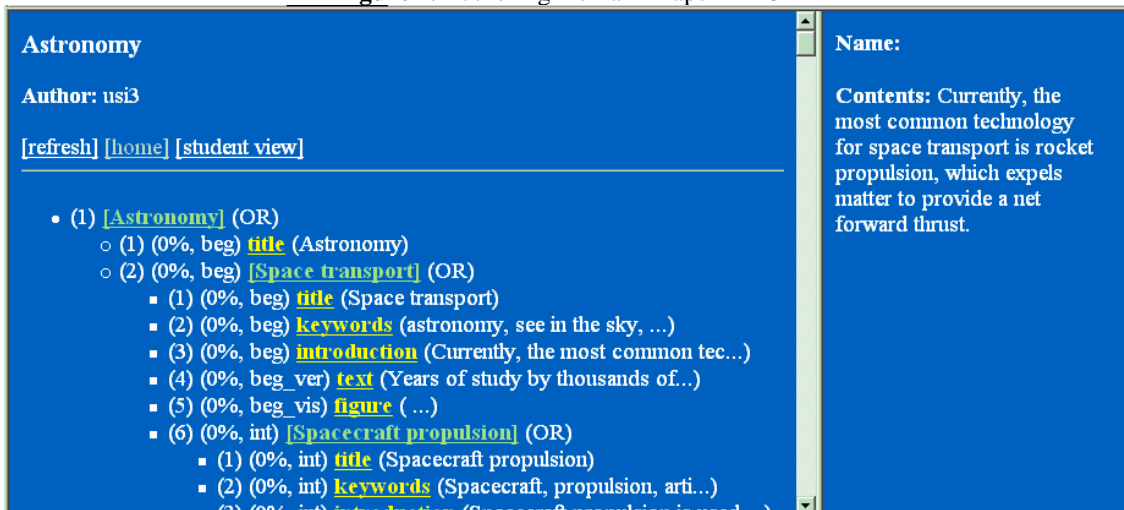

Figure 1. Authoring Domain Maps in MOT

Figure 2. Authoring Lesson Maps in MOT

2.3 User & Presentation Maps

The LAOS model specifies the representation and authoring of separate user maps and presentation maps.

The *user maps* should contain all the necessary variables and initial values that are necessary to represent the user (for educational applications, the *learner*). Typical variables are knowledge level, interests, learning styles, etc. The user model can be an overlay to either the goal and constraints or the domain model (e.g., the knowledge of a certain topic represented by a goal and constraints model concept). Alternatively, the user model can also contain free variables (such as background knowledge).

Presentation has to take into account the physical properties and the environment of the presentation and provide the bridge to the actual code generation for the different platforms (e.g., HTML, SMIL, XHTML, etc.). Presentation makes the difference between different devices of display, such as handheld devices, desktops, laptops, etc. This part in the LAOS model is concerned with the formatting so that the information appears nicely in the page, with questions such as the ideal page length, where chapters of the presentation should be cut to form pages, how and where multimedia should appear (from the point of view of display possibilities), colours, fonts, etc. In a simplified authoring experience, these settings should be generated fully automatically, in order to make the authoring process easier. For more complex authoring, the authors should be able to specify this type of settings. Taking a further step, presentation should be concerned with other user-related settings that are however not directly dependant on the user, such as adaptation to different load on the network, traffic peaks, scalability, alternate routes, distributed servers, etc.

The most recent implementation of MOT actually contains these two separate authoring systems, and some initial testing has been done with the user map authoring environment. However, in the current conversion into AHA! these systems haven't been used. Instead, a simplified version of user and presentation map initialization have been used, directly represented in the adaptation maps, as follows.

User model variables were defined as (in the following, UM stays for User map, GM for Goal and Constraints (lesson) map, PM for Presentation map and DM for domain map):

- o *Overlay user map variables* declaration: e.g., the knowledge variable for every concept in the lesson map:

```
// VARS UM.GM.Concept.knowledge
```

- o *Independent user map variables* declaration: e.g., the stereotype variable for any user

```
// VARS UM.GM.stereotype1
```

The actual initialization of these variables is done in the first part of the adaptation map:

- o *Overlay user map variables* declaration: e.g., the knowledge variable for every concept in the lesson map is set initially to 0:

```
initialization ( // … other initializations
while UM.GM.Concept.knowledge != 0 ( UM.GM.Concept.knowledge = 0)
)
```

- o *Independent user map variables* declaration: e.g., the stereotype variable for any user in using this lesson is set initially to 0:

```
initialization ( UM.GM.stereotype1 = beg )
```

Presentation model variables were defined as:

- o *Overlay presentation map variables* declaration: e.g., the display ('show') variable for every concept in the lesson map:

```
// VARS PM.GM.Concept.show
```

o *Independent presentation map variables* declaration: e.g., the Boolean variables determining if a table of contents ('Menu'), 'next' button or list if 'To Do' items appears in the presentation:

```
// VARS PM.GM.Menu, PM.GM.ToDo, PM.GM.Next
```

The actual initialization of these variables is done in the first part of the adaptation map:

o *Overlay presentation map variables* declaration: e.g., the showability variable for every concept in the lesson map is set initially to true for concepts without specific semantic labels, so they can be visualized by any type of user:

```
initialization ( // … other initializations
while GM.Concept.label = null
  ( PM.GM.Concept.show = true ) // … other initializations
)
```

o *Independent presentation map variables* declaration: e.g., the table of contents and 'To Do' list variables are set to false, so that a linear presentation results (a *tour*, in which the learner only needs to press the 'next' button to proceed through the course):

```
initialization ( // … other initializations
PM.GM.Menu = false
PM.GM.ToDo = false // … other initializations
)
```

During the interaction with the user (learner) some of the values of these variables can be dynamically changed (depending on the variable type), as in the following subsection.

2.4 Adaptation Maps

The adaptation model is the only part of the LAOS model describing the dynamics of the adaptation process. Static data exchange is covered also by standards such as SCORM, etc. Whilst the other layers are concerned with the static properties, variables and values required for personalization, the adaptation model brings them together and specifies how they will be used. Hence we can consider that the other sub-models in LAOS specify the ingredients of the personalization process, whilst the adaptation model specifies the recipe. The adaptation model itself is based on the LAG model, the model of three layers of granularity. LAG is a classification method for adaptive techniques as *direct adaptation rules*, *adaptation language* and *adaptation strategies*. The repartition on layers follows the increasing level of semantics. MOT is implementing all three levels, for increased compatibility and flexibility.

Direct adaptation rules build the assembly language of adaptation. These rules are the basis of every adaptive system, but are, just as every assembly language, system-specific. In MOT, an example of such a (generic) assembly level IF-THEN rule is, e.g., a rule about showing concepts with a weight above a certain threshold:

```
if GM.Concept.weight > 10
then  ( PM.GM.Concept.show = true )
```

The semantics of the rule above is not evident. It may mean, for instance, that the weight represents the *importance* of the respective concept, and therefore concepts with importance above 10 should be shown. This would imply that important concepts have a threshold above 10. Such a rule doesn't help the author who has to label the concepts with weights, and might consider important concepts with a threshold above 20, for instance. Also, the same rule might have a completely different semantics, such as the weight representing the *difficulty* of the concept. Then, the rule can be read as: show concepts with a difficulty above 10.

Moreover, the typical assembly rules are not *generic*, as the one above, but *specific* (i.e., they can only be applied to a given, specific concept in the domain or lesson map). In MOT, specific

rules can be applied as follows. The example rule checks if the title of the specific concept called 'video presentation' has been accessed, and deduces then that the user is visual:

```
if '\Main topic\video presentation.title'.access == true
then ( UM.GM.stereotype1 = vis )
```

The LAG *adaptation language* [12] is developed to increase the level of semantics of rules applied. Therefore, it allows, e.g., *repetitive* actions to be performed within a WHILE statement (as opposed to a list of IF-statements). For instance, a rule stating that, as long as there are non-labeled concepts still available, make them readable, is presented below:

```
while GM.Concept.label == null
   (  PM.GM.Concept.show = true )
```

The LAG language has also constructs inspired from games, such as the notion of *enough* conditions satisfied to proceed to a next level. For instance, a rule stating that, if the concepts are labelled as being for intermediate ('int'), and the user is an intermediate, then the concepts should be shown, is written as follows:

```
if enough(
    GM.Concept.label == int
    UM.GM.stereotype1 == int
    , 2 ) then
   (  PM.GM.Concept.show = true )
```

Within the 'enough' construct there are two conditions, and the number following shows how many of those have to be satisfied in order for the concept to be shown. As the number is 2, both conditions have to be satisfied. However, other combinations are possible with this construct.

Moreover, the LAG language borrows structural semantics from the domain or lesson maps. For instance, attributes such as the type or order of a given concept in the domain map can be addressed (although cannot yet be converted into AHA!), as follows:

- Type of Attributes (in domain maps) usage: e.g., selection of titles:
  ```
  DM.Concept.Attribute.Type == Title
  ```
- Order of Attributes (in lessons) usage: e.g., selection of the first lesson in a lesson container:
  ```
  GM.Concept.Order == 1
  ```

The LAG language has also other constructs that introduce structural semantics, such as *generalize* and *specialize* commands [12], which are not further detailed here.

The *adaptation strategies* build the actual *adaptation maps* and are written as LAG programs that can represent teaching or pedagogic strategies. A typical LAG program conform to the latest development consists of the following four parts:

```
// Description
// Variables
initialization ()
implementation ()
```

The *description* represents the semantic label of the strategy at the highest level of the LAG hierarchy.

In MOT, adaptation strategies can be reused in different contexts and for different combinations of domain and lesson maps, as required by the LAOS model. Therefore, the *variable declarations* can be used for checking the compatibility between domain, lesson and adaptation map. For instance, if a strategy uses a the label `GM.Concept.label` variable of the lessons to decide what to show to the user, it only makes sense to apply it to a lesson where these labels are not null and have the same domain (labels as in Figure 2, left frame). Moreover, if more than one adaptation strategy is applied to a domain, a possible clash can result from two strategies using the same variable. The *initialization* part is used for instantiating all the user map variables that are used by the strategy, as well as for initializing the presentation map variables (such as what will be shown to

the user the first time he enters the system). As said, these two initialization types can be omitted if the user model is initialized as according to LAOS, in a separate interface, but for the current project they are initialized here. The implementation is used for the actual description of the strategy steps, to describe the interaction with the user (learner).

3. The Common Adaptation Format (CAF)

In order for the MOT static content to be converted to AHA!, an intermediary step was introduced, in the form of the conversion to a so-called 'common adaptation format' (CAF). This format represents the same elements as extant in the MOT mysql database, but in an XML language, which is more appropriate for web conversions, as well as semantically clearer for humans. Moreover, this representation of MOT is nearer to the LAOS XML Schema representation [1]. The idea is that with this format it will be easier to transfer between systems, as well as that it can be used as an intermediate language for a platform whereto and wherefrom systems can write and read complete adaptive hypermedia descriptions. Therefore, although we proceed via a 'converter between AEHs' approach in this research, the results can be used towards defining a 'lingua franca' for AEH description. CAF therefore represents a proposal for the static component of the AEH description, whilst the LAG-language describes the system dynamics component. The CAF language DTD is presented in Figure 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT CAF (domainmodel?, goalmodel?)>

<!ELEMENT domainmodel (concept+)>
<!ELEMENT concept (name, attribute*, concept*)>
<!ELEMENT attribute (name, contents)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT contents (#PCDATA)>

<!ATTLIST contents
    weight CDATA ""
    label CDATA ""
>

<!ELEMENT goalmodel (lesson)>
<!ELEMENT lesson (contents*, lesson*)>
```

Figure 3. The CAF Language

The CAF format encodes *domain* and *lesson* maps. As the figure shows, domains need to encode at least one concept, which have lists of attributes and sub-concepts, just as in the MOT hierarchy (see also Figure 1). Attributes can have a name, representing the type of attribute, and the actual data they contain.

Figure 4 shows a domain map with at least one concept, named 'Adaptive', which has at least one sub-concept, named 'Adaptive Hypermedia'. The latter concept has at least one attribute, with the name 'Title' and the contents 'Adaptive Hypermedia'.

```
<CAF>
<domainmodel>
    <concept>
      <name>Adaptive</name>
      <concept>
        <name>Adaptive HyperMedia</name>
            <attribute>
              <name>title</name>
              <contents>Adaptive HyperMedia</contents>
            </attribute>
        ...
      </concept>
...
</domainmodel>
</CAF>
```

Figure 4. Domain Maps in CAF

Lesson maps are built of a hierarchy of sub-lessons, just as in MOT (see also Figure 2). Figure 5 shows an extract of a lesson description in CAF, with a high hierarchy lesson container with two sub-lessons, one pointing to the title attribute of the 'Adaptive Hypermedia' concept of the 'Adaptive' concept map, and the other one to the text attribute. This lesson container also contains at least one more lesson container. The content of the sub-lessons is actually a pointer to the corresponding concept attribute, just as in MOT. Sub-lessons can also have weights and labels, as shown in the figure.

Currently, CAF translates all contents in the MOT domain and lesson maps with the exception of the relatedness relations.

```
<CAF>
 ...
<goalmodel>
  <lesson>
    <contents weight="0" label="">Adaptive\Adaptive
HyperMedia\title</contents>
    <contents weight="0" label="">Adaptive\Adaptive HyperMedia\text</contents>
    <lesson>
      <contents weight="0" label="">Adaptive\Adaptive
          HyperMedia\Welcome\title</contents>
      <contents weight="0" label="vid">Adaptive\Adaptive
          HyperMedia\Welcome\video</contents>
    </lesson>
      ...
  </lesson>
```

Figure 5. Lesson Maps in CAF

4. AHA! as a delivery system for AEH created in MOT

AHA! is an open source adaptive hypermedia Web-based adaptive engine, based loosely on the AHAM model [20]. Information about AHA! (and the software) can be found at [1]. AHA! was created written in an '*assembly language*' of adaptive hypermedia, in the sense that any higher-level adaptation paradigm can be expressed in terms of AHA! and simulated by the AHA! engine [13]. AHA! offers lower level *direct adaptation rules* (as defined by the LAG model), based on the fact that, behind the diversity of ways of adaptation, there is a limited set of basic adaptation methods and techniques [5],[7].

The conversion engine from MOT to AHA! uses the CAF format domain and lesson map descriptions and the adaptive strategies written in LAG (user, presentation and adaptation maps) to create AHA! applications (adaptive presentations in AHA!).

Figure 6 presents an example of an AHA! application authored with MOT and delivered in AHA!. The central frame of the image presents a number of sub-lessons from the lesson map, belonging to the same lesson container.
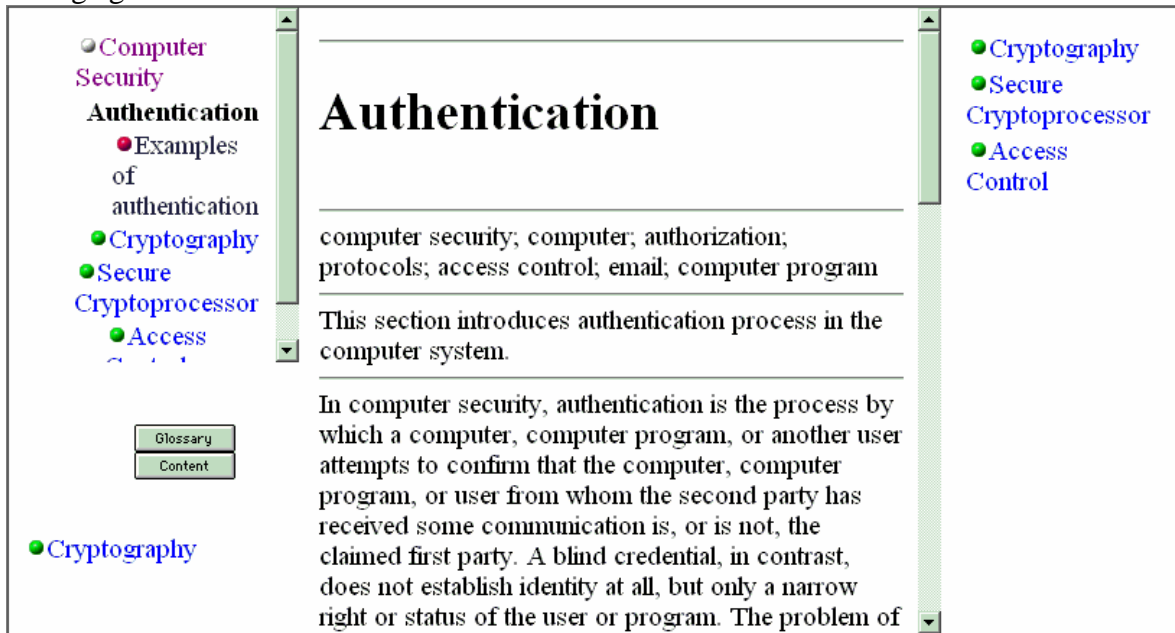
Figure 6. An adaptive application in AHA! authored with MOT

These sub-lessons are delimited with a separator line in-between them. The actual decision if to display the sub-lesson or not is dependent on the adaptation strategy contents, as described in section 2.4.

The left upper frame presents the 'Table of Contents'. This is adaptively annotated with green bullets for concepts that are ready to be visited, white bullets for already visited ones and red bullets for concepts for which the user is not ready yet, according to the standard AHA! color framework. Again, the decision about the color of the bullet in front of the respective link is determined by the adaptive strategy.

The right frame presents a list of 'To Do' (i.e., to read) concepts, that are recommended by AHA! based on the adaptive strategy.

The left lower frame presents only one concept, the 'Next' concept. This is the same as the first concept in the 'To Do' list.

The existence of these frames in the converted resulting AHA! application is also dependant on the settings in the adaptive strategy, as explained in section 2.4.

5. Discussion and Conclusions

This paper presents one of the first attempts to convert information between two different adaptive educational hypermedia systems: an authoring system and a delivery system. It is interesting to note that the two systems start from a different basic paradigm:

- o MOT is based on LAOS and therefore stresses the notion of separation of concerns: domain, lesson, adaptation maps are authored separately, with great focus on explicit semantics. This notion is required by the 'authoring once, delivering many' authoring paradigm: if the adaptive educational hypermedia content has an intertwined representation of static and dynamic material, the reuse of this content will be difficult, if not impossible, in a different setting. Moreover, this separation of concerns allows authors to specialize in

only one of the authoring targets (e.g., a domain specialist writing domain maps can be different from a pedagogy specialist, which creates adaptive pedagogic strategies), collaborating with each-other and reusing each-others materials.

o AHA! is based on the assembly-language approach. Moreover, in AHA! the statics and dynamics of the AEH are interlinked. This approach functions perfectly for delivery purposes, as it allows faster response to the user (student). However, reading the semantics out of the model, or reusing it in other contexts is not possible.

The process of conversion from MOT to AHA! has been evaluated with the help of a class of 4th year undergraduate students of Computer Science at the Eindhoven University of Technology. The students were following a course on Adaptive Hypermedia, and were asked to perform a project constructed based on this conversion process. The first results show that students unfamiliar with any of the two systems were able to produce materials via MOT, convert it and visualize it in AHA!. Students came with interesting alternatives for pedagogic labels for the material in the lesson maps, with their own variations of the adaptive strategies, and most of them created rich multi-media material presentations. The students replied to a set of questionnaires which are not discussed here due to lack of space.

It is important here to note that, as a result of the first experiments, some changes to the CAF language were performed. However, more seem to be required for the future, such as the use and expression of relatedness relations in MOT, for instance. Also, CAF needs to be extended according to LAOS model.

Extensions for the adaptation language are also being considered. Similarly to the extension required for the CAF language, the processing of relatedness relations is necessary, in order to be able to display, given a certain condition, for instance, the related concept content. Other extensions planned for the near future are the explicit differentiation between the display of a link to a certain content versus the display of the content itself. Also, parent-child relations within concept hierarchies have been introduced after these first tests have taken place, and are going to be extended with level information.

A change that occurred immediately after the first tests was the moving of the three systems (MOT, converter and AHA!) on-line. The next generation tests were performed with this new environment.

This paper represents yet another step to establish a common platform and a common format for the representation of adaptive educational hypermedia. From our first tests with different systems it is clear that this common format has to have a rich semantics.

References
[1] AHA!. http://aha.win.tue.nl.

[2] Arteaga, C., Fabregat, R., Eyzaguirre, G., Merida, D. Adaptive Support for Collaborative and Individual Learning (ASCIL): Integrating AHA!and CLAROLINE, Adaptive Hypermedia conference (AH'04), Eindhoven, August 2004, The Netherlands, Springer, LNCS 3137, 279-282.

[3] Beaumont, I., and Brusilovsky, P. (1995) Adaptive educational hypermedia: From ideas to real systems. In H. Maurer (Eds.), Proceedings of ED-MEDIA'95 - World conference on educational multimedia and hypermedia, Graz, Austria, June 17-21, 1995. Charlottesville, AACE. - pp. 93-98.

[4] Blackboard, http://www.blackboard.com/

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

[6] 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, 14-18 April 1998) 30 (1-7), 291-300.

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

[8]Claroline, Open Source e-learning, http://www.claroline.net/

[9]Cristea, A. I. & De Mooij, A. Adaptive Course Authoring: My Online Teacher. Proceedings of ICT'03, Papeete, French Polynesia, 2003.

[10] Cristea, A. What can the Semantic Web do for Adaptive Educational Hypermedia? International Peer-Reviewed On-line Journal "Educational Technology and Society" 7(4), Special Issue on Ontologies and the Semantic Web for E-learning, IEEE, LTSC, pp 40-58, 2004.

[11] Cristea, A., De Mooij, A. LAOS: Layered WWW AHS Authoring Model and its corresponding Algebraic Operators. In Proceedings of WWW'03, Alternate Education track. (Budapest, Hungary 20-24 May 2003). ACM.

[12] Cristea, A.I., and Calvi, L. The three Layers of Adaptation Granularity. UM'03. Springer.

[13] De Bra, P., Santic, T., Brusilovsky, P., AHA! meets Interbook, and more... Proceedings of the AACE ELearn 2003 Conference, Phoenix, Arizona, November 2003, pp. 57-64.

[14] De Bra, P. and Calvi, L., AHA! an open adaptive hypermedia architecture. The New Review of Hypermedia and Multimedia 4 (1998), 115-139.

[15] Felder, R.M. & Soloman, B.A. (2000). Learning styles and strategies. http://www2.ncsu.edu/unity/lockers/users   /f/felder/public/ILSdir/styles.html

[16] Moore, A., Brailsford T. J. & Stewart, C. D. (2001) Personally tailored teaching in WHURLE using conditional transclusion. Proc. of the twelfth ACM conference on Hypertext and Hypermedia, Denmark.

[17] MOT (My Online Teacher), http://wwwis.win.tue.nl/~acristea/mot.html

[18] Power, G., Davis, H.C., Cristea, A.I., Stewart, C. and Helen Ashman, H., Goal Oriented Personalisation with SCORM, ICALT'05, IEEE, July 5-8, 2005, Kaohsiung, Taiwan.

[19] Stewart, C., Cristea, A.I., Brailsford, T. and Ashman, H. (2005) 'Authoring once, Delivering many': Creating reusable Adaptive Courseware, WBE'05, February, 2005, Grindelwald, Switzerland.

[20] Wu, H. A. Reference Architecture for Adaptive Hypermedia Applications, doctoral thesis, Eindhoven University of Technology, The Netherlands, ISBN 90-386-0572-2, 2002.