IS SEMI-AUTOMATIC AUTHORING OF ADAPTIVE EDUCATIONAL HYPERMEDIA POSSIBLE?

Alexandra Cristea
Department of Computer Science
Eindhoven University of Technology
P.O. Box 513, Eindhoven
The Netherlands
a.i.cristea@tue.nl

Abstract

Adaptive Hypermedia (AEH) is considered, in principle, superior to regular Hypermedia, due to the fact that it allows personalization and customization. However, the creation of good quality AEH is not trivial. Nowadays, a lot of research concentrates on the authoring challenge in adaptive hypermedia. We previously introduced the LAOS model, a five-layer adaptive hypermedia authoring model that describes AEH in a detailed way, to allow flexible re-composition of its elements, according to the personalization requirements. However, such a detailed structure claims a lot of time to populate with AEH instances. Alternatively, we propose semi-automatic authoring techniques that populate the whole structure based on a small initial subset that has been actually authored by a human. We analyze here the different possible initial subsets, and the resulting structures, based on the LAOS architecture. Moreover, we examine if the flexibility of the whole was in any way affected by the replacement of human authoring with automatic authoring. We see the latter as yet another step towards adaptive hypermedia that '*writes itself*'.

Key Words

Adaptive Educational Hypermedia (AEH), Authoring of Adaptive Educational Hypermedia, design patterns, adaptive patterns

1. Introduction

Adaptive hypermedia started as a spin-off of hypermedia and Intelligent Tutoring Systems (ITS). Its goal was to bring the user model capacity of ITSs into hypermedia. However, due to technical restrictions, such as bandwidth and time constraints, AEH only used very simple user models. This simplicity gave AEH its power, as, all of a sudden, the application fields were multiple and implementation was easier. Early AEH research concentrated on variations of simple techniques of adaptive response to changes in user model [4,5,6]. No wonder, therefore, that most of AEH development was research-oriented, with application only for the limited domain of courses the researchers themselves were giving (AHA! [17], Interbook [6], TANGOW [7]) and with very rare commercial applications (Firefly).

However, with the development of the *semantic Web* [24], and the new community strive towards developing *ontologies* [22] for various domains, adaptive hypermedia (AH) starts to stand out as the vehicle of transport for the static information coming from all these sides, as the best way of bringing it to life.

Moreover, AH starts moving from its traditional application domain, education, to others, especially the commercial domain which is eager to be able to provide customization. Indeed, we see the phenomenon very often that people from other communities re-invent adaptive hypermedia for their own purposes and applications (e.g. Amazon).

Given these facts, the AH community has the obligation to make adaptive hypermedia more widely known and more accessible for the different research and user communities.

From the point of view of user community, authoring is one of the biggest issues [3,9].

Therefore, there is a strong need for powerful AH authoring tools [4,9].

Previously [2,8-16] we have analyzed different aspects of AH authoring, proposed LAOS, a framework for adaptive hypermedia (authoring), and implemented MOT, a system that serves as test bed for the LAOS framework, and that has been already used in some given settings. Moreover, we discussed automatic, adaptive authoring principles and hinted at implementation possibilities.

In this paper we focus on automatic authoring techniques in the LAOS model [15]. A few of these automatic techniques have been tested in practice [13]; however, this is beyond the scope of the current paper.

The remainder of the paper is organized as follows. In section 2 we sketch the LAOS model. Section 3 elaborates on semi-automatic transformations that allow *designer adaptation*. In section 4 we conclude by summarizing our contributions. The Annex presents the basic definitions.

2. LAOS Review

2.1 The LAOS components

To show what type of semi-automatic authoring techniques are possible in LAOS, we have to briefly describe LAOS first. The LAOS model (figure 1), introduced in [15], is a generalized model for *dynamic adaptive hypermedia authoring*, based on the AHAM model 27. The model comprises five components:

- the *domain model* (DM),
- *goal and constraints model* (GM) or *lesson model* (LM) for AEH,
- *user model* (UM),
- *adaptation model* (AM) and
- *presentation model* (PM).

These components are shortly reviewed in the following.

The *domain model* (DM) contains the basic *domain concepts* of the contents, and their representation, in the form of *domain (concept) attributes*. As can be seen in figure 1, the DM is a hierarchical model. This hierarchy can be seen as the hierarchy of chapters, sections and sub-sections in a book. Each component in this hierarchy is a concept. Concepts in LAOS must have a *semantic* unity, so that they can be labeled appropriately, and reused. The domain attributes are these labels for the concepts. Domain attributes in LAOS represent different aspects (views) about the same concept; for instance, an *introduction* is one of the aspects of a concept. These domain attributes can be specified by any recognized, given standard (e.g., for education, IMS [20], LOM [21], SCORM [23], etc.) or be designer-defined domain attributes.

The formal definitions of the domain model components are given in the Annex.

The *goal and constraints model* (GM; second layer in Figure 1) has its origin in the book–course or book–presentation metaphor. If the first layer (DM) was the book, this second layer extracts from the book exactly the information from that book that is needed for a specific presentation. Therefore, the GM is a constrained version of the domain model (DM). The *constraints* are based on some *goal*, so the full name of this layer is the 'goal and constraints model'. Moreover, a presentation (GM) can be based on one or more books (DM). At the same time, from one book, several presentations can be produced, so for one DM, multiple GM versions can be generated (this is why the GM layer is so dense - see Figure 1).

Therefore, the GM-DM relation is a multi-multi relation.

This division of the contents into DM and GM doesn't restrict the user view, but is designed only to give it more flexibility. The actual presentation seen by the user should, in principle, be based on the GM only, but can also contain elements of the DM (for instance, for more information about a goal and constraints concept from the GM, other domain attributes of the respective DM domain concept, or other DM domain concepts related to it can be referred). This increases the flexibility and expressivity of the created adaptive presentations, but, more importantly, separates links based on *content relatedness* (such as appear in the domain model) from links based on *presentation structure* (which now appear only in the GM).

The GM becomes the lesson model (LM) in educational context, as it represents the first filtering step towards putting together and organizing the material for the lesson.

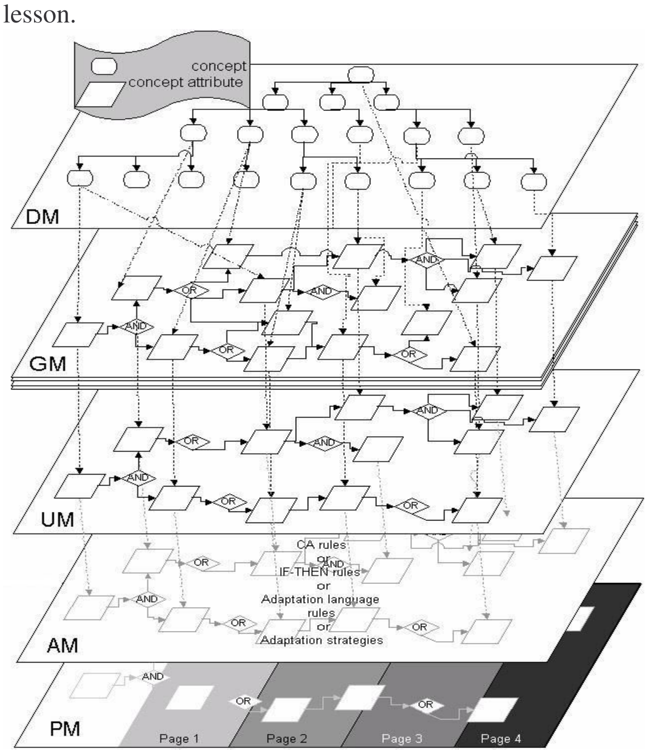


Figure 1. The five level AHS authoring model

The formal definitions for the components of the goal and constraints model are in the Annex.

The *user model* (UM, third layer in Figure 1) can be a pure overlay model, as in AHAM [26]. A slightly more advanced approach is to represent the UM [12] as a concept map, allowing explicit relations between the variables within the UM, instead of hiding them between the adaptation rules.

The *adaptation model* (fourth layer in Figure 1) is actually a more complex layer, as it is the one that determines the whole dynamics of the adaptive hypermedia system. Traditionally, this layer is composed of IF-THEN rules. For instance:

IF concept.knowledge > threshold THEN concept.show

We actually consider that for easy authoring, a higher level expression of adaptive behavior is necessary, and therefore created a new, three-layer adaptation model [9]. The IF-THEN rules are still there, as low-level *assembly adaptation language*, but higher level specification is possible via the medium level programming *adaptation language* or the high level *adaptation strategies*. The whole description of these extensions is beyond the scope of the present paper.

The *presentation model* (PM; the lowest level in Figure 1) is provided to reflect the physical properties and the environment of the presentation. It is there to create the bridge to the actual code generation for the different platforms (e.g., HTML, SMIL [25]). This paper does not go into details about the PM.

2.2 Authoring in LAOS

STEP 1: *Create the domain model skeleton* (DM); this comprises writing of concepts and deciding about their hierarchy
STEP 2: *Extend the domain model* (DM); this means deciding if to use only standard concept attributes or to create extra ones
STEP 3: *Fill-in concept domain attributes* (DM); i.e., write the contents of the defined attributes
STEP 4: *Create the goal and constraints model skeleton* (GM); this means deciding which of the attributes from the DM are to be accepted in the GM, as well as if the same attributes will be selected for the whole GM, or not; it also means deciding upon a hierarchical structure
STEP 5: *Fill-in the GM*; this means actually selecting the concepts that are to be accepted in the GM; the attributes of those concepts have been selected in step 4, but can be extended here
STEP 6: *Extend the link-structure of the GM*; this means selecting design alternatives – AND, OR [15], and setting GM attributes such as weights [15], etc.
STEP 7: *Add UM related features*; simplest way, tables, with attribute-value pairs for user-related entities (AHAM [26]), or as a concept map)
STEP 8: *Create the adaptation model* (AM); this step actually contains sub-steps, such as deciding among adaptation strategies, writing medium-level adaptation rules (as in [10]) in adaptation language or giving a complete set of low level rules [9] (such as condition-action (CA [27]) or IF-THEN rules).
STEP 9: *Define format* (PM); presentation means-related; selecting of target device (desktop, laptop, palmtop, etc.), coloring scheme and formatting, etc.

3. Automatic Authoring

Automatic authoring [13] is a type of adaptive behavior that doesn't aim at the end-user, as in traditional adaptive hypermedia, but at the designer. Designer adaptation means *adaptation* (or *adaptability)*[1] *to* the *design* (or *authoring*) *goals*. The main focus in this paper is to describe several ways of implementing designer adaptation, based on the LAOS [15] adaptive hypermedia authoring model. The designer adaptation (or adaptability) can range from adaptation to some small, recent changes, to adaptation to some generic authoring strategies, as we shall see. In this section, we explain the designer adaptation paradigm via some of the many possible automatic transformations from one layer to the other in the LAOS model (Figure 1). These transformations use the LAOS structure, as well as reappearing *design patterns* within this structure, and automatically interpret them into structures and content at the different levels of the LAOS model. This means that the author doesn't have to create the contents of these levels manually, but can rely heavily on automatic processing, or at least consider system's suggestions.
Moreover, we also look at the *flexibility index* (as defined in the Annex) for many of the automatic transformations presented.

[1] i.e., adaptation by the system's choice versus by the user's explicit selection;

3.1 Transformation within the Domain Model

The domain model (DM) can enrich itself automatically [14]. As it is basically impossible to create new content, what remains is to signal that content is missing, or to create new links. In the following, we analyze these two possibilities.

Adding missing attributes in the domain model
Link analysis can be performed to compare similar concepts[2]. In this way, the system can automatically signal that some attributes (or even sub-concepts) are missing.

Example 1: The domain concept 'Theme 2: Course Delivery System' from the 'E-learning' course has an 'Introduction to theme' domain attribute (Figure 2).
On the other hand, the domain concept 'Learning in the Biological Network' from the domain concept map 'Neural Networks II' doesn't (Figure 3), although they are linked via their 'Keywords' domain attribute with a weight of 50% (Figure 2, Relatedness relations). The system can notify the author about the missing attribute, leaving it up to the author to decide if that is an error or part of the required structure.

The *flexibility index* for this link-based concept attribute retrieval between the given domain concept C1 (current concept) and another domain concept C2 can be defined as:

$$flex(1,2) = card(\mathcal{A}_{c2} - \mathcal{A}_{c1}) \geq 0 \qquad (1)$$

For a whole domain map C, the *flexibility index* becomes:

$$flex(*,*) = \sum_{i=1}^{C} \sum_{j=i+1}^{C} card(\mathcal{A}_{cj} - \mathcal{A}_{ci}) \geq 0 \qquad (2)$$

[2] Similar from a link-point of view, such as concepts sharing the same father-concept, e.g., or concepts at the same level of the hierarchy, or concepts related with each other via some special link (of a given type), etc.

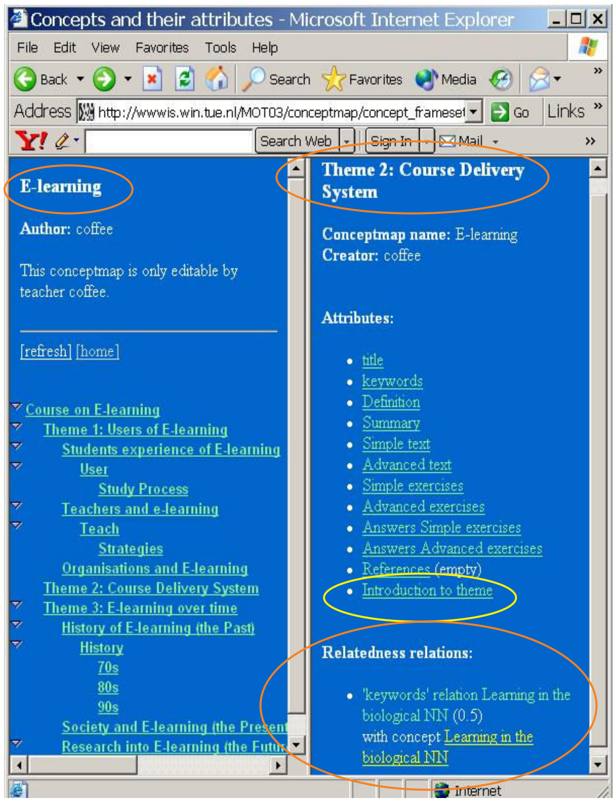


Figure 2. Example of domain model with extra domain attribute

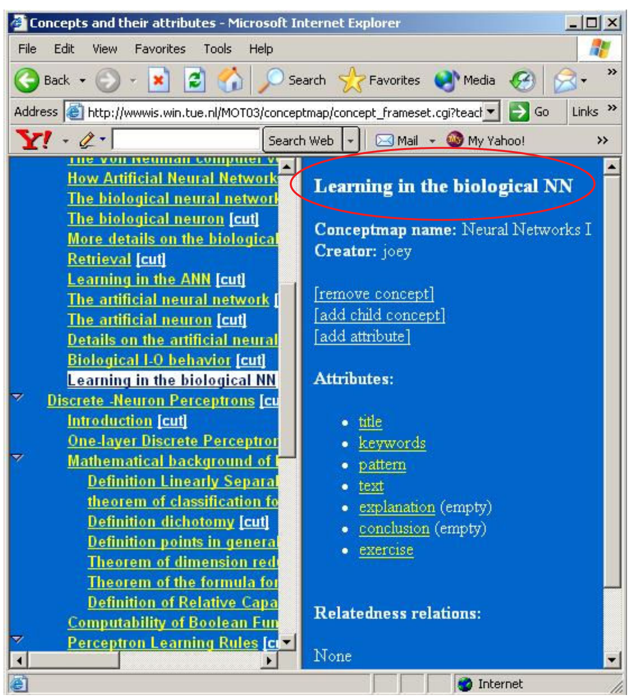


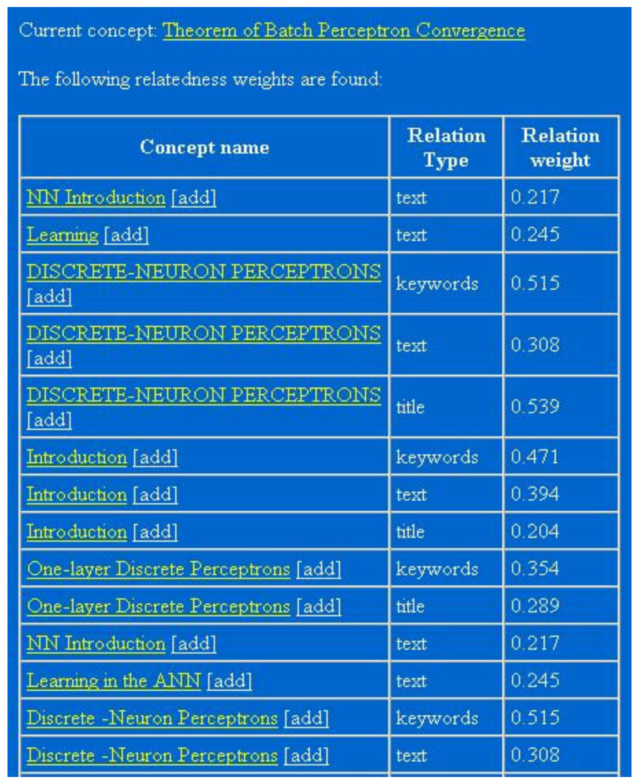
Figure 3. Example of domain model with missing domain attribute.

Creating new links in the domain model

Finding new links between existing concepts[3] automatically represents the easiest way to enrich the domain model. In [14] we developed some computation formulas for *relatedness calculations* [13] between different domain concepts with a common topic, as already mentioned in Example 1. In short, we could describe such links[4] as (Figure 4):

$$\{link(C1,C2,label,weight) \mid \exists C1.attr[1], \exists C2.attr[2], label=type(C1.attr[1])=type(C2.attr[2]), \exists weight>0, weight=number\_common\_features(C1.attr[1],C2.attr[2])\}$$

where C1, C2 are two domain concepts, and domain attribute *j* of domain concept *i* is expressed as *Ci.attr*[*j*]; *label* represents a text string that marks the link and *weight* is a numeric link label. Please note that, as we are describing a one-to-one link, the set notation from the Annex has been simplified.



Current concept: Theorem of Batch Perceptron Convergence

The following relatedness weights are found:

Concept name	Relation Type	Relation weight
NN Introduction [add]	text	0.217
Learning [add]	text	0.245
DISCRETE-NEURON PERCEPTRONS [add]	keywords	0.515
DISCRETE-NEURON PERCEPTRONS [add]	text	0.308
DISCRETE-NEURON PERCEPTRONS [add]	title	0.539
Introduction [add]	keywords	0.471
Introduction [add]	text	0.394
Introduction [add]	title	0.204
One-layer Discrete Perceptrons [add]	keywords	0.354
One-layer Discrete Perceptrons [add]	title	0.289
NN Introduction [add]	text	0.217
Learning in the ANN [add]	text	0.245
Discrete -Neuron Perceptrons [add]	keywords	0.515
Discrete -Neuron Perceptrons [add]	text	0.308

Figure 4. Implementation of DM-DM automatic relation generation.

The *flexibility index* of links that can be generated between domain concepts C1 (current domain concept)

[3] These new domain links can be between the domain concepts of the *current content* (domain map: e.g., course), between the current content and some *other content created by the same author*, or finally between the current content and some *other content created by a different author*.

[4] Another, stronger way of connecting domain concepts would look at ontologies.

and C2 is, if we consider domain links with unequivocal type[5], as follows [15]:

$$flex(1,2) = card(\mathcal{A}_{c1} \cap \mathcal{A}_{c2}) \geq card(\mathcal{A}_{\min}) = A_{\min} \quad (3)$$

where A_{ci} is the set of domain attributes of concept i and A_{min} is the minimal set of obligatory domain attributes (Annex, [15]). If we consider C=card(C) domain concepts in the domain map C, the *flexibility index* between domain concept C1 and the rest of the concepts in C, showing to how many domain concepts C1 can be linked to semantically, in the given model, is given by:

$$flex(1,*) = \sum_{j=2}^{C} card(\mathcal{A}_{c1} \cap \mathcal{A}_{cj}) \geq (C-1)A_{\min} \quad (4)$$

Generally speaking, the *flexibility index* of domain map C is given by the following relation:

$$flex(*,*) = \sum_{i=1}^{C} \sum_{j=i+1}^{C} card(\mathcal{A}_{ci} \cap \mathcal{A}_{cj}) \geq \quad (5)$$

$$\geq \sum_{i=1}^{C} \sum_{j=i+1}^{C} A_{\min} = \frac{C(C-1)}{2} A_{\min}$$

Example 2: To give a concrete example, in the MOT [*16*] *adaptive hypermedia authoring system,* A_{min} *={title, keywords, introduction, text, explanation, pattern, conclusion} so* A_{min} *=7; in the domain map called 'Neural Networks I'* C*=card(C)=145, so flex(*,*)≥10440*7= 73080 and mixflex(*,*)≥10440*49= 511560.*

These are connections implied by one concept map. MOT already allows inter-linking of concept maps, which increases this number. Therefore, it is obvious that a great number of connections can be generated automatically, making the adaptive hypermedia generation process easier.

3.2. From Domain - to Goal and Constraints Model (DM→GM)

This is one of the most natural transformations: given a book, we want to have the presentation based on it automatically. This will mean applying a filter, according to presentation constraints and goals [8]. The filter can be based on domain concept attribute or link type.

Filtering by domain attribute type from domain model to goal and constraints model
Domain attributes can be grouped into types that determine a filter for the selection of the items that will appear in the goal and constraints model. Figure 5 shows the implementation in MOT [16] of this filter.

[5] meaning that the domain attributes that determine the domain link are of the same type in both domain concepts, as stated by the domain link description above.

Example 4: E.g., for A_{min} *={title, keywords, introduction, text, explanation, pattern, conclusion}* (A_{min} *=7) as in the Annex, if we define* $A_{transf} \subseteq A_{min}$ *as* A_{transf} *={title, keywords}* (A_{transf} *=2), the transfer set from DM to GM, we can implement a goal-constraints map representing the elements for the pedagogical goal "short introduction" (e.g., for a very quick overview of the whole material).*

The *flexibility degree* that can be generated (showing the different ways of selecting domain attributes from a domain concept C1, considering that in the goal and constraints map the order of concepts also matters - as opposed to the domain map), is as follows:

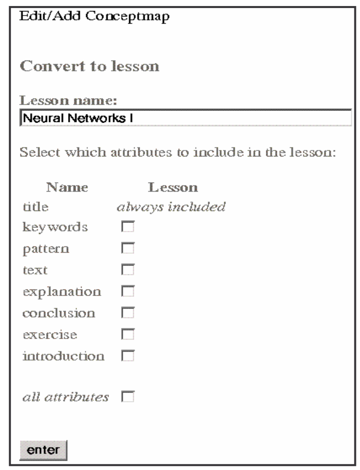


Figure 5. Implementation of DM-GM automatic concept generating transformation.

$$flex(1) = \sum_{i=1}^{card(\mathcal{A}_{c1})} P(card(\mathcal{A}_{c1}), i) \geq \sum_{i=1}^{A_{min}} P(A_{min}, i) = \quad (6)$$

$$= \sum_{i=1}^{A_{min}} \frac{A_{min}!}{(A_{min} - i)!}$$

where $P(a,b)$ are permutations of a elements taken b at a time.
So, the flexibility degree for one single domain concept and its extracted domain attributes is flex(1)≥ 13699 (for A_{min} =7), which is a relatively large number. If domain concepts are transformed independently, e.g., in special groups, this flexibility degree can grow significantly. This large number means that we can create a great number of presentations automatically, based on the structure imposed by LAOS, while keeping the result semantically significant.

Filtering by domain link type from domain model to goal and constraints model
Domain links in the domain map are defined (Annex) as either hierarchical, or of other nature. These link types can be used to generate specific links at the level of the GM model.

Example 5: If a domain concept Ca is a sub-concept of domain concept Cb in the DM and we use a similar transformation as in the previous subsection, of choosing this time the {title, text} domain attributes ($A_{transf}=2$) for filtering; then, the generated goal and constraints concepts La1=Ca.title and La2=Ca.text will be goal and constraints sub-concepts of Lb1=Cb.title, and the former domain attribute Cb.text becomes goal and constraints concept Lb2, which is also a goal and constraints sub-concept of Lb1. Therefore, the hierarchical prerequisite link structure in DM is transformed into a new hierarchical link structure for the GM[6] (Figure 6): Lb1 $\supseteq$ Lb2, La1, La2.

Furthermore, goal and constraints concepts in the GM are ordered, as opposed to domain concepts in the DM: Lb1 > Lb2 > La1 > La2 (see numbering in Figure 6 and definitions in Annex).

Figure 6. Implementation of DM-GM automatic transformation based on domain link structure.

Moreover, relations in the GM have a type; they can be hierarchical (prerequisite), as describe before, or {AND/OR}. The latter are relations between elements at the same hierarchical depth, or siblings. In MOT, all elements at a certain hierarchical depth in the domain model are automatically transformed into goal and constraints concepts connected via an 'AND' relation (Figure 6): AND(Lb1 , AND(Lb2 , La1 , La2).

In the format defined in the Annex, this is written as:

<Lb1,Group ,AND,NULL>; Group={Lb2,La1,La2};
<Lb2, La1,AND,NULL>;< La1,Lb2, AND,NULL>;
<Lb2,La2,AND,NULL,>; < La2,Lb2, AND,NULL,>;
<La1, La2,AND,NULL>;< La2,La1, AND,NULL>;

However, these AND goal and constraints links can afterwards be manually altered[7].

The illustrated link-based transformation above is simple, as it only takes into account the hierarchical link relations in the DM. However, it is useful in order to illustrate the many different types of links that can be generated for the GM even from such a simple link sub-set. The result, again, has semantics.

3.3. From Domain - to Adaptation Model (DM→AM)

The rules in the adaptation model can be also automatically generated from the domain model, according to some (goal, e.g., pedagogical) strategy. Normally, the AM is supposed to work only with the GM data that is already pre-selected for presentation. This means that a teacher, for instance, only uses material pre-selected for presentation for the students, and doesn't directly read from the book that is the basis of this presentation.

The rules determine what and how is shown to the user. Rules can be generated from a template, applying the same rule for different concept attributes, or on the link structure.

DM→AM transformation based on domain attribute type

Domain attribute types can be used to show only specific domain attributes in specific conditions.

Example 6: A specific automatic adaptive rule can express the fact that we only want to show the 'text' attribute of domain concept C1 after 'title' and 'introduction' were read:

IF(C1.title.access='true'
AND C1.introduction.access='true')
THEN C1.text.available='true';

In order for this to be a generic automatic transformation rule, for any concept C in the DM, all concepts in the UM that are an overlay model of the DM should have also attribute states 'access' and 'available', and the following low-level rule has to be added to the AM:

IF(C.title.access='true'
AND C.introduction.access='true')
THEN C.text.available='true';

This only means that the rules can only be applied if the respective attributes also exist.

If generic rules as the one above are permitted, for each such transformation [27] only one rule will be added. The number of possible rules to generate is potentially infinite,

[6] which can be regarded also as a hierarchical inclusion relation.

[7] e.g., into weighted 'OR' relations, not further detailed here.

because it is dependent also on newly added states into the UM, which can be arbitrarily large. If we consider the case where *s=2*, as in the above example, and if we enforce that the 'access' state can only be found on the left side, while the 'available' state can appear on both sides of the rule, we obtain as *flexibility degree*:

$$flex(1) = \left(\sum_{i=1}^{A_{min}} C(A_{min}, i)\right)^3 = \left(\sum_{i=1}^{A_{min}} \frac{A_{min}!}{(A_{min}-i)!i!}\right)^3 \quad (7)$$

For $A_{min}=7$, $flex(1)=(87)^3= 658503$. We obtain such a huge number because the events of having 'access' states on the left, 'available' states on the left and 'available' states on the right are independent, meaning that for each state determining the attributes that appear as 'access'-ed on the left side of the IF all combinations of attributes with 'available' on the left are possible, etc.

DM→AM transformation based on Domain Link Type

Domain links between domain concepts can be interpreted into an adaptation model, so that, e.g., only some links are 'fired' by the adaptation engine. In [10] we have already used the inherent structure of the DM by defining the generalizing and specializing adaptation language commands:

GENERALIZE (COND, COND1, … , CONDn)
SPECIALIZE (COND, COND1, … , CONDn)

The first command goes up the hierarchical tree, using the father-concept connection, while the second one goes down, in a depth-first approach, using the child-concept relationship. They both turn the 'available' state of the respective concepts to 'true'.

Example 7: With such commands, it is easy to write here an automatic transformation rule from the domain model to the adaptation model, that only allows the reading of the sub-concepts after all attributes of the current concept are read:

SPECIALIZE(C.title.access='true',
C.keywords.access='true', C.introduction.access='true',
C.text.access='true', C.explanation.access='true',
C.pattern.access='true', C. conclusion.access='true');

3.4. From Goal and Constraints - to Adaptation Model (GM→AM)

The transformation GM to AM is in principle similar to the one from DM to AM in section 3.2. However, this type of transformation is more natural in the LAOS structure, as the existence of the GM model supposes a pre-selection of the material that is to be presented to the hypermedia user, according to some (pedagogical) *goal* and delimited by some (spatial, time, pedagogical, etc.) *constraints*.

GM→AM: by domain attribute type

For domain concepts that appear in the GM, the domain attribute number is reduced (A_{min}=card(A_{min})=3; where A_{min}={'name' , 'contents', 'order'}[8]). Therefore, not so many combinations of the domain attributes can be performed.

GM→AM: by domain link type

The GM, as said, contains pre-ordered information from the DM. This structure can already be interpreted in terms of adaptation. For instance, the GM allows 'AND' goal and constraints relations between concepts (also 'OR' relations with weights).

Example 8: These can be used to express that all goal and constraints concepts in an 'AND' goal and constraints relation should be read, e.g:

IF ((C.name.access='true'OR C.contents.access='true')
AND link(C,C2,'AND',NULL))
THEN {C2.name.accessible='true';
C2.contents.accessible='true';}

In such a way, various constructs can be automatically added to the generic adaptation rules, directly by interpreting the *goal and constraints model*.

3.5. From User - to Adaptation Model (UM→AM)

The user model can generate automatically adaptation rules by applying templates for specific user model attributes or links.

UM→AM: by user model attribute type

Example 9: To illustrate a pure usage of UM elements to generate an AM rule, we consider the same state of 'interest' about a user concept. We want a rule that displays everything in the user concept, if this concept is of interest to the user:

IF (C.interest > threshold)
THEN{C.name.available='true';
C.contents.available='true';}

This rule is a generic rule, which can be applied on all user concepts in a user map, therefore drastically reducing the workload.

UM→AM: by link type

User link type can only be used when the UM is itself a concept map. In this way, we can express for instance the fact that two states in the UM are related.

Example 10: Let's consider the user model link of type 'influence'. We add a rule saying that the interest in a subject C might decrease if the user is interested in another subject C2.

IF LINK(C,C2,'influence',*)

[8] Please note that weight and label attributes are not obligatory.

THEN {C.interest= C.interest – C2.interest;}

3.6. From Domain - to Presentation Model (DM→PM)

Finally, we discuss transformations into the PM.

DM→PM: by Concept Attribute Type

In MOT we had to decide on some simple display options for the AEH designer environment, for tests and overview. This is why a button 'Student view' was created, that interprets the existing domain model into a simple book structure[9]. This option displays the whole created material in a concept map, looking exactly at the domain attribute types and interpreting them into a display format (such that titles appear in bold and green, keywords appear in yellow underneath, etc).

Example 11: Here is what such a transformation function can look like:

IF C.title.access='true'
THEN{C.title.color='Green'; C.title.style='Bold';}
IF C.keywords.access='true'
THEN{C.keywords.color='Yellow';
C.keywords.style='Italics';}
IF C. text.access='true'
THEN { C.text.color='Black'; C.text.style='Normal';}
IF C. conclusion.access='true'
THEN{C.conclusion.color='Red';
C.conclusion.style='Normal';}

DM→PM: by link type

Similar to concept attribute types, the different links themselves can be made visible and displayed in different colors; however we are not going into details about this here.

4. Conclusion

In this paper we have approached the issue of improving and making AEH authoring easier [39] by enumerating a number of different types of automatic transformations that can be directly performed by the adaptive (educational) hypermedia authoring system (section 3). These possible automatic authoring techniques (transformations) are based on the data design given by the LAOS model, allowing concept-oriented approach for data design, analysis and usage. The implementation was done in MOT [16], an authoring system based on the LAOS framework, and evaluated partially previously [13].

Here we have described the transformations and their underlying design from a theoretical point of view. These transformations are based on the design patterns and on the LAOS structure.

Moreover, we have given several working examples for the automatic transformations based on the LAOS structure, pointing to the great number of different design possibilities that these automatic functions still allow and finally showing that the authoring capacity is not inhibited by the added automatic authoring functionality.

The range of possibilities of outcomes was computed in the form of a *flexibility degree*, showing also the range of the adaptivity of the final system.

Although these transformations have been discussed and analyzed separately, in practice it is reasonable to expect that these transformations can and will be in parallel. This parallelization may be leading to a situation where one transformation may set restrictions on another one, but most of the time, these multiple transformations can generate a higher flexibility degree.

It is interesting to consider, for future research, the combination of these automatic transformations and, e.g., presentation strategies bound to specific cognitive styles. We expect that applying such strategies would affect several layers at once.

Formal concept analysis [18] allows discovery of patterns between application data, on the one hand, and the usage of concepts and relations and the semantics given by their hierarchies, on the other hand, so we have to study the connection of our research with this one.

In this way, we gradually advance towards adaptive hypermedia that *'writes itself'*.

[9] as our model was based on the book structure paradigm anyhow.

7. Acknowledgements

This research is linked to the European Community Socrates Minerva project "Adaptivity and adaptability in ODL based on ICT" (101144-CP-1-2002-NL-MINERVA-MPP).

References:

1. 2L690: Hypermedia Structures and Systems, Lecturer: Prof. De Bra. http://wwwis.win.tue.nl/~debra/2L690/
2. Aroyo, L., Cristea, A.I., and Dicheva, D. A Layered Approach towards Domain Authoring Support. ICAI 2002 (Las Vegas, US) CSREA Press.
3. Brusilovsky, P. Developing adaptive educational hypermedia systems: From design models to authoring tools. In: T. Murray, S. Blessing and S. Ainsworth (eds.): Authoring Tools for Advanced Technology Learning Environment. Dordrecht: Kluwer Academic Publishers, 2003.
4. Brusilovsky, P. Adaptive hypermedia, User Modeling and User Adapted Interaction, Ten Year Anniversary Issue (Alfred Kobsa, ed.) 11 (1/2), 2002, 87-110.
5. Brusilovsky, P., Schwarz, E., Weber, G. ELM-ART: An intelligent tutoring system on world wide web. International Conference on Intelligent Tutoring Systems (ITS'96) (Montreal, Canada, June 1996), 261-269.
6. Brusilovsky, P., Eklund, J., and Schwarz, E. Web-based education for all: A tool for developing adaptive courseware. Computer Networks and ISDN Systems, Seventh

International World Wide Web Conference (14-18 April 1998) 30 (1-7), 291-300.
7. Carro, R. M., Pulido, E. Rodríguez, P. Designing Adaptive Web-based Courses with TANGOW. 7th International Conference on Computers in Education, ICCE'99 (Chiba, Japan, November 4 - 7, 1999) V. 2, 697-704.
8. Cristea, A.I., Automatic Authoring in the LAOS AHS Authoring Model, Hypertext 2003, Workshop on Adaptive Hypermedia and Adaptive Web-Based Systems.
9. Cristea, A.I., and Aroyo, L. Adaptive Authoring of Adaptive Educational Hypermedia, AH 2002, Adaptive Hypermedia and Adaptive Web-Based Systems, LNCS 2347, Springer, 122-132.
10. Cristea, A.I., and Calvi, L. The three Layers of Adaptation Granularity. UM'03. Springer.
11. Cristea, A.I., and De Bra, P. Towards Adaptable and Adaptive ODL Environments. AACE E-Learn'02 (Montreal, Canada, October 2002), 232-239.
12. Cristea, A.I., and Kinshuk. Considerations on LAOS, LAG and their Integration in MOT. ED-MEDIA'03.
13. Cristea, A., De Mooij, A. Evaluation of MOT, an AHS Authoring Tool: URD Checklist and a special evaluation class, CATE'03 (International Conference on Computers and Advanced Technology in Education) Rhodos, Greece, IASTED, ACTA Press, ISBN 0-88986-361-X, pp. 241-246.
14. Cristea, A., De Mooij, A. Designer Adaptation in Adaptive Hypermedia. ITCC'03 (Las Vegas, US 28-30 April 2003) IEEE Computer Society.
15. Cristea, A., De Mooij, A. LAOS: Layered WWW AHS Authoring Model and its corresponding Algebraic Operators. WWW'03, Alternate Education track. (Budapest, Hungary 20-24 May 2003). ACM.
16. Cristea, A., De Mooij, A. Adaptive Course Authoring: MOT, My Online Teacher. ICT-2003, IEEE LTTF International Conference on Telecommunications, "Telecommunications + Education" Workshop (Feb 23 - March 1, 2003 Tahiti Island in Papetee - French Polynesia).
17. De Bra, P. and Calvi, L. AHA! An open Adaptive Hypermedia Architecture. The New Review of Hypermedia and Multimedia, vol. 4, Taylor Graham Publishers, 1998,115-139.
18. Ganter, B., Wille, R. Formal Concept Analysis. Mathematical Foundations, 1999, Springer, ISBN: 3-540-62771-5.
19. IEEE LTTF, Learning Technology Task Force. http://lttf.ieee.org/
20. IMS (Instructional Management System) http://www.imsproject.org
21. LOM standard, http://ltsc.ieee.org/wg12/
22. Mizoguchi, R., Bourdeau, J. Using Ontological Engineering to Overcome Common AI-ED Problems, International Journal of AI in Education, 11 (2), 107-121.
23. SCORM standard, http://www.adlnet.org/index.cfm?fuseaction=scormabt
24. WC3, Semantic Web. http://www.w3.org/2001/sw/
25. W3C, SMIL, Synchronized Multimedia Language. http://www.w3.org/AudioVideo/
26. Wu, H., De Kort, E., De Bra, P. Design Issues for General-Purpose Adaptive Hypermedia Systems. ACM Conference on Hypertext and Hypermedia (Aarhus, Denmark, August 2001) 141-150.
27. Wu, H. A Reference Architecture for Adaptive Hypermedia Applications, doctoral thesis, Eindhoven University of Technology, The Netherlands, ISBN 90-386-0572-2.

Appendix: Definitions

Table 1. Generic Definitions.

Definition 1. Let **CM** be the set of all **AHS concept maps**, **C** the set of all **concepts** and **A** the set of all **attributes**.

Table 2. Domain Model Definitions.

Definition 2. Let the domain model **DM** be formed by the **set of all domain maps** (**DM** $\subseteq$**CM**), containing all information (resources and links between them) of the AHS relevant to the domain of the resources.
Definition 3. A **domain map** DM of the AHS is determined by the tuple <*C*, *L*, *Att*>; where *C* a set of concepts; *L* a set of links and *Att* a set of DM attributes[10] (DM $\in$**DM**).
Definition 4. A **domain concept** $c\in DM_i.C$ is defined by the tuple <*A*, *C*>; where $A\neq\emptyset$ is a set of DM attributes; *C* a set of DM sub-concepts; DM_i the domain map instance the concept belongs to.
Definition 5. A **domain link** $l\in L$ is a tuple <*S*, *E*, *N*, *W*> with $S,E \subseteq\{DM_i.c_k\}_{i,k}$ $(S, E\neq\emptyset)$ *start* and *end* sets of DM concept instances[11], respectively; *N* set of labels of the links; *W* set of weights of the links.
Definition 6. A **domain attribute** $a\in DM_i.C.A$ is a tuple <*type*, *val*>, where *type* is the name of the DM attribute; *val* is the value (contents) of the DM attribute.
Constraint 1. Each domain concept is required to have a **minimal set of (standard) attributes**[12]**,** A_{min} $(A\supseteq A_{min}\neq\emptyset)$.
Constraint 2. Each domain concept *c* must be involved in at least one special link *l*, called **hierarchical link** (link to ancestor concept). Exception: root concept.

Table 3. Goal and Constraints Model Definitions.

Definition 6. Let the goal and constraints model **GM** be formed by the **set of all goal and constraints maps** (**GM** $\subseteq$**CM**), containing all information (resources and links between them) of the AHS relevant to presentation purposes (goals).
Definition 7. A **goal and constraints map** GM of the AHS is a tuple <*G*, *GL*, *GAtt*>; *G* represents a set of GM concepts; *GL* a set of GM links and *GAtt* is a set of GM attributes (GM $\in$**GM**).
Definition 8. A **goal and constraints concept** $g\in GM_i.G$ is defined by the tuple <*GA*, *G*, $DM_j.c.a$> $GA\neq\emptyset$ is a set of attributes; *G* a set of sub-concepts; $DM_j.c\in C$ is the ancestor DM concept[13] and $DM_j.c.a\in A$ is an attribute of that concept; GM_i is the name of the GM map instance to whom it belongs.
Definition 9. A **goal and constraints link** $gl\in L$ is a tuple <*S*, *E*, *N*, *W*> with $S,E \subseteq\{DM_i.c_k\}_{i,k}$ $(S, E\neq\emptyset)$ *start* and *end* sets of GM concept instances, respectively; *N* set of labels of the links; *W* set of weights of the links.
Definition 10. A **goal and constraints attribute** $ga\in DM_i.C.A$ is a tuple <*type*, *val*>, where *type* is the name of the GM attribute; *val* is the value (contents) of the GM attribute.
Constraint 3. Each goal and constraints concept is required to have a **minimal set of (standard) attributes,** GA_{min} $(GA\supseteq GA_{min}\neq\emptyset)$[14].
Constraint 4. Each goal and constraints concept *g* must be involved in at least one special link *gl*, called **prerequisite link** (link to ancestor concept)[15]. Exception: root concept.

---

[10] Note that these are attributes at the level of the domain map, describing it directly, and not the concepts in it.
[11] c_k is a concept instance in an arbitrary domain DM_i. Please note that the generic definition allows loop links between a concept and itself. In praxis, links can be added between any concepts of the owned domain maps to any concepts of the whole space of domain concept maps.
[12] To specify what we REALLY want the authors to fill in.
[13] Can be void.
[14] One such standard attribute is the order number.
[15] GM concepts are also expected to participate in one of the special links called **AND/OR link** (link to sibling concepts), but as there is no constraint requiring the number of siblings to be above zero, this cannot

Constraint 5. Each goal and constraints concept *g* must have at least one special, numerical, goal and constraints attribute *ga*, called **order attribute**. This attribute reflects the order of the concept among siblings with the same prerequisite goal and constraints concept. Exception: non-existence of siblings.[16]

Table 4. User Model Definitions.

Definition 11. Let the user model **UM** be formed by the set of all **user maps** (**UM** $\subseteq$**CM**), containing all information (resources and links between them) of the users.
Definition 12. A **user map** UM of the AHS is a tuple <*U*, *UL,UAtt*>; *U* a set of UM concepts; *UL* a set of UM links and *UAtt* is a set of UM attributes (UM $\in$ **UM**).
Definition 13. A **user concept** $u\in UM_i.U$ is defined by the tuple $<AU, U, GM_i.g/DM_i.c>$; $AU\neq\emptyset$ is a set of UM attributes; $UM_i.U$ a set of UM sub-concepts; $GM_i.g/DM_i.c\in G/C$ is the ancestor GM (or DM) concept.
Constraint 6. Each user concept is required to have a **minimal set of (standard) attributes,** UA_{min} ($UA\supseteq UA_{min}\neq\emptyset$)[17].

Table 5. Presentation Model Definitions.

Definition 14. Let the presentation model **PM** be formed by the set of all **domain maps** (**PM** $\subseteq$**CM**), containing all information (resources and links between them) of machines on which the presentation is performed.
Definition 15. A **presentation map** PM of the AHS is a tuple <*P*, *PL, PAtt*>; *P* represents a set of PM concepts; *PL* a set of PM links and *PAtt* is a set of PM attributes (PM $\in$ **PM**).
Definition 16. A **presentation concept** $p\in PM_i.P$ is defined by the tuple $<PA, P, GM_i.g/DM_i.c>$; $PA\neq\emptyset$ is a set of PM attributes; $PM_i.P$ a set of PM sub-concepts; $GM_i.g/DM_i.c \in G/c$ is the ancestor GM (or DM) concept.
Constraint 7. Each presentation concept is required to have a **minimal set of (standard) attributes,** PA_{min} ($PA\supseteq PA_{min}\neq\emptyset$)[18].

Table 6. Flexibility Index Definition.

Definition 17. The **flexibility index** is the combinatorial index computing the number of different outcomes that can be generated by a given transformation.
Definition 18. The **flexibility index for link-based concept attribute retrieval** from the link properties between the given concept *c1* (current concept) and some other concept *c2* can be defined as the number of potentially missing items (attributes).

be mentioned as a constraint. If two concepts g1, g2 are linked via a prerequisite link, we can simply say that g1 > g2.

[16] Note: if g1.ga > g2.ga we can simply say that g1 > g2.

[17] Containing for instance the knowledge attribute.

[18] Containing for instance the access attribute.

Biography:

Alexandra I. Cristea received her IS Dr. title and worked at the University of Electro-Communications, Tokyo, Japan. She is presently Assistant Professor at the IS Group, Faculty of Mathematics & Computer Science, Eindhoven, University of Technology, The Netherlands. Her research interests include AEH and adaptive hypermedia authoring, UM, Semantic Web, AI, Neural Networks, Adaptive Systems, Concept Mapping, ITS, Web-based Educational Environments. She authored and co-authored over 90 research papers and course booklets. She is a member of IEEE, was program committee member of Hypertext, AH, ICCE, ICAI, IKE (a.o.) and was reviewer or session chair for many conferences; she is executive peer reviewer of the ET&S Journal.