Design of the CAM model and authoring tool

Maurice Hendrix[1] , Alexandra I. Cristea[1]

[1] Department of Computer Science, The University of Warwick,
Coventry CV4 7AL, UK
{maurice, acristea}@dcs.warwick.ac.uk

Abstract. Students benefit from personalised attention; however, often teachers are unable to provide this. An Adaptive Hypermedia (AH) system can offer a richer learning experience in an educational environment, by giving personalised attention to students. On-line courses are becoming increasingly popular by means of Learning Management Systems (LSM). The aim of the GRAPPLE project is to integrate an AH with major LMS, to provide an environment that delivers personalised courses in a LMS interface. However, designing an AH is a much more complex and time-consuming task, than creating a course in a LMS. Several models and systems were developed previously, but the (re)-usability by educational authors of the adaptation remains limited. To simplify adaptive behaviour authoring for an educational author, a visual environment was selected as being most intuitive. This paper describes a reference model for authoring in a visual way and introduces an authoring tool based upon this model.

Keywords: CAM, MOT, AHA!, Adaptive Hypermedia, Authoring, Adaptation.

1 Introduction

In education students generally benefit from personalised attention by their teachers. Often teachers are however unable to provide much, personalised, attention to each student however, due to time constraints, or other constraints. At the same time, on-line courses are becoming increasingly popular and the use of so called Learning Management Systems (LMS) is becoming more widespread. Adaptive Hypermedia (AH) is perceived to have the potential to offer a richer learning experience, personalised for each learner. To realise this potential however, education authors need to have the ability to easily create adaptive material. The GRAPPLE[1] project aims to integrate an AH with major Learning Management Systems and to provide an environment that delivers personalised courses in a LMS interface.

However, designing an AH is a much more complex and time-consuming task, than creating a course in a LMS. There exist several Adaptive Hypermedia reference models, like AHAM (Adaptive Hypermedia Application Model) [13] and LAOS (Layered WWW AH Authoring Model and their corresponding Algebraic Operators) [4] that are specifically developed for authoring (instead of general purpose models of

[1] http://www.grapple-project.org/

AH, such as Dexter [6], XAHM [2], Munich [9], GAHM [10]). However, even when using tools developed based upon these models, authoring remains a difficult and time consuming task. A possible solution is to use graphical tools. Existing graphical authoring tools (e.g. the Graph Author developed for AHA! [1]) use concrete connections between concepts, and, the adaptivity is specified in a single layer. This approach means that the re-usability of the adaptivity is limited.

To simplify adaptive behaviour authoring for an educational author, a visual environment is considered most intuitive. We have previously introduced the CAM model (Conceptual Adaptation Model) [7], a generic model for authoring that allows adaptation to be defined in a visual, graphical, flexible and user friendly way. In this paper we refine this model and introduce the *design of the adaptation languages and authoring tool*, which allows the creation of a CAM instance in a visual manner.

The remainder of this paper is organized as follows. Section 2 will briefly introduce the CAM model. In section 3 we outline the design for adaptation languages and the authoring tool, and finally we conclude in section 4.

2 The CAM Model

The CAM model contains an extensible number of layers, which may be different for each application. This is due to the fact that, different aspects of the adaptation strategy might need to be stored independently. Thus the author can define new application-dependent layers. However the DM, UM and AM layers are mandatory, as they are required for every adaptive course to function.

To summarise the CAM model consists of the following layers:

- *domain model (DM)*, similar to AHAM and LAOS. The DM describes domain concepts, their attributes and their semantic relationships. These relationships do not attach any behaviour but indicate a semantic link and only domain-specific information is allowed, unlike in AHAM;
- *user model (UM)* similar to the one in AHAM and LAOS.
- a number (at least 1) of *adaptation models (AM)*, extending the ideas of generality and specificity: generic adaptation rules by means of CRTs: can be applied to specific concepts in the adaptation model. Each type of relationship provides a different layer, or view on the course. In the AM, concepts from the DM can be selected and the adaptive behaviour can be attached to these concepts.

The CAM can be exported to an adaptation engine. This engine can be the GRAPPLE adaptation engine or any other engine that can work with the system independent, intermediary Grapple Adaptation Language (GAL) [11] language.

3 Design of the adaptation languages and authoring tool

To describe adaptive behaviour in the scope of the CAM, a language format is needed to describe each of the three main components, the DM, CRT and the AM. The languages will have to deal both with the visual as well as the semantic aspects of the model. In this section we will describe each of the adaptation languages. The languages share a common header, in order to simplify looking up the information that they all share. The header stores a *unique identifier*, the *type of model*, the *id* of the *author*, the editing *authorisation* that other authors have, the date and time the model was *created* and last *updated*, the *title* and *description*.

3.1 DM language

The DM language is a language that describes the concepts and relationships between concepts. It is based on the IMS VDEX format [8]. It also specifies physical resources, the actual content. The DM format specifies which resources are linked with which concepts. The DM-tool uses a visual representation of the DM XML format, the VDEX based format, in order for the author to be able to create and edit Domain Models. The author is not meant to deal with the XML format directly. The DM format is included in the CAM format in the *domainModel* tag.

3.2 CRT language

The CRT language is not meant for human consumption, but is a description language of the adaptive behaviour, which is encapsulated and represented by a graphical symbol in the CAM environment; the CRT language is included in the *crtModel* tag of the CAM language. The language used for the CRT expressions is the GAL language [11]. For the scope of the AM-tool and language, the number and type of entities that connect to a CRT are important, as well as any restriction on combinations. Moreover, the metadata describing the CRT behaviour in layman's terms is vital for the non-programmer author, to be able use CRTs efficiently in the AM-tool.

3.3 CAM language

The CAM language is the main portable format for the output of the AM-tool. It is designed to be used by other systems, whilst they interface with the AM-tool. Prior experience with the LAG language [3] shows that non-programmer authors prefer not to be involved at this level. Prior research [12] showed that an XML-based language is preferable, as it is more portable, and easier to process than a pure programming language. The LAG-XLS language [12], and XML version of the LAG adaptation language has been previously developed. However it was aimed at learning styles, and we wish to adopt a wider scope of adaptivity, therefore a new language has to be created. The CAM format can be used to describe, not only direct CRT relations, but also more global information & flow of a course:

- *Start and end states* can be described with a special CRT. The CRT is different only in that the GAL [11] code simply stats "*start*" or "*end*".
- *Main concepts of a course*, useful for visualising the course, can be indicated in the name/meta data of the sockets.

- *Learning goals and objectives* can be derived from the selection of start and end states and main concepts.

Below we illustrate the new CAM language via an example. We see this type of CAM output as XML descriptions of groups of concepts and *named typed* relations between them. For example, if concept A is to be seen before concept B. Below an annotated example of a CAM is shown:

```
<model><header> .. </header><body>
  <cam><camInternal>
     <domainModel>..</domainModel>
  <crtModel> .. </crtModel>
     <crt>
      <uuid>cf5de7f5...</uuid>
      <shape>diamond</shape>
      <colour>#C0C0C0</colour>
      <camSocket>
       <uuid>e9b4...</uuid>
      <socketId>cf5de7f5...
      </socketId>
       <caption>source</caption>
       <position><x>100</x><y>250</y>
       </position><size>10</size>
       <shape>rectangle</shape>
       <colour>#006633</colour>
       <entity><dmId>..</dmId>
       </entity>
      </camSocket>
      <camSocket>
        <uuid>f539...</uuid>
        <socketId>2b5-...</socketId>
        <position><x>100</x><y>400</y>
        </position><size>10</size>
        <shape>rectangle</shape>
        <colour>#006633</colour>
        <entity><dmId>...</dmId>
        </entity>
       </camSocket></crt>
    </camInternal></cam>
  </body></model>
```

The *header* as described before

The domain models() used

The crt model(s) used

A unique identifier the *uuid*

The *shape* and *colour* of the crt, , defaults will be used if the tags are omitted.

An instantiation of a socket in a crt

The *socketId*, is the identifier of the socket in the crtModel, and a unique identifier *uuid* of this socket instance.

Caption, an optional name for the socket.

Position, size, shape and colour elements. For binary crts the position is calculated and should be omitted.

At least 1 *entity* tag, containing the instantiation with concepts Either of:

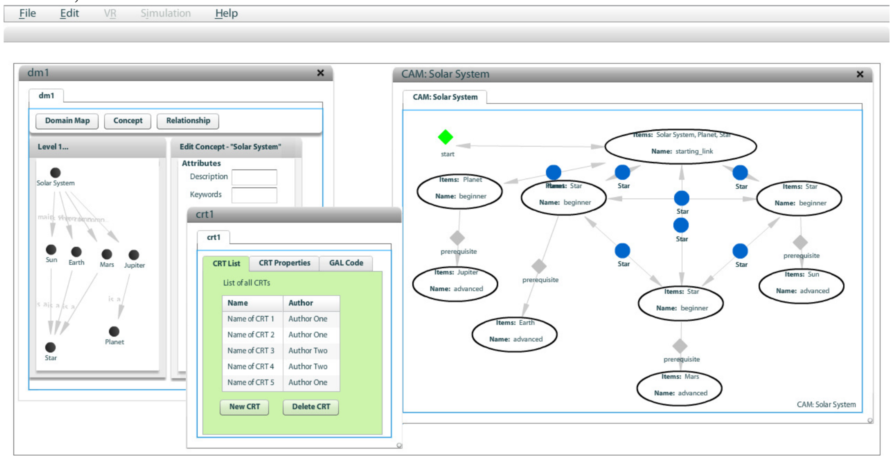
- A dmID, the ID of the concept in the DM
- A number of labels for the resource of a concept
- A location (URI) for a resource

The description of the subject domain and behaviour semantics of the CRT called 'prerequisite' needs to be separately imported from the DM and CRT repositories. These descriptions are then included in the domainModel and crtModel parts.

3.4 The authoring tool

The Authoring tool is a tool in which authors are able to specify CAM instance models (as such CAM is a meta-modelling system). The author is able to define all layers of their specific CAM model with this tool. The tool consist of three components corresponding to the mandatory layers: DM editing; CRT editing and CAM editing, held together in an integration component. The tool is implemented as a web-based application and was developed in Adobe Flex. The integration component contains general infrastructure for manipulating windows, opening/ saving models and the toolbar, which interacts with each component. Below we see a screenshot of the authoring tool. Below we see a screenshot of the Authoring Tool. The three main components are visible in floating windows, from left to right the DM-tool, CRT-tool and AM-tool. In the DM-tool we see a graph, with several nodes and links. The nodes represent the concepts in the subject domain and the links represent semantic relationships, without any attached behaviour. Authors can create conceptual Domain Models, by inserting the concepts and their links in the CRT-tool.

The CRT-tool gives the opportunity to edit the adaptive behaviour of a CRT. Authors specify a title and description of the CRT. Many authors with less technical skill will use the CRT-tool mainly to look at the descriptions of and possible User Model variables in use by a CRT. Authors can, if they wish to customise a CRT or create one from scratch, specify the number of sockets; create a list of User Model values that can be used in the code, and write the actual adaptive behaviour in GAL code [11]. The AM-tool instantiates CRTs with concepts. In the screenshot below we see the nodes, which represent sockets and the links. The links with a diamond label represent the CRTs. The round labels indicate that sockets share concepts. Authors can insert CRTs, from the list of existing CRTs, and copy concepts from a Domain Model, in a DM-tool window to instantiate the CRTs.



Name	Author
Name of CRT 1	Author One
Name of CRT 2	Author One
Name of CRT 3	Author Two
Name of CRT 4	Author Two
Name of CRT 5	Author One

Fig. 1. Authoring tool

4 Conclusions and further work

In this paper we have shown the refined model based on the previously published general scenarios for the Conceptual Adaptation Model (CAM) as well as the authoring tool. We have outlined the design for the authoring that focuses on a division of components, along the lines of the layers in the CAM model. The integration component of the authoring tool brings together these components into one single authoring system.

Moreover, we have defined adaptation languages that serve as 1) interfaces between the *human author and the machine*, as in the case of the CAM visual language, as well as 2) an interface for communication *between system parts*, as in the case of the concept relationship (CRT) language, 3) to serve as an interface for communication *between different systems*, as in the case of the CAM language.

Where possible, such as with the CRT language, we based our design upon existing proven languages (such as LAG) and extended from there.

The next steps in this process will involve finalising the development of the authoring tools, as well as the tools for translating CAM instances into usable adaptive courses. From a research perspective, *model definition* and *adaptation language development*, especially on a larger scale, such as the GRAPPLE EU project (involving 15 European partners), brings us closer to the aim of having largely adopted, reusable adaptation specifications, and ultimately, standards.

The Authoring Tool is intended to facilitate the creation of adaptive courses, *corresponding to their desired teaching strategy* and *in a supported and as easy as possible way*. Therefore the extent in which these goals are reached is addressed in a specific evaluation step. The evaluation aims to gather usability and user acceptance data and consists therefore of two parts. A standard SUS [1] usability test and a more extensive formative evaluation will be conducted.

Acknowledgement. The work presented is funded by the GRAPPLE EU FP7 project.

References

1. Brooke, J.: SUS: a “quick and dirty” usability scale. In: Jordan, P.W., Thomas, B., Weerdmeester, B.A. and McClelland, A.L. (eds.). Usability Evaluation in Industry. London: Taylor and Francis. 1996
2. Cannataro, M., Pugliese, A.: XAHM: an XML-based Adaptive Hypermedia Model and its Implementation, 3rd Workshop on Adaptive Hypertext and Hypermedia, AH2001) in conjunction with Twelfth ACM Conference on Hypertext and Hypermedia, Arhus, Denmark (2001)
3. Cristea, A.I., Calvi, L.: The three Layers of Adaptation Granularity, UM’03, International Conference on User Modelling, Pittsburgh, US, Springer, ISBN: 978-3-540-40381-4, pp. 145, DOI 10.1007/3-540-44963-9, DOI 10.1007/3-540-44963-9_4 (2003)
4. Cristea, A.I., de Mooij, A.: LAOS: Layered WWW AHS Authoring Model and their corresponding Algebraic Operators, WWW’03, The Twelfth International World Wide Web Conference, Alternate Track on Education, Budapest, Hungary (2003)
5. Garzotto F. and Cristea A.I.: ADAPT: Major design dimensions for educational adaptive hypermedia. *ED-MEDIA 2004 Conference*, AACE, Lugano, Switzerland, June (2004)
6. Halasz F., Schwartz, M.: The Dexter hypertext reference model: Hypermedia. Communications of the ACM, 37(2), 30-39 (1994)
7. Hendrix, M., De Bra, P., Pechenizkiy, M. , Smits, D., Cristea, A. I.: Defining adaptation in a generic multi layer model: CAM: The GRAPPLE Conceptual Adaptation Model, In proceedings of Third European Conference on Technology Enhanced Learning ECTEL (2008)
8. IMS VDEX Specifications, http://www.imsglobal.org/vdex/ (last accessed: 28/06/09)
9. Koch, N., Wirsing, M., The Munich Reference Model for adaptive hypermedia applications, in Adaptive Hypermedia and Adaptive Web-Based Systems, Second International Conference, AH 2002. 2002, Vol. 2347, Lecture Notes in Computer Science, Springer, pp. 213–222
10. Ohene-Djan J. A Formal Approach to Personalisable, Adaptive Hyperlink-Based Interaction. PhD thesis, Department of Computing, Goldsmiths College, University of London (2000)
11. van der Sluijs, K., Hidders, J., Leonardi, E., Houben G.J.: Generic Adaptation Language for describing Adaptive Hypermedia, Proc of International Workshop on Dynamic and Adaptive Hypertext: Generic Frameworks, Approaches and Techniques (2009)
12. Stash, N., Cristea A.I., De Bra, P.: Explicit Intelligence in Adaptive Hypermedia: Generic Adaptation Languages for Learning Preferences and Styles, Proceedings of the HT 2005 CIAH Workshop, Salzburg (2005)
13. Wu, H.: A Reference Architecture for Adaptive Hypermedia Applications, doctoral thesis, Eindhoven University of Technology, The Netherlands, ISBN 90-386-0572-2 (2002)