

ACTIVITIES WORKFLOW OF ADAPTIVE COURSEWARE DESIGN AND DELIVERY

Dessislava Vassileva, Boyan Bontchev
*Sofia University, Dep. of Software Engineering
5, J. Bourchier Blv., 1164 Sofia, BULGARIA*

ABSTRACT

Adaptive hypermedia systems (AHS) tend to be more and more appealing research topic promising elegant and effective software solutions in near future. The majority of AHS have been used for educational purposes, although some online information systems such as information retrieval systems and serious games have been made adaptive as well. In this paper we describe an innovation enhancement of the AHAM reference model for adaptive hypermedia applications and an e-learning software platform based on it. The platform consists of authoring tool, instructor tool and adaptation engine controlling the content delivery adaptable to individual learners. Navigation adaptive to learning styles is realized through link hiding and annotation plus calibration of the adaptive engine after assessment of the learner. Content selection adaptive to learner performance is achieved by means of showing only part of the content to an individual learner. For this purpose, system actors should follow specific workflows of adaptive courseware design and delivery.

KEYWORDS

AHS, adaptive, user model, learning style.

1. INTRODUCTION

Technology enhanced learning is inconceivable without system personalization and adaptation to needs and performance of individual learners. Adaptive hypermedia systems (AHS) use advanced learner models (Brusilovsky P., 1994) in order to facilitate content and behavior adaptation for a better satisfaction of specific learner expectations. There have been proposed many conceptual models of AHS (Dagger, D., Wade, V. & Conlan, O., 2005) and, based on these models, several successive AHS have been designed and used in real practice. They use various techniques to assure adaptation towards the learner model, especially to goals and preferences of individual learners. Usually, learner preferences control personalization of content presentation such as layout and stylistics, information control, and usage of delivery methods/channels. Learner goals are used for adaptive content selection, where additional courseware content is shown to given learners when they have chosen specific learning goal. Thus, besides the mandatory content, there may be some optional content covering one or several additional learning goals.

Other adaptation techniques used by contemporary AHS, are adaptive navigation through link hiding and link annotation. Some of the approaches realize navigation adaptive to learner styles. Many of the existing models are fixed to one of the existing learning styles models - such as that of Honey and Mumford, Gregoric-Mind, Dunn and Dunn, etc. (Guin-Duclosson, N. et al., 2002) - and can be used for this specific style model. Therefore, the model of learning style has to be detached as another learner sub-model allowing AHS to provide adaptation according various families of learning styles. The ADOPTA project (ADaptive technOlogy-enhanced PlatForm for edutAinment) being currently under development at Sofia University, Bulgaria, makes use of an advanced model of learner, content and adaptation. The rest of the paper provides our motivation to construct and elaborate the ADOPTA model, its comparison to other similar models of AHS, and a detailed description of activities workflow of adaptive courseware design and delivery. Workflows of activities are very important for all the ADOPTA stakeholders including content authors, course instructors, supervisors of the engine controlling adaptation processed and, finally, the learners. Special attention is paid to activities of authors, who have to design learning ontology graphs organizing multimedia learning objects (LOs), and metadata for both the ontology and LOs. The result of the authors'

work is used by instructors in order to build narrative storyboard graphs with paths and content pages suitable for individual learner needs. Finally, the workflow of the adaptation engine provides selection of best paths and most suitable learning content for every individual learner.

2. PRINCIPAL MODEL AND GENERAL ISSUES

In this chapter, there will be described an innovative conceptual model of AHS with its specific characters and advantages over other similar proposals serving as a motivation for its creation and elaboration.

2.1 Motivation

Contemporary AHS are entirely oriented to individual user's needs, preferences or knowledge (Ohene-Djan, J. et al, 2003). They follow some of existing models dealing with learner, domain and adaptation concepts:

- Dexter Hypertext Reference Model (De Bra P. et al, 1999) – the goal of the model is to provide a systematic basis for comparing systems and to develop interchange and interoperability standards. The Dexter model divides a hypertext system into three layers. The storage layer describes the network of nodes and links that is the essence of hypertext. The runtime layer describes mechanisms supporting the user's interaction with the hypertext. The within-component layer covers the content and structures within hypertext nodes. The focus of the model is on the storage layer as well as on the mechanisms of anchoring and presentation specification.
- AHAM reference model (Halasz, F., Schwartz, M., 1994) - augments the Dexter Hypertext Reference Model with features for doing adaptation based on a user model which persists beyond the duration of a session. The goal of this model is to describe most features that are found in AHS.
- Munich Reference Model (Koch, N. & Wirsing, M., 2002) – it is similar to AHAM and it is an extension of the Dexter Hypertext Reference Model by including a user model and an adaptation model in the Storage Layer as well as adapting the domain model and the Run-Time Layer. The main difference between The Munich Reference Model and AHAM is that AHAM specifies the adaptation rule language, while The Munich Reference Model uses object-oriented specification.
- Goldsmiths Adaptive Hypermedia Model (GAHM) (Ohene-Djan, J. et al, 2003) – it provides a formal approach to the modeling of personalisable, adaptive hyperlink-based systems. The model consists of three groups of functions - H-Region, P-Region, A-Region. The H-Region functions model non-personalisable hypermedia based interaction, the P-Region functions model user-initiated tailoring and the A-Region functions model system-initiated tailoring of hypermedia content.

The main shortcomings of above models consist in their support of adaptability only according particular learner's knowledge. They do not support adaptability according learner's goals and learning styles, which is established in last years as one of the most effective conception in modern AHS. More, the existing models of AHS put specific pedagogical strategy in the learning content and this hampers content interoperability. As well, it impedes using of the same pedagogical strategy for different learning content.

We will fill these gaps by proposing a new AHS model, which will aim at solving the problems defined above. Therefore, our model will define explicitly differentiation between:

- learning content
- adaptive rules and pedagogical strategy
- learner profile

More, user model will be extended with an enhanced support of learning styles and learner's goals/preferences, with intention to use learning styles for adaptation of the content and goals/preferences - for content personalization. These extensions will allow a greater reusability and interoperability of learning content and a flexible and effective construction of an AHS following this model.

2.2 A principal model of AHS

The main aim of our AHS model is to be most flexible and to allow effectiveness and easy expandability in terms of adaptive content management and support. For this purpose, we propose a model which refines the

established and widely used model - the AHAM reference model (Halasz, F., Schwartz, M., 1994). Our model is divided into three sub-models, strongly independent one from another. This independence allows each one of the sub-models to be easily changed, without this to affect the others. This hierarchical model consists of two levels. At first level, the model is based on a precise separation between Learner, Domain and Adaptation sub-model, while at second level each of these sub-models is divided into three others sub-models (Vassileva D. & Bontchev B., 2006). Our model follows a metadata driven approach, explicitly separating on one side the narrative storyboard from narrative metadata and rules and, on other side, the content and its organizing ontology from metadata of both content and ontology. It allows easily to add new requirements and to improve the adaptation engine without necessity to change the AHS model. This is a new hierarchical organizational model for building adaptive hypermedia learning content management system (LCMS).

In table 1, it is presented the structure of our model together with explanation of the most important characteristics for each sub-model at the first level as well as their sub-models at the second level. As shown in the table, in the Learner model we separate goals and preferences from shown knowledge and performance, which misses in other similar models and allows to adapt content according learner's knowledge and performance and to personalize it according learner's goals and preferences. Other difference between our model and similar ones is that we add a new sub-model – the learning style. In this sub-model, for each learner are defined her/his learning style, such as activist, theorist, reflector, pragmatist. This learning style can be polymorphic, which means that it is presented by order quadruple, for instance (activist_value=10, theorist_value=20, reflector_value=70, pragmatist_value=10) since usually a particular learner is not fixed to a concrete style but rather to several ones at different level.

Table 1. Tabular presentation of the structure of the conceptual model

Conceptual Model			
(The proposed model has a hierarchal structure of two levels. At first level it is divided into three independent from one another sub-models – learner model, adaptation model and domain model. At second level each of these sub-model again is divided into three others sub-models)			
Adaptation Engine	Learner Model	It contains information for the learner profile. Depending on its meaning, it is stored in <i>Goals and Preferences</i> , <i>Learning Style</i> or <i>Knowledge and Performance</i> sub-models.	Goals and Preferences
			Learning Style
			Knowledge and Performance
	Adaptation Model	It includes description of each course storyboard graph (in <i>Narrative Storyboard</i> sub-model), metadata (such as link annotations, exam thresholds, etc.) of each narrative storyboard graph (in <i>Narrative Metadata</i> sub-model) and selection logic for passing over particular graph (in <i>Storyboard Rules</i> sub-model).	Narrative Metadata
			Narrative Storyboard
			Storyboard Rules
	Domain Model	It is responsible for structuring of learning content. The content is granulized in LOs, which for theirs part are connected among themselves in relevant knowledge domain ontology. LOs and ontology are described by their metadata (in <i>Content Metadata</i> sub-model) respectively according IEEE LOM specification and Ontology Metadata Vocabulary OMV standard	Ontology graph
			Learning objects
			Content Metadata
The Adaptation Engine communicates with each of the three sub-models at first level in order to generate and delivery to particular learner the most appropriate learning content for her/him.			

The domain model is composed of content itself (granulized in LOs), LO's and ontologies' metadata and semantic ontologies organizing the content (LOs). There are supported various types of LOs – not only narrative content but also tasks and test questions.

The adaptation model (AM) captures the semantics of the pedagogical strategy employed by a course. It includes support of course storyboard graphs. This graph consists of narrative pages (with learning content compound of LOs), control points (CP) and so called work paths (WP) between them (CPs). The instructor

may define a weight of a WP for each learning style. Therefore a particular working path (WP) may be suitable for one or several learning styles. The control points are used for assessment of current knowledge and performance for a learner, by test generation. This test is composed of questions corresponding to the LOs in the pages, which the learner is visited. The obtained assessment result is used for update of WP weights.

2.3 Practical benefits of model usage

The main benefits of the proposed models in practical point view are following:

- the model provides strong independence of any of the building models and facilitates a flexible adaptation;
- the model can be supported by different system architectures with no limits of application of any adaptation techniques, such as adaptive presentation, navigation support and content selection;
- the learner model is not fixed to a concrete learning style;
- the adaptive content may vary not only depending on learner performance but on learning styles and learner goals and preferences.

3. ACTIVITIES WORKFLOW OF ADAPTIVE COURSEWARE DESIGN AND DELIVERY

3.1 General workflow

Fig. 1 represents a general view of the process of mastering and delivery of adaptive courseware including modules of the ADOPTA e-learning platform. They include an authoring tool, an instructor tool, an adaptive engine and a set of administration tools, all communication through a common repository. There are shown five working roles:

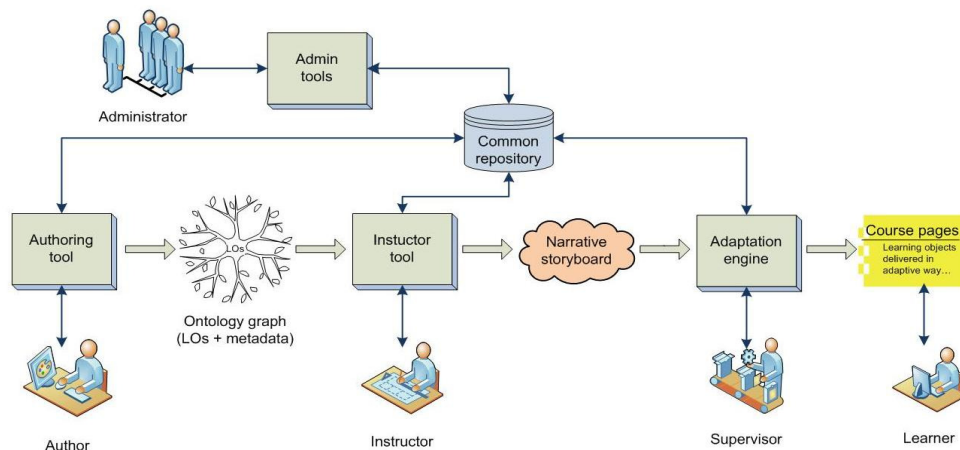


Figure 1. View of the general process workflow of design and delivery of adaptive courseware

- Author – responsible for design of learning materials (objects) by organizing them within any ontology. Each ontology is represented graphically by an ontology graph (instead tree), which facilitates representation of multiple inheritance – whereas in a tree representation we have to copy such a subclass after all its superclasses within the tree, in a graph the subclass node just has several inheritance links. As well, ontology graph make easy presentation of *has-a* relations, which is not directly possible when using ontology trees. Besides designing ontology graph and included learning objects, the author has to describe metadata about LOs and about ontology itself as

explained in 5.2. The ontology, LOs, and metadata about ontology and each LO are preserved in a central repository. Moreover, ontology, LOs, and metadata may be exported in appropriate formats (Vassileva, D., et al, 2008).

- **Instructor** – uses the instructor tool to design a course as a narrative storyboard. The instructor defines course pages and links between them. For a content page, he/she has to drag-and-drop at the page one or more learning objects from a proper ontology defined by an author. For each LO, the instructor defines coefficient of knowledge level, which will be used for adaptive content selection depending on learner performance shown at the last control page. Both the instructor and the learner are able to view the content page shown by a XSL transformation. For control pages, the content will be automatically generated by a selection of question learning objects referred by objects placed at pages visited by the learner. However, here the instructor has to specify a performance threshold used the adaptation engine for deciding to let him/her to go ahead or to be returned back to the previous control point. Moreover, the instructor may set annotations for the outgoing links to other pages which are used by the adaptation engine for adaptive navigation.
- **Supervisor** – responsible for controlling the adaptation engine. The supervisor may start and stop adaptation behavior for a given student group, may track the path of every learner, etc. Usually, the supervisor role is played by an instructor.
- **Learner** – follows a course by receiving adaptive content and solving tests at control points. The learner is supposed to start at the first control point by filling an initial test about determining his/her learning style. Next, he/she follows the work path proposed by the adaptation engine but may opt to links to pages not belonging to the path and, thus, to divert to another work path. In such a case, they are always able to return back to the last visited page of the proposed path or, otherwise, to follow the new path until reaching a control point. There, the learner has to solve a test compiled by automatically selected questions about the LOs he/she has passed through. Depending on the result, the learner goes ahead through a newly selected work path, or is returned back to the previous CP.
- **Administrator** – controls all the users by means of administrative tools. For example, the administrator approves users after registration, manages curriculums, sets internal parameters, etc.

3.2 Authoring learning courseware

Authoring process supposes definition of LOs and their semantic organization for a facile viewing and searching. The ontology may be presented not only by a tree but also by a graph, which adds more semantic power and facilitates presentation of multiple inheritance and references from one LO to another. There are two possible types of relations supported between the nodes (i.e., between the learning objects involved within the definition): inheritance relations (called also *is-a*) and references with semantic *concerns* or *has-a*. While node relations of type *is-a* are defined directly within the ontology graph, reference relations (usually known as *has-a* relations) are defined while designing the learning object. This constraint is especially dedicated to force the author to allocate a hyperlink to the referenced object within the content of the referring object.

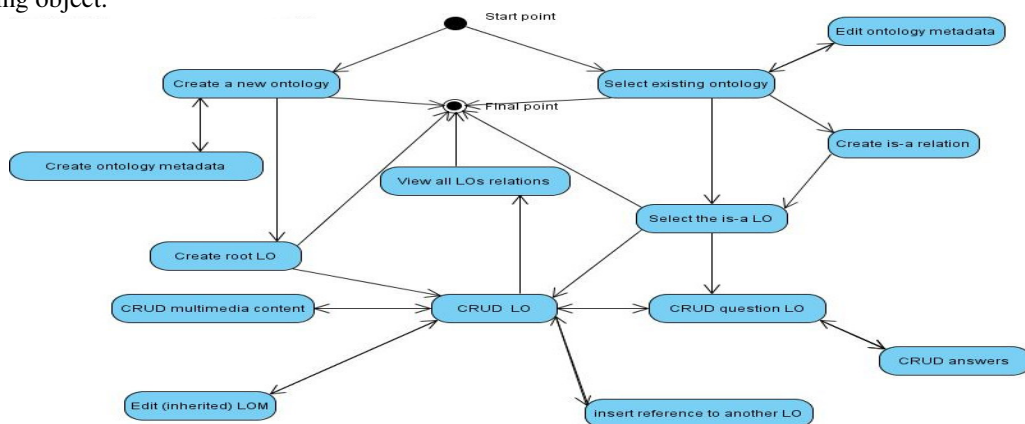


Figure 2. Activity workflow of the authoring process

Activities workflow of the authoring process is presented in fig. 2. The author creates/edits ontology and its LOs of conceptual and question types (LOs- questions are linked to one or more conceptual LOs by *has-a* relations). Besides creation/read/edit/delete operations (presented as CRUD in the figure), the author is supposed to enter metadata records. This process has two steps: 1) entering ontology metadata by means of emerging Ontology Metadata Vocabulary (OMV), which should be done only once for given ontology; 2) describing metadata for LOs according IEEE Learning Object Metadata (LOM) standard – here, the author may use a mechanism of metadata inheritance from the root LO for the ontology to its successors. If metadata records of each LO are identical, then the author has to specify them only for the root LO. Otherwise, the author has to describe only the differences between metadata records of ancestor LO and its successor (if there are any).

LOs may be of various types such as conceptual issue, project task, essay, etc. For anyone of these LOs, the author may design one or several assessment LOs. An assessment object is a Question and Test Interoperability (QTI) question (Petrov M., B. Bontchev, 2004) with several answers of type one-of-many or many-of-many. For each answer, the instructor defines a result value. Questions may be only referred by other LOs of type not being question, and cannot refer to other LOs. Moreover, question LOs are not shown at course pages but are used by the adaptation engine to build an assessment tests at next CP.

3.3 Instructional design

As explained in 4.1, the instructor tool is a module of ADOPTA platform for creating courses adaptable to different users. The instructor composes a course in terms of interconnected pages represented as nodes of the narrative storyboard. The narrative storyboard graph is to be processed by the adaptation engine (AE) in order to choose the best path for a particular user. Pages are easily modified by drag and drop of available learning objects. Within the course graph, there may be several CPs representing assessment of learners. At a CP, the test generated and assessed automatically by using randomly selected questions.

Fig. 3 shows the activity workflow for the instructional course design. The instructor may start creation of a new course or may continue with updates of already existing course. At any time of the work, he/she may choose a new ontology to be used for selecting LOs and placing them via drag-and-drop at a narrative course page. Moreover, the instructor has also the responsibility to annotate page links (for adaptive content selection) and to set performance parameters for each LO. Performance parameters show how much given LO is suitable to be shown to a learner with given knowledge level, i.e. assessment result at the last control point. These assessment parameters are very important for tuning the system as far as the adaptation engine uses them to decide whether given page would be useful for particular user or not.

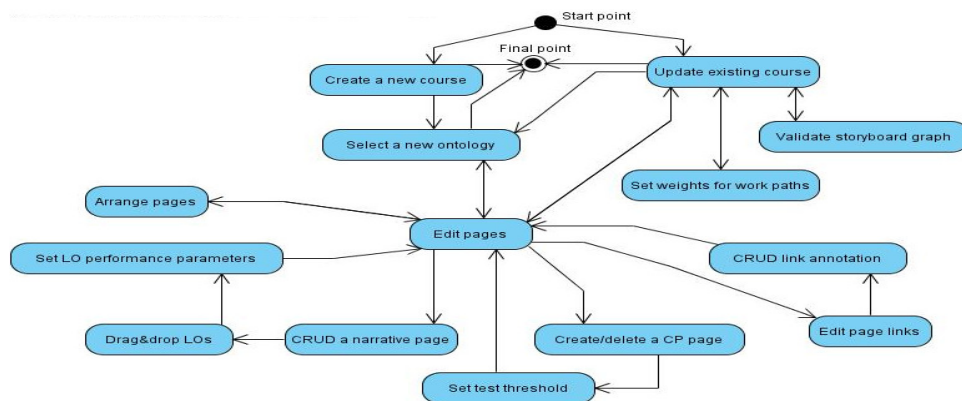


Figure 3. Activity workflow of the instructional process

Within the ontology graph, these questions should be linked with *has-a* relationship to some of the learning objects shown in pages on the work path leading to that CP. The instructor has to adjust CP thresholds values, i.e. to state the minimal value of assessment results for passing the test. The assessment parameters given by the instructor for any of the LOs are very important for tuning the system as far as the adaptation engine uses them to decide whether given page would be useful for particular user or not.

3.4 Adaptation control

The adaptation engine is responsible for performing all necessary adaptation mechanism for content delivery to a specific learner. As explained in previous chapter, this includes content selection, content hiding, link annotation, link hiding, etc. Fig. 4 represents a simplified activity diagram of the AE. When learner starts a new course, he/she has to pass through a CP with assessment of his/her learning style. Based on this assessment, the adaptive engine finds the best path for a given learner at any of the CPs within the course storyboard graph. The best path is that one with the highest weighed score and is calculated for a particular user as a sum of multiplications between weights (for each existing learning style) of working paths starting at the CP, and weights of their correspondent learner's styles. This path is stored for learner as current work path and is shown to the learner by a link with appealing link annotation, while other paths are shown with worse annotations or are hidden. When learner asks for the next page, adaptive engine may hide objects that are not appropriate for this user. At any page with some links leading away of the selected work path, the learner is free to press such a link and to abandon the path. In such a case, the learner may choose to navigate to pages through the new path or to return back to the selected path. If he/she navigates through the new path until reaching a new CP, then the AE stores the pages of the new path and selects questions related to the shown LOs placed by the instructor at these pages.

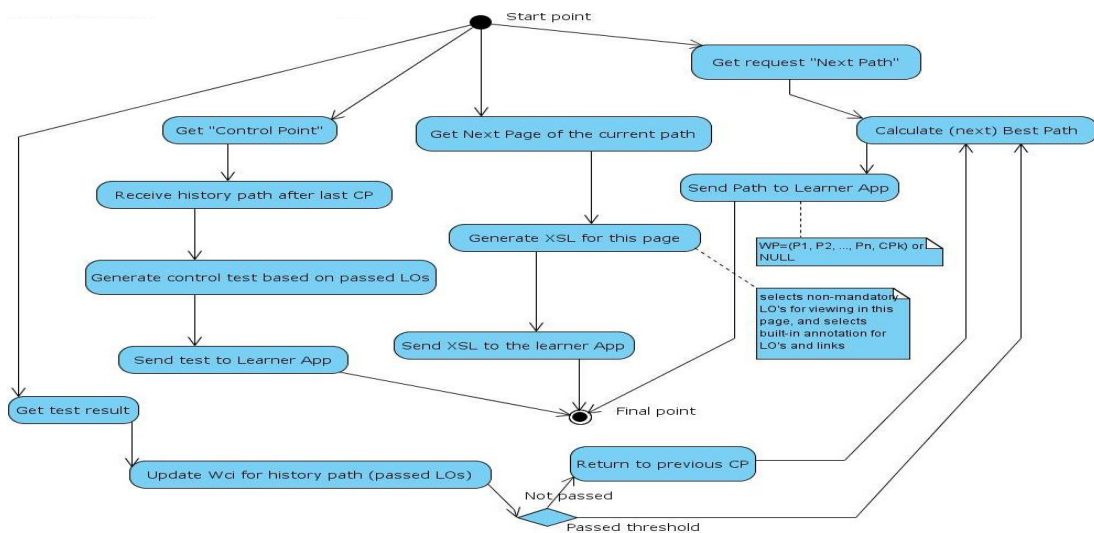


Figure 4. Activity workflow of the adaptation process

After learner assessment, AE recalculates the work path weights for each of the existing learning styles, using the learner performance result at this CP. Thus, the AE tunes (or calibrates) the paths' weights conforming shown learner performance for the path passed by the learner. As well, AE may store some statistics of learner feedbacks to determine which pages are useful for which kind of users. This gives the adaptation engine ability to learn from their skills and perform better estimations for paths for further learners. While performance parameters for each LO are used for adaptive content selection, link annotations are used by the AE for adaptive navigation through the narrative storyboard graph

4. CONCLUSION

The paper presented an innovative work in the area of adaptive hypermedia systems for e-learning. The new features of strong separation between the learner, content and adaptation models and inter-models issues on one side, and the distinction between authoring and instructional design of adaptive courseware on other side, allow powerful and flexible adaptation control and good reuse of learning content and courses. With an easy instructional design of a course and a facile control over the adaptation engine, a given course may be

presented in different ways to different audience. We may encounter here several of the main advantages of both the model and process of adaptive courseware design and delivery, which are expected to be proven during the experiments:

- The learner model is not fixed to anyone of known learning styles groups and separates goals and preferences from learning styles and from learner knowledge and performance, as far as these issues are used for personalization, adaptive navigation and adaptive content selection;
- Both the content ontology and narrative storyboard are structured as graphs but not trees, which makes them much more powerful than other approaches based on trees – for example, multiple subclass inheritance and has-s references have a straightforward implementation by graphs;
- Both the model and process are focused on conceptual issues, answering more the question “Why?” rather than “How?” – for example, metadata definitions of LOs are inherited, assessments at control pages are not specified but built automatically by the adaptation engine using the has-a ontology relation and LOs the learner has passed through, and many others;
- The adaptation process uses adaptive presentation depending on learner preferences, adaptive content selection according shown assessment results, and adaptive navigation at control points by selecting the best path to the next CP and using link annotations but without obliging the learner to select and follow this path; assessment results are used to update the weights of the path the learner has gone through, in order to calibrate the storyboard graph.

Currently the ADOPTA team continues with elaboration of the platform after finalizing the pilot tests. One of the research directions is to develop adaptable assessment at control points by using the last extensions of QTI specification. Another point of investigations will be how to use the model and the process in order to construct adaptive gaming environment.

ACKNOWLEDGEMENT

The work reported in this paper is supported by the ADOPTA project funded by the Bulgarian Ministry of Education and Science via contract no. D002/155.

REFERENCES

- Brusilovsky P., 1994. Adaptive Hypermedia: An attempt to analyze and generalize. Proc. of First Int. Conf. on Multimedia, Hypermedia and Virtual Reality. In *LNC3 1077*, Springer Verlag, pp.288-304
- Dagger, D., Wade, V. & Conlan, O., 2005. Personalization for All: Making Adaptive Course Composition Easy. *Special issue of the Educational Technology and Society Journal*, IEEE IFETS.
- De Bra P., Houben G.-J. & Wu H., 1999. AHAM: A Dexter-based Reference Model for adaptive Hypermedia. *Proceedings of the ACM Conference on Hypertext and Hypermedia*, 147-156.
- Guin-Duclosson, N. et al., 2002. The AMBRE ILE: How to Use Case-Based Reasoning to Teach Methods, *Lecture Notes in Computer Science*, ISBN 978-3-540-43750-5 , Volume 2363/2002, pp. 782-791.
- Halasz, F., Schwartz, M., 1994. The Dexter hypertext reference model. *Communications of the ACM*, Vol. 37, Issue 2, ISSN:0001-0782, pp. 30 - 39.
- Koch, N. & Wirsing, M., 2002. The Munich Reference Model for Adaptive Hypermedia Applications. *Proc. of the Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, ISBN: 3-540-43737-1, pp. 213 – 222.
- Ohene-Djan, J. et al, 2003. Understanding Adaptive Hypermedia: An Architecture for Personalisation and Adaptivity. *Proc. of Workshop on Adaptive Hypermedia AH2003*, August 26-30, 2003, Nottingham, UK.
- Petrov M., Bontchev, 2004. Question and test interoperability – organization, reuse and management of assessments’ contents, Proc. of IADAT Int. Conf. on Education – IADATe-2004, ISBN: 84-933971-0-5, Bilbao, pp.435-439.
- Rey-López, M. et al., 2006. Providing SCORM with adaptivity. *Proceedings of the 15th international conference on World Wide Web*, ISBN:1-59593-323-9, pp.981-982.
- Vassileva D. & Bontchev B., 2006. Self adaptive hypermedia navigation based on learner model characters, *Proc. of IADAT-e2006*, Barcelona, Spain, pp.46-52.
- Vassileva, D., Bontchev, B. & Grigorov, S., 2008. Mastering Adaptive Hypermedia Courseware, *Proc. of 6th Int. Conf. on Emerging eLearning Technologies and Applications ICETA2008*.