6th Int. Conference on

Emerging e-learning Technologies and Applications

The High Tatras, Slovakia
September 11-13, 2008

MASTERING ADAPTIVE HYPERMEDIA COURSEWARE

Dessislava Vassileva, Boyan Bontchev, Slavomir Grigorov
Faculty of Mathematics and Informatics, Sofia University, 5, J. Baurchier blv., Sofia 1164, Bulgaria
Tel./fax +359 2 971 35 43, bbontchev@fmi.uni-sofia.bg, http://www.fmi.uni-sofia.bg/

Abstract. One of the still open gaps in modern e-learning platforms is the lack of adaptation of learning process regarding well-structured learner models. The paper presents an adaptive model of hypermedia learning courseware and the processes of its construction and delivery. Next, it sketches the software architecture of an adaptive hypermedia system (AHS) being under development at Sofia University, Bulgaria. Unlike other AHS, the present one does support of adaptive navigation, presentation and content selection without defining complex rules for controlling narrative storyboards. There are discussed authoring and instructional design of hypermedia courseware for adaptive delivery, and the work process of the adaptive engine for delivering learning objects in a way adapted to a well-structured learner model. In order to illustrate our prototype, we provide UML use case diagrams of the authoring and instructor's application and, as well, explanation of the workflow of the adaptation engine.

Keywords: hypermedia, learning object, metadata, platform, authoring tools, adaptive systems

1. INTRODUCTION

In last fifteen years, authoring and delivery of adaptable e-learning courseware appears to be very important for design of modern learning management platforms. During that period, there have been proposed a lot of works identifying the key challenges in adaptive Web based multimedia information delivery. The chief goal of personalised and adaptive e-learning was formulated by Wade in [1] as assuring of "e-learning content, activities and collaboration, adapted to the specific needs and influenced by specific preferences and context of the student, based on the sound pedagogic strategies". In order to achieve that goal, Adaptive Hypermedia Systems (AHS) possess abilities for provisioning of various forms of adaptation, such as adaptive navigation, structural adaptation, adaptive presentation and historical adaptation [2]. Some research groups focus on adaptability to learners' current knowledge based on the theory of knowledge spaces [3]. The use of learning objects provides an excellent opportunity for learners to apply their own meanings and context to available information [4]. Dynamic adaptation is used in different instructional scenarios with content package adaptation facilitated by wide usage of Web services [5]. Other researchers introduce additional level of system self adaptability based on the idea that different forms of learner model can be used to adapt content and links of hypermedia pages to given user [6]. The self adaptability is based on clean separation of the learner model from the content model and from the adaptation model, without narrative or pedagogical model to be embedded in the content or the adaptation engine. It supposes dynamic changes in adaptation process based on modification of the content parameters according input from learner passing hypermedia resources and assessment about their understanding.

Until the present moment, there have been investigated several main techniques for adaptation [7], as follows:

- Adaptive navigation – the system may manipulate hyperlinks in various ways, e.g. by hiding some of them, and sorting or annotation them;
- Adaptive presentation – here, presentation of page content is adapted for each system user regarding his/her level of knowledge, performance, learning goals or some characters specific for given user;
- Adaptive content selection – the system could show or hide content depending on specific user behaviour;
- Adaptive problem resolution – by means this technique the system would be able to help given user in solving a difficulty or problem when executing a task, in a way adapted to the specific user model.

It is obvious, that applying one or some of the techniques above will strongly depend on organisation and structuring both the models of system user (in particular – the learner) and the domain model known as AHAM reference model [8]. On other way, realisation of techniques normally is presented by the adaptation model (partially supported by the environment model). Thus, we focus in this research on several data models:

- Learner (student) model – stores information about the final user – receiver of the e-learning content – such as personal data, preferences, goals, level of knowledge, performance shown during assessment, etc.;
- Domain model – serves as a repository for structured content for given domain, as well as for its metadata;
- Adaptation model – stores specific rules for adaptable content delivery based on usage of both the learner and the domain models; the rules are to be executed by the adaptation engine to assure e-learning really adapted to individual learners. On other way, the adaptation engine

may change some information within the learner and adaptation models, e.g. information how the assessment result will influence next content delivery.

The next of the paper is structured as follows: part two explains in brief our triangular model for AHS, which extends the AHAM reference model in terms of refining each one of the three basic models explained over. Part three deals with the software architecture of an adaptive hypermedia system supporting our triangular model. Next, we go to construction and delivery of adapted e-learning content by revealing the use case semantics of the authoring tool and the instructor's tool and by explaining the work flow of the adaptation engine. Finally, there are provided conclusion remarks and some of the directions of our future work.

2. A TRIANGULAR MODEL FOR AHS

The AHS model described in details in [11] follows a metadata-driven approach, explicitly separating narrative storyboard from the content and adaptation engine (AE). Fig. 1 represents the triangular structure of our model which refines the AHAM reference model [8] by dividing in three each one of the learner's (or, generally speaking – user's), domain, and adaptation models. This is a new hierarchical organizational model for building adaptive hypermedia learning management system (LMS). At first level, the model is based on a precise separation between learner, content and adaptation model, while at second level each of these sub-model is divided into three others sub-models [9]. All the sub-models should be defined as XML schemas representing the characteristics of a learner that must be modelled and used for cross-session interoperability and consistency. The sub-models may consist of several concepts related or not related each other by some ontology links.

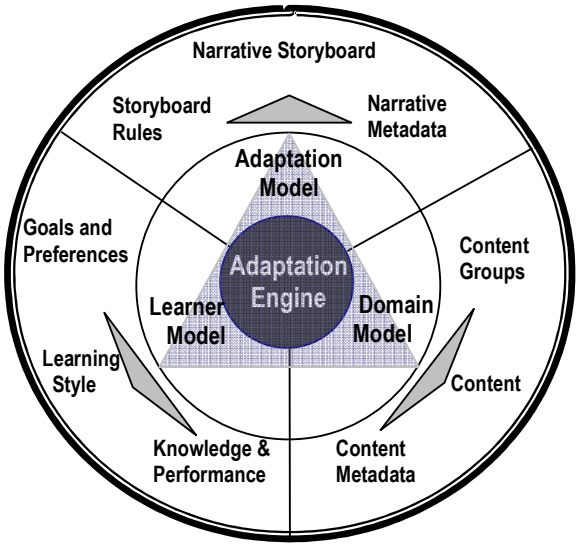

Fig. 1. The triangular model structure

Fig. 2. A sample conceptual learner model

The main benefit of the proposed model is in assuring strong independence of any of the building models and, at the same time, in facilitating a flexible adaptation of content delivery. It can be supported by different system architectures not limiting application of various adaptation techniques, such as adaptive presentation, navigation support and content selection. In order to be able to describe polymorphic learner profiles, we define conceptual characters of given domain such as characteristics of the learning style, psychology characters, etc. Each of the conceptual characters describing the learner has a weight factor Wci (zero or any integer number, or percent between 0% or 100% incl.) specifying the importance or the level of presence of that concept (character) inside the learner model as shown in fig. 2. Thus, a conceptual character having no importance or not being present receives zero weight.

The learner model

Unlike other approaches, in the learner model we separate goals and preferences from shown knowledge and performance, as the first sub-model is static while the second one is rather dynamic and takes a part in the event-driven storyboard monitoring. The model of learning style (learner characters such as visual, auditory, kinaesthetic and others) is detached as another learner sub-model and can be used for choosing contents for given learning style. While the learning style can be determined in the very beginning of the learning explicitly by the learner or by appropriate pre-tests, other tests should be exercised during the e-learning process in order to assess prior or gained knowledge and performance results of each individual student.

The domain model

The domain model is composed of content itself (granulized in learning objects (LOs) according to the SCORM standard) [10], LO's metadata (LOM) and LO's content assets (images, text, tables, etc.) forming a logical taxonomy for the knowledge domain built upon domain ontology during the course composition process by the course author. The content LOs are placed by the instructor on course pages, while pages represent nodes within course storyboard graph. Content pages delivery is controlled by the adaptation engine (AE) for choosing most appropriate content for presenting it to the user with given learning

model. Instead of choosing dynamically a page (i.e. node of the storyboard graph) with its content, we propose choice of best working path within the graph for specific learner with given learning style on one hand, and shown prior knowledge and performance on the other. For this purpose, we define storyboard Control Points (CPs) as nodes of the storyboard graph, where AE either measures learner knowledge/performance, or receives input about satisfaction level of learner's goals and preferences. For the sample narrative storyboard graph presented in fig. 3, CPs are shown as black circles. The path from one control point to another is referred as Working Path (WP). Each working path may consist of one or more nodes (pages) each of one specifying (by storyboard metadata) its LO or LOs.

The adaptation model

The adaptation model (AM) captures the semantics of the pedagogical strategy employed by a course and describes the selection logic and delivery of learning activities/concepts. AM includes a narrative storyboard sub-model supporting course storyboard graphs, which may differ for different learning styles. It consists of control points (CP) and work paths (WP).

Fig. 3 presents a sample narrative storyboard graph summarizing storyboard graphs for several different learning styles. CP's are given in black circles, while other points (nodes) without any control functions are shown in white. With dotted hairlines there are presented all the four WP's starting from CP1 and finishing in CP2. Some of them could be available for a given narrative storyboard for a specific model, some not. Moreover, AM should provide a schema of storyboard rules used for controlling the e-learning process. Storyboard rules determine sequencing of the course pages upon inputs from learner sub-models. The narrative metadata sub-model sets such rules for passing a CP (e.g., as threshold level of assessment performance at that CP) or for returning back to the previous CP.

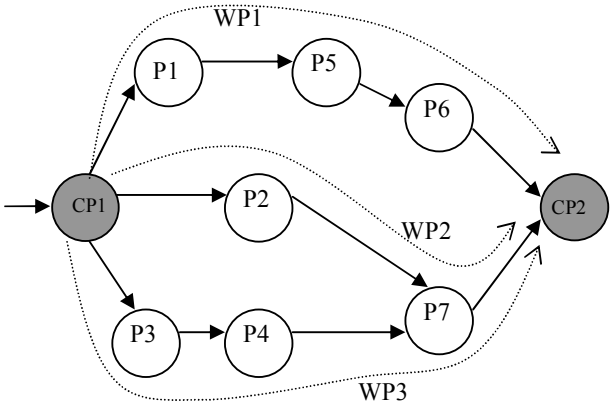

Fig. 3. Sample narrative storyboard graph

The adaptation engine

The core of our model is the adaptation engine (AE) which is responsible for generating the actual adaptation outcomes by manipulating link anchors or fragments of the pages' content before sending the adapted pages to a browser. The AE uses an event-driven mechanism for controlling the storyboard execution based on the storyboard rules applied to the inputs from the learner model. AE selects the best storyboard WP within the graph by evaluating weight coefficient of the pages within the WP for the given learner style [11].

3. SOFTWARE ARCHITECTURE OF ADAPTIVE HYPERMEDIA SYSTEM

The software architecture of the adaptive hypermedia system being under development is component based. Fig. 4 shows a general view of the system by representing a UML deployment diagram. There are four application clients – one of each of the actors (author, instructor, learner and administrator). The server side components of the author and instructor clients are respectively an authoring tool and storyboard graph and page composers. All of them use a common business API. Learning content is structured by means of usage of XML schema/DTD for LOs and metadata and is stores within a content database, while storyboards and learner models are saved in separate databases. The adaptation engine takes central part in the system and communicates to the business API and to the administrator and the learner applications. Next part of the article explains its role and how it assures adaptation of content delivery by means of using the pages and rules mastered with both the authoring and instructor tools.

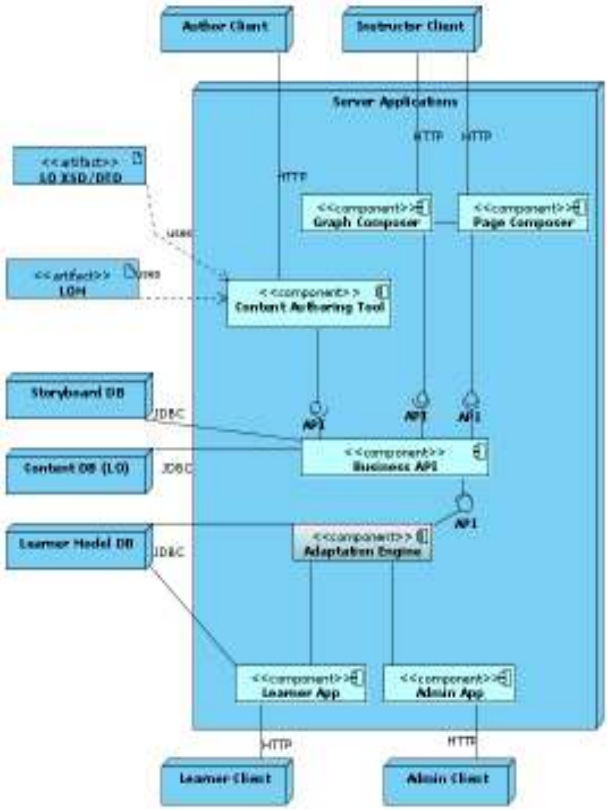

Fig. 4. General view of the system architecture

4. CONSTRUCTION OF HYPERMEDIA COURSEWARE FOR ADAPTIVE DELIVERY

The authoring tool

Our authoring tool (fig. 5) makes a part of the ARCADE (Architecture for Reusable Courseware Authoring and Delivery) e-learning platform [12] but can be used as a separate application. We have integrated its extended version into our system. In this version (fig. 5), the learning content is presented by learning objects (LOs) connected each other within an ontology tree. Each of LOs is described with its metadata accordingly IEEE Learning Object Metadata (LOM). LOM provides more effective search for LOs, reuse of learning content and possibilities interoperability with other authoring tools, environments or repositories. We use XML format for creating LOs, which facilitates usage of web services for learner data exchange and portability towards various platforms. There are supported two kind of LOs – for learning content and for assessment materials. For each LOs the author may define several test questions (being presented as LOs, as well). Then the instructor may use them for test generation and learner knowledge examination. The test questions have different status and user interface from the learning content. A LO may have a hierarchical structure - it may contain several assets - images, texts, multimedia files, external resources, or links and other LOs. LOs representing test questions have to contain as well their answers. Our authoring tool supports three questions type – multi-choice, single choice and Boolean. The LO content is constructed accordingly the Sharable Content Object Reference Model (SCORM) standards and specifications for packaging of web-based e-learning content.

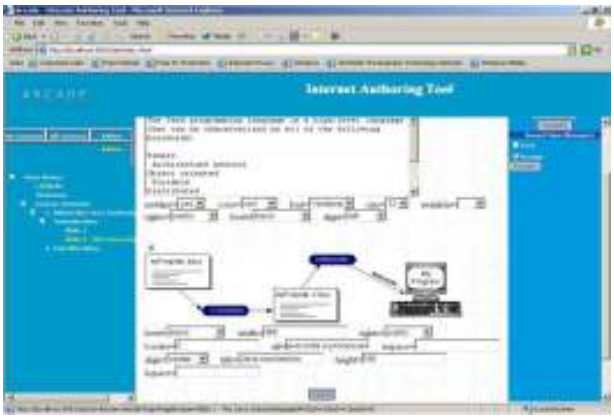

Fig. 5. View of the authoring tool view – creation of LO

As shown in fig. 4, the content authoring application communicates with several other components. Fig. 6 presents main use case diagram of the authoring tool. When the author is logged in, he/she could create, update, delete and read LOs and its internal elements. The author could as well set type of test question and add, delete, and edit answers to it. The author has to fill LOM information for each LO. More, he/she can edit ontology three by moving, adding new or removing LO.

The adaptation engine reads the narrative storyboard graph (created by the instructor) and content pages containing LOs. After then it transforms learning content depending on learner profile and LO type (test or content) to HTML format and delivers this personalised content to the student. If the page is a control point (i.e., assessment page), the adaptation engine generates automatically a test (parameterized by the instructor) according to LOs contained within the working path finishing with this control point.

The instructor (by using the instructor application) may browse LO ontology, read LOs and, finally, compose pages with learning content. Moreover, he/she could copy or drag-and-drop branch of the ontology three or only a single LO.

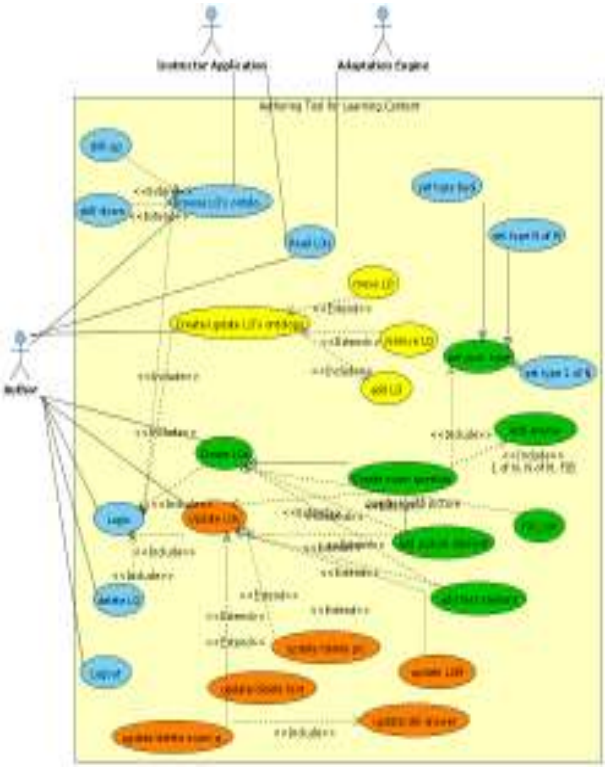

Fig. 6. Author use case diagram

The instructor tool

The instructor tool is an application for creating courses adaptable to different users. Instructor composes a course in terms of interconnected pages represented as nodes of the narrative storyboard and connected each other. 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. Fig. 7 shows instructors drag action from learning objects browser where they are organized in an ontology tree as defined by the author. In the course graph, there is one terminal vertex that represents control point (CP), i.e. course exam. A course exam is generated automatically based on the learning objects used in pages on the work path leading to that CP, and questions related to these LO (as far as they are designed by the course

author and linked to correspondent LO within the ontology tree). Thus, it is not up to the instructor to determine every single question. To tune the course feedback he/she can adjust CP thresholds values, i.e. assessment results for passed exam.

Instructor has also the responsibility to annotate page links and to set page weight parameters for each of the characteristics of the learner model (i.e., parameters showing how much given page with LOs is suitable for given learner character). These page parameters are very important for tuning the system. Adaptation engine use them to decide whether given page would be useful for particular user or not. If a page has high value of the parameter for given learner character and this character is dominant for a particular learner, then this page should be principally shown to that particular learner. Thus, if a work path (from the current control point to the next one) contains many pages suitable for particular user while other path do not, than this work path will be nominated for the best path for such a user. Links annotation labels can be added also by instructor to influent user's decision when a particular user is choosing among several links.

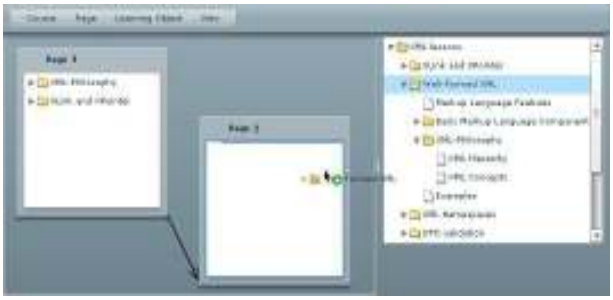

Fig. 7. View of the instructor application

The next figure (fig. 8) represents use case diagram for the instructor tool. The main actors for this module are the instructor and adaptation engine. The instructor uses a web based client application (developed in Adobe FLEX 3, as rich internet application) to login and then to perform all the tasks (the server-side of the application is developed in Java). That includes creating courses, creating pages, filling pages with learning objects, interconnecting pages, adjusting learning objects characteristics, setting link annotations, adjusting exam thresholds, checking user feedback. Adaptive engine uses business API (machine to machine interface) to read course's graph and learning objects characteristics. Then it performs its calculations of the best path for particular user.

The adaptation engine

The adaptation engine (AE) is responsible for performing all necessary adaptation mechanism for content delivery to a specific learner. This includes content selection, content hiding, link annotation, link hiding, etc. Fig. 9 represents the activity diagram of the AE. When learner starts a new course, adaptive engine finds the best path for him in the course graph. The best path is that one with the highest weighed score. For a particular user, the best path is calculated by a sum of multiplications between page parameters values and weights of their correspondent learner's characters. This path is stored for learner as current work path. When learner asks for the next page, adaptive engine may hide objects that are not important for this user. It may also select proper link annotations.

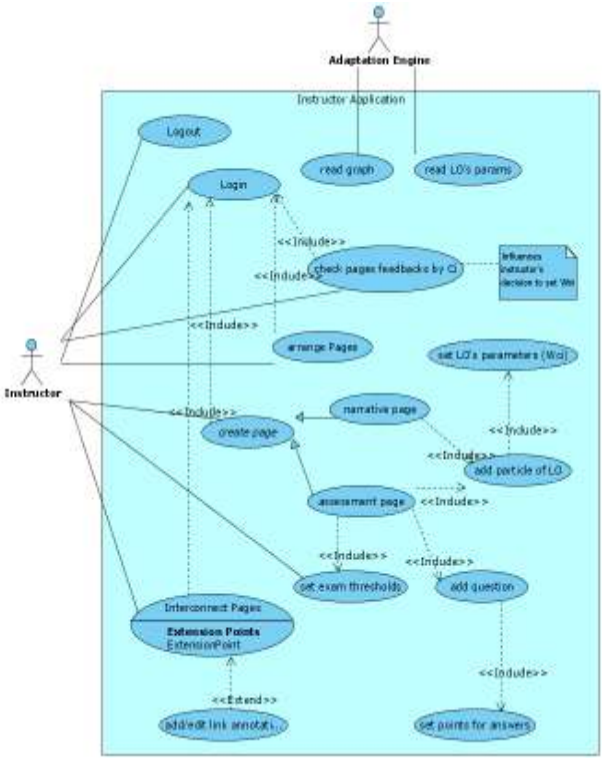

Fig. 8. Instructor use case diagram

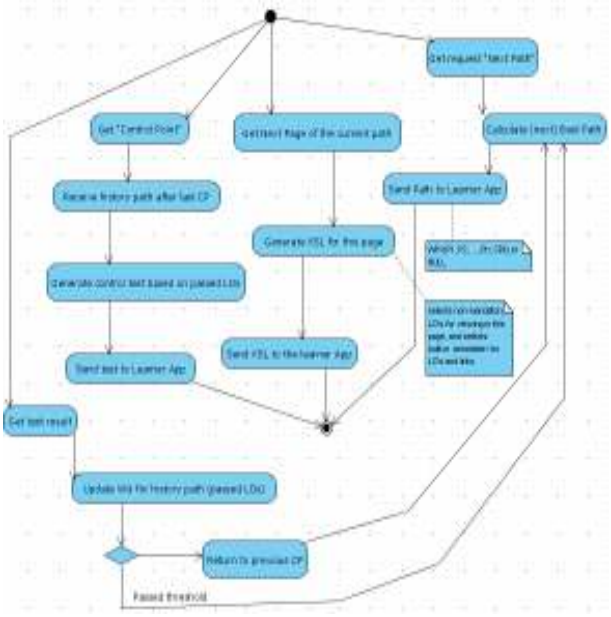

Fig. 9. Adaptation engine activity diagram

As many users are passing through the courses, adaptive engine has to remember user tracks. If a user abandons the

work path determined by AE (by clicking on a link leading to another page outside of the path), the AE continues tracking pages the user has passed through giving the user ability to return back to the path by adding the link "Return to the WP" to each of the pages. 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.

5. CONCLUSIONS

Adaptive e-learning platforms tend to open one of the most promising research areas in next several years. The article presented the conceptual model of an adaptive hypermedia system and, based on this model, the software architecture of a platform for content authoring and delivery. It explained how authors and instructors can master adaptive content and how the adaptation itself is controlled via special engine. Though the project is still under development, we started planning directions for further elaboration of the software platform prototype. One of the issues for future improvement is improvement of the adaptation engine. For better decision making process, its algorithm can be replaced by another one using artificial intelligence and neural networks.

Another improvement can be done in learner application. Accurate information about user can lead to better paths and less noise in feedback statistics. The learner application could monitor user interactions as mouse movement, keyboard stroke, and learn more about his preferences and his learning style. The authoring module can also be improved in terms of extended cross-platform interoperability. We plan to develop tools for import and export of single LOs and ontology branches or threes from external resources as learning repositories or platforms. In this way, authors would have more and more versatile learning content.

6. REFERENCES

[1] Dagger, D., Wade, V., Conlan, O.: Personalisation for All: Making Adaptive Course Composition Easy. Special issue of the Educational Technology and Society journal, IEEE IFETS, 2005.

[2] De Bra, P., Brusilovsky, P., Conejo, R.: Adaptive Hypermedia and Adaptive Web-Based Systems. New York: Springer-Verlag, 2002.

[3] Kayama, M., Okamoto, T.: A Knowledge based Navigation System with a Semantic Map Approach for Exploratory Learning in Hyperspace. Proc. of Int. Conf. ISSEI'2000, 14 - 18 August, 2000.

[4] Conlan, O., Dagger, D., Wade, V.: Towards a Standards-based Approach to e-Learning Personalization using Reusable Learning Objects. Proc. of World Conf. on E-Learning, 2002, pp. 210-217.

[5] Leune, K., W.J. van den Heuvel, Papazoglou M.P.: Exploring a Multi-Faceted Framework for SOC: How to develop secure web-service interactions? Proc. of the 14th Int. Workshop on RIDE, USA, 2004, pp. 485-501.

[6] Díaz, P., Sicilia, M.A. and Aedo, I.: Evaluation of Hypermedia Educational Systems: Criteria and Imperfect Measures. Proc. of the Int. Conf. on Computers in Education, USA, 2002, pp. 621-626.

[7] Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User Adapted Interaction. Ten Year Anniversary Issue, Alfred Kobsa, ed., 2001, pp. 87-110.

[8] De Bra P., Houben G.-J., Wu H.: AHAM: A Dexter-based Reference Model of Adaptive Hypermedia. Proceedings of the tenth ACM Conf. on Hypertext and hypermedia, ISBN:1-58113-064-3, pp. 147-156.

[9] Vassileva D., B. Bontchev.: Modelling reusable adaptive hypermedia for e-learning platforms, Proc. of IADATe-2004, Bilbao, Spain, July 7-9, 2004, pp. 225-229.

[10] Rey-López, M.,Fernández-Vilas A.,Díaz-Redondo R., Pazos-Arias J.: Providing SCORM with adaptivity. Proceedings of the 15th international conference on World Wide Web, ISBN:1-59593-323-9, pp.981-982.

[11] Vassileva D., Bontchev B.: Self adaptive hypermedia navigation based on learner model characters, Proc. of IADAT-e2006, Barcelona, Spain, July 12-14, 2006, pp.46-52.

[12] Bontchev B., Vassileva D.: Internet Authoring Tool for E-learning Courseware, Proc. of the 7th WSEAS Int. Conf. on Computers. Corfu, Greece, pp. 305-311

THE AUTHORS

Dessislava Dakova Vassileva: obtained MSc degree in Mathematics (2002) at Sofia University, Faculty of Mathematics and Informatics. Software developer and consultant. Since 2005 is Ph.D. student in Computer Science at Sofia University, Department of Software Engineering.

Boyan Paskalev Bontchev: obtained MSc degree in Computer Science (1988) at TU-Sofia and PhD degree at Bulgarian Academy of Sciences (1993). Dr. Bontchev followed a career of software engineer and consultant in Portugal, Spain, Italy and Bulgaria, and since 2004 is Associate Professor at Dep. of Software Engineering at Sofia University. Author of more than 50 scientific publications.

Slavomir Vladimirov Grigorov: Graduated with bachelor degree at Technical University of Sofia, Faculty of Computer System and Control. At the moment finishes his MSc degree in Sofia University "St. Kl. Ohridski", Faculty of Mathematics and Informatics. Professional Java developer