DOI:10.2298/CSIS090608021K

Integration of Recommendations and Adaptive Hypermedia into Java Tutoring System

Aleksandra Klašnja-Milićević[1], Boban Vesin[1], Mirjana Ivanović[2], and Zoran Budimac[2]

[1]Higher School of Professional Business Studies
Vladimira Perića Valtera 4, 21000 Novi Sad, Serbia
{aklasnja, bobanvesin}@yahoo.com
[2]Faculty of Science, Department of Mathematics and Informatics
Trg Dositeja Obradovica 3, 21000 Novi Sad, Serbia
{mira, zjb}@dmi.uns.ac.rs

Abstract. A way to improve the effectiveness in e-learning is to offer the personalized approach to the learner. Adaptive e-learning system needs to use different strategies and technologies to predict and recommend the most likely preferred options for further learning material. This can be achieved by recommending and adapting the appearance of hyperlinks or simply by recommending actions and recourses. This paper presents an idea for integration of such recommender system into existing web-based Java tutoring system in order to provide various adaptive programming courses.

Key words: e-learning, recommender systems, adaptive hypermedia, Java tutoring system

1. Introduction

Contemporary e-Learning systems provide useful tools for computer-supported learning, such as forums, chat rooms, discussion groups and e-mail. However, most of them display content and educational material in the same way to all learners, allowing them to choose their own learning pathway through the course, which is not necessarily the most effective one in terms of their previous knowledge or needs. One possible solution to this problem is to use adaptive and intelligent web-based educational systems. These systems can use different recommendation techniques in order to suggest online learning activities or optimal browsing pathways to learners, based on their preferences, knowledge and the browsing history of other learners with similar characteristics. Their main objective is to adapt and personalize learning to the needs of each learner [29].

The task of delivering personalized content is often framed in terms of a recommendation task in which the system recommends items to an active user [15]. Recommender systems help users find and evaluate items of

interest. Such systems have become powerful tools in many domains from electronic commerce to digital libraries and knowledge management [18]. Some recommender systems have also been applied to e-Learning systems for recommending lessons (learning objects or concepts) that learners should study next [11] or for providing course recommendation about courses offered that contribute to the learner's progress towards particular goals [7].

Recommender systems can use data mining techniques for making recommendations using knowledge learnt from the action and attributes of users [18]. The objective of data mining is to discover new, interesting and useful knowledge using a variety of techniques such as prediction, classification, clustering, association rule mining and sequential pattern discovery. Currently, there is an increasing interest in data mining and educational systems, making educational data mining a new and growing research community [17]. The data mining approach to personalization uses all the available information about learners on the web site (in the web course) in order to create user/learner models and to use these models for adaptation of content.

Another approach for course personalization is the use of *adaptive hypermedia* methods and techniques that are used in *Adaptive educational hypermedia systems* [5]. Adaptive educational hypermedia systems can adaptively sort, annotate, or partly hide the links on web pages to make it easier to choose or to recommend to the learners where they should go from a certain point based on his/her goals, preferences and level of knowledge. Learners can be informed about importance and relevance of certain links.

We propose an architecture framework for new web-based learning system that will provide adaptive courses using hypermedia and recommender systems, based on existing web-based Java tutoring system called Mag [26]. To personalize the learning process for each individual leaner, our system needs to use strategies of both recommender systems and adaptive hypermedia.

The previous version of Mag [26] has been used by our students during one semester as a learning tool. Although they used system in informal manner, we gained positive response and comments. It gave us opportunity to evaluate the main features of the system, the results obtained by students and the educational objectives in order to improve its functionalities and characteristics. The new architecture and supplements to Mag system will be proposed based on the opinion of students in order to increase efficiency of this kind of adaptive Java programming course and benefits that it could offer to a learner.

The rest of this paper is organized as follows. Section 2 presents related work from the area of research. Section 3 explains the general proposed architecture and system design for combining recommender systems and adaptive hypermedia. Finally, in section 4, conclusions are drawn and future work is considered.

2. Related Work

Two important ways of increasing the quality of service in e-Learning systems are to make them intelligent and adaptive. These goals are likely to be achieved by different approaches. In some systems, developers apply different forms of learner models to recommend the content and the links of hypermedia course pages to the learner. These systems are named *Adaptive educational hypermedia systems* [5]. Different approach uses aggregation of the recommended materials from other e-Learning web sites and prediction of more suitable material to learners.

Most of the tutoring systems for learning programming languages found on the Web are more or less only well formatted versions of lecture notes or textbooks. Consequently, these systems do not have implemented interactivity and adaptivity. The functions that such systems can perform vary. Some of them are used for learner assessment like JavaBugs [22] and JITS [24], [23], or basic tutoring like Jeliot 3 and Logic-ITA [1], while some of them are adaptive web-based tutorials [16], [21]. One step further in implementation of adaptation was made by systems like JOSH-online [2], iWeaver [28] and CIMEL ITS [27], [8]. Systems are presented in order from those that do not implement any adaptation, towards systems with considerate amount of intelligence implemented.

JavaBugs examines a complete Java program and identifies the most similar correct program to the learner's solution among a collection of correct solutions. After that, it builds trees of misconceptions using similarity measures and background knowledge [22]. They focused on the construction of a bug library for novice Java programmer errors, which is a collection of commonly occurring errors and misconceptions.

Java Intelligent Tutoring System - JITS is a tutoring system designed for learning Java programming [24]. JITS implements JECA (*Java Error Correction Algorithm*), an algorithm for a compiler that enables error correction intelligently changing code, and identifies errors more clearly than other compilers. This practical compiler intelligently learns and corrects errors in learners' program.

Jeliot 3 is a program animation tool aimed to support novices in their learning to program [1]. It graphically displays the execution of object-oriented programs written in Java. The Logic-ITA is a web-based intelligent teaching assistant system that allows students to practice formal proofs in propositional logic whilst receiving feedback [13].

Romero et al. developed a personalized recommender system that uses web mining techniques for recommending a learner which (next) links to visit within an adaptable educational hypermedia system [16]. They presented a specific mining tool and a recommender engine that they have integrated in the AHA! system, in order to help the teacher to carry out the whole web mining process. They made several experiments with real data in order to show the suitability of using both clustering and sequential pattern mining algorithms together for discovering personalized recommendation links.

Another system implemented by Soonthornphisaj et al. [21] allows all learners to collaborate their expertise in order to predict the most suitable learning materials to each learner. This smart e-Learning system applies the collaborative filtering [15] approach that has an ability to predict the most suitable documents to the learner. All learners have the chance to introduce new material by uploading the documents to the server or pointing out the web link from the Internet and rate the currently available materials.

JOSH is an interpreter for the Java programming language [27] originally designed to make easier teaching Java to beginners. Recently the interpreter was restructured into a server based interpreter applet and integrated into an online tutorial on Java programming called *JOSH-online.*

iWeaver is an interactive web-based adaptive learning environment, developed as a multidisciplinary research project at RMIT University Melbourne, Australia [28]. *iWeaver* was designed to provide an environment for the learner by implementing adaptive hypermedia techniques to teach the Java programming language. It implements several established adaptation techniques, including link sorting, link hiding and conditional page content.

CIMEL ITS is an intelligent tutoring system that provides one-on-one tutoring to help beginners in learning object-oriented analysis and design. It uses elements of UML before implementing any code [27]. A three-layered *Learner Model* is included which supports adaptive tutoring by deducing the problem-specific knowledge state from learner solutions, the historical knowledge state of the learner and cognitive reasons about why the learner makes an error [8]. This *Learner Model* provides an accurate profile of a learner so that the ITS can support adaptive tutoring.

None of the above stated systems is complete web-based tutoring system with personalisation options. Some of them are not web-based and are only executed on stand alone machine (JavaBugs, JITS, CIMEL ITS, Jeliot 3) and some of them had just basic interactivity and adaptivity implemented (JOSH-online, JavaBugs, Logic-ITA). Systems like Logic-ITA [13] and Jeliot 3 [1] offered us good ideas and perspective which functionalities could be included in new web-based tutoring system.

This new version of Mag system will integrate content and link adaptation in order to accomplish completely functional web-based tutoring system with personalization possibilities.

3. Proposed Architecture and System Design

In this section, architecture for new web-based learning system that will provide adaptive courses using hypermedia and recommender systems will be presented. The architecture is an extension of existing web-based Java tutoring system called Mag [9] that is developed at Department of Mathematics and Informatics, Faculty of Science, Novi Sad.

3.1. Mag System

Mag is a tutoring system designed to help learners in learning programming languages in different courses [25]. It is an interactive system that allows learners to use teaching material prepared for programming languages and to test acquired knowledge within appropriate courses.

Mag is multifunctional educational system that fulfills three primary goals, identified by earlier exploration in this field [10]. The first goal is to provide intelligent tutoring system for learners in a platform independent manner. The second goal is to provide the teachers with useful reports identifying the strengths and weaknesses of learner's learning process. Finally, the third goal is to provide a rapid development tool for creating basic elements of tutoring system: new learning objects, units, tutorials and tests.

In spite of fact that this system is designed and implemented as a general tutoring system for different programming languages, the first completely proposed and tested version was used for introductory Java programming course.

Preliminary design of he *Mag* system was influenced by several basic system requirements that every on-line learning system for a programming language should have [8]:

- separated user interfaces for learners and their mentors
- easy-to-access tutorials for learners
- various examples for every particular lesson (learning module)
- different tests for every particular lesson that can be adjusted to particular learner
- online programming, compiling and running of programs
- summaries and reports about learner's work
- functionalities for easy monitoring of learner's work
- functionalities for adding new lessons, examples, and tests
- possibilities for communication between learners and mentors.

Two main roles exist in the system, intended for two types of users [26]:

- *learners* - they are taking the Java programming course and will be using the system in order to gain certain knowledge and
- *instructors* (learners' mentors) - their role is to be the lesson and learner database administrator, to track progress of learners learning and to help them with their assignments.

Therefore, two separated user interfaces are provided for both roles: learner (student) and instructor (learner's mentor). Instructor's interface helps in process of managing data about a learner and course material. Learner's interface is a series of web pages that provide two options: taking lessons and testing learner's knowledge. All data about learner and his progress in the course, as well as data about tutorials, tests and examples are stored in the system's server.

3.2. Adaptive Learning and Personalization of a Content Delivery

The ultimate goal for developing Mag system is increasing the learning opportunities, challenges and efficiency. Two important ways of increasing the quality of service of Mag are to make it intelligent and adaptive. Different techniques must be implemented to adapt content delivery to individual learners according to their learning characteristics, preferences, styles, and goals. In order to support adaptive learning and personalization of a content delivery, we must constantly measure the learner's knowledge and progress, build learner model and possibly redirect the course accordingly.

Our goal is to provide two general categories of personalization in the system, based on previously mentioned levels of personalization:

- *Content adaptation* - presenting the content in different ways, according to the domain model and information from the learner model. All learners and contents will be grouped into classes of similar objects in order to recommend optimum resources and pathways. The principle of clustering is maximizing the similarity inside an object group and minimizing the similarity between the different object groups. Such clusters need to be defined in Mag system in order to provide learner with the most suitable learning material and to form the most suitable pathway.
- *Link adaptation* - the system modifies the appearance and/or availability of every link that appears in a course web page, in order to show the learner whether the link leads to interesting new information, to new information the learner is not ready for, or to a page that provides no new knowledge.

Three different levels of personalization (based on the levels of increasing abstraction and sophistication) must be included in the Mag system, which are suggested in [5] and [21]: self-described personalization, segmented personalization and cognitive-based personalization.

Self-described Personalization

The learners will describe their preferences and common attributes with use of surveys or questionnaire, as well as identify their backgrounds and previous experiences. These create the initial learner model to start with in the instruction to follow. Preferred style for every learner is determined with questionnaire filled at the beginning of the course and with optional questionnaire filled at the end of every completed lesson. We consider the fact that surveys and questionnaires are intrusive and distracting in a learning environment [29]. Therefore, Mag also needs to track learner’s achievement and update learner model accordingly.

Segmented Personalization

Learners will be grouped into smaller, identifiable and manageable clusters, based on their common attributes (e.g., class, age), preferences and results

of surveys. Parts of the instruction are then tailored to the groups, and are applied in the same or similar way to all members of a segmented group. Learning material must also be clustered by its purpose, based on benefits it delivers to learners. Therefore, tests in our system will contain three types of questions [9]:

- *Multiple – choice of syntax check*. This type of test is used to ask the learner to trace the correct sample code.
- *Multiple – choice of execution results***.** This type of test is used to ask the learner to choose correct result after execution of offered piece of code.
- *Code completion***.** Problem is presented in form of skeleton program with the specific missed parts of code. The learner is expected to enter appropriate code snippet according to program specification.

Multiple-choice of syntax check questions are used to grade learner's understanding of syntax rules, while multiple-choice of execution results questions shows how learner understands existing code. Code completions tasks are used to check application skills of a learner. If learner has difficulties with a particular kind of questions or tasks, the system will increase their number in next session in order to provide learner with additional opportunities to improve particular skills.

Cognitive-based Personalization

Cognitive-based personalization represents process of adapting and delivering content and instruction to specific types of learners, defined according to information about their capabilities, and preferences. These may include, for example: a learner's preference for specific type of tests or tasks, or linear sequencing over grouping of hyperlinks, as well as recognition of the learner's reasoning capacity and capability for inductive reasoning. This type of personalization is more complex for implementation than the previous types, as it requires collecting data, monitoring the learner's activities, comparing it to other learners' behavior, building a learner model and predicting and recommending what the learner would like to do or see next.

System will consider preferred learning styles for every particular learner before presenting him/her appropriate material. Every learner has his/her own learning style that indicates a preference for some media type(s) over others. Mag will distinguish three learner's learning styles (among others identified by Dunn and Dunn [6]) and therefore use three different presentation methods:

- *Textual*. Learners that prefer to perceive materials as text are provided with lessons, which are in form of text pages with rich formatting and highlighted source code.
- *Visual*. Learners that prefer to perceive materials in form of pictures are provided with illustrations, figures, diagrams, flowcharts, etc.
- *Interactive*. Learners that prefer to interact physically with learning material are provided with interactive flesh animation.

Mag will track improvement that specific learners make while using specific presentation method and update learner model accordingly. In every moment learner can manually switch between different presentation methods.

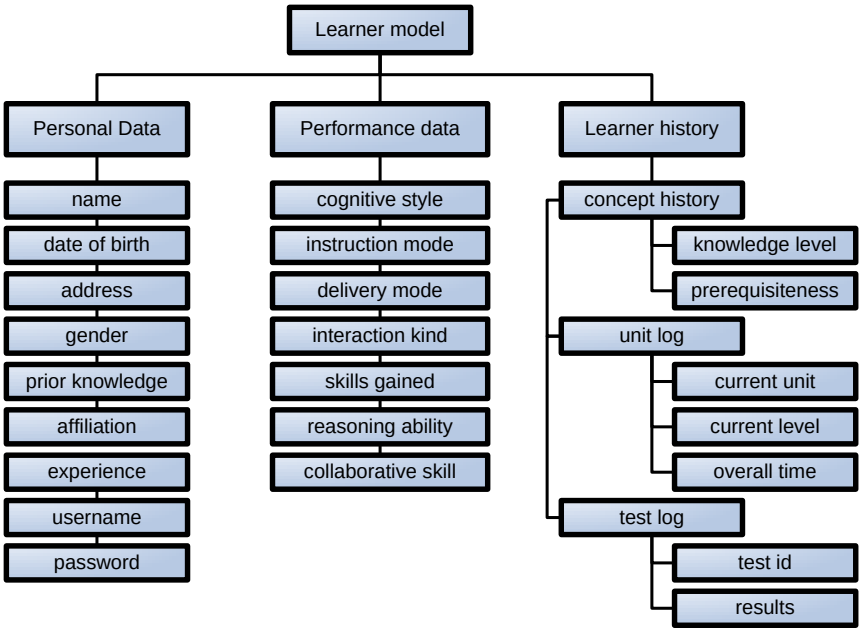

Fig 1. Learner model

We need to add new categories of questions, tests, tasks and tutorials that will provide variety of testing and presenting possibilities during usage of the system. In addition, learners must be categorized into clusters based on the preferable categories of content delivery. That categorization will be accomplished by different surveys (that the learner will be prompt to fill during the registration with the system and optionally after every lesson) and by monitoring the learner's actions, progress and overall performance. In this vein, we plan to track characteristics of the learner and collect a variety of useful information:

- information about the learner, including cognitive, affective and social characteristics,
- information about the learner's perspectives on the content itself, including the learner's feedback on the content, the learner's knowledge of the content (as determined, for example, by a test administered during the learner's interactions with the system),
- information about the technical context of use, including characteristics of the learner's software and hardware environment,

- information about that how the learner interacts with content, including observed metrics such as dwell time, number of learner keystrokes, patterns of access.

Gathered information will be classified along three layers in Learner model that is presented in figure 1.

3.3. Recommendation possibilities

Recommender systems are becoming very popular in e-commerce applications to recommend the online purchase of some products [29]. These systems can be very useful in an e-learning environment to recommend actions, recourses or simply links to be followed.

Recommendation system integrated in a learning environment recommend next task to a learner based on the tasks already done by the learner and his/her success, and based on tasks made by other ”similar” learners. The similarity between learners is established using learner profiles, or is based on common previous access patterns [16]. There are several ways to implement recommendation in e-learning systems [29]. We plan to implement two the most common recommender techniques in the new version of Mag: collaborative filtering and association rule mining [17].

Collaborative filtering. When learner needs suggestion about which location to visit or which test or example will provide the most benefits, the learner profile is compared to the other profiles to find similar [29]. A selection from these similar profiles will be used to produce recommendation. For this mater, we will need good and trusted ratings entered by the learners. The learners will be prompt to fill short questionnaire after every lesson. This questionnaire will be optional because entering ratings could be considered intrusive. If learner refuses to fill the questionnaire, system will use history logs by other learners as input for his/her profile.

Association rule mining. Authors of courses, when setting up the structure of the course, have a certain navigation pattern in mind and assume that the most of the learners would follow a consistent path, materialized by some hyperlinks. Learners, on the other hand, could follow different paths based on their preferences and generate a variety of series of learning activities. Often this series are not the optimum series, and probably they differ from series intended by the designer. All those variety of series of learning activities are noted down, ordered by success of learners that performed them and recommended to the future learners. Therefore, the automatic recommendation in Mag will be based on the author’s intended sequence of navigation in the course material, or based on navigation patterns of other successful learners. This technique is used for recommending shortcuts or jumps to some resources to help learners better navigate the course materials [29].

Those two recommender techniques present a means for the personalization implemented by the Mag adaptation model. This model is part of general architecture of Mag and will be addressed in next section.

3.4. Proposed Architecture of the Hybrid System

The proposed architecture is concerned with producing previously mentioned types of personalization that will be implemented in the already existing learning environment and used by a large number of learners. Open standards, like XML, RDF and OWL [14], [19] needed to be used in order to allow the specification of ontologies to standardize and formalize meaning and to enable the reuse and interoperability. Figure 2 shows a graphical representation of the proposed architecture. This architecture presents adapted architecture of first version of Mag [25] based on experiences of similar web-based learning systems [3], [20], [14] and architecture for ontology-supported adaptive web-based education systems [5] and [4].

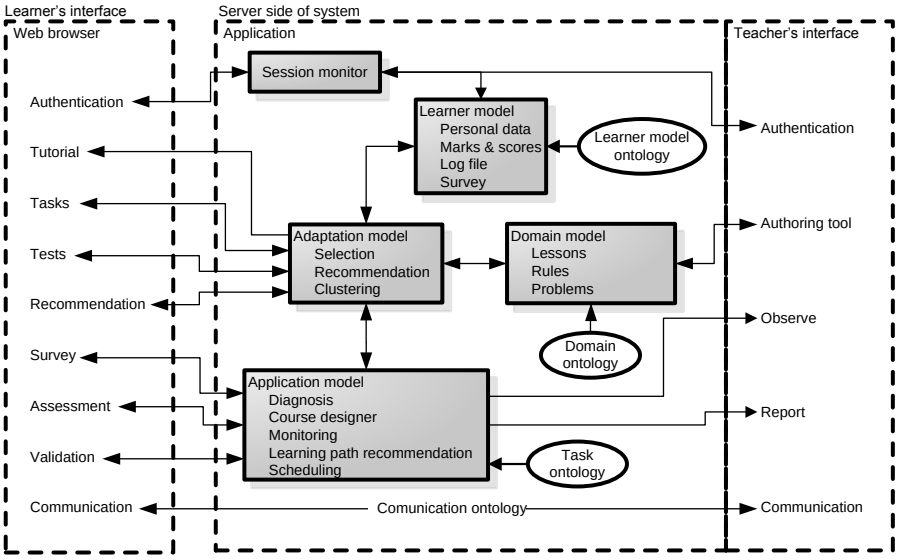

Fig. 2. Proposed system architecture

This is essentially a centralized architecture. The core of the system includes the *adaptation model*, *learner model*, *application model* and *domain model*, all of them stored on a central server. *Domain model* presents storage for all essential concepts in the domain, tutorials and tests. It describes how the information content is structured. Instructors (learner's mentors) can create *domain model* using appropriate authoring tool.

The *application model* applies different strategies and techniques to ensure efficient tailoring of the learning content to the individual learners and personalized task and navigation sequencing. It supports a given pedagogical strategy. For example, that strategy often consists of selecting or computing a specific navigation sequences among the resources based on the information contained in the learner model. The *adaptation model* follows the instructional

directions specified by the *application model* and creates navigation sequence of recourses recommended for the particular learner. These two components are separated in order to make easier adding new content clusters and adaptation functionalities.

The adaptation model is also responsible for building and updating learner model characteristics and for personalization of the application to the learner. It processes changing of learner's characteristics based on learner's activities and it provides an adaptation of visible aspects of the system for specific learner. Its main tasks also include storage and management of course data, ways of presenting courses to learners, provision of reports and test results etc.

Each *learner model* is a collection of both static (personal data, specific course objectives, etc.) and dynamic (marks, scores, time spent on specific lesson, etc.) data about the learner, as well as a representation of the learner's performances and learning history. The system uses that information in order to predict the learner's behavior, and thereby adapt course to his/her individual needs.

Within *session monitor* component, the system gradually builds the *learner model* during each session, in order to keep track of the learner's actions and his/her progress, detect and correct his/her errors and possibly redirect the session accordingly. In the end of the session, the *learner model* is updated. It is then used along with other information and knowledge to initialize the next session with the same learner.

Ontology engineering is a key aspect for the success of proposed web-based educational system. Educational ontologies for different purposes must be included, such as for presenting a domain (*domain ontologies*), for building learner model (*learner model ontologies*) or for presenting activities in the system (*task ontologies*) [20]. A repository of ontologies must be built to achieve easier knowledge sharing and reuse, more effective learner modeling and easier extension of a system. These ontologies will be built based on the e-Learning standards defined in SCORM (*Sharable Content Object Reference Model*) [19]. This ontological representation (OWL/RDF) will enable not only to represent meta-data but also reasoning in order to provide the best solution for each individual learner.

4. Conclusion and Future Work

E-learning environments can use different recommendation techniques in order to suggest the most appropriate online learning activities to learners, based on their preferences, knowledge and the browsing history of other learners with similar characteristics. The ultimate goal is improving the teaching and learning process by providing the learners with personalized courses.

In this paper, we presented the idea for integration of recommender system into an existing web-based Java tutoring system [25] in order to introduce

intelligence in the system and to make it adaptive to individual learner's needs and interactions. The proposed architecture contains elements of two different categories of e-Learning systems. First category is one that applies different forms of learner models to adapt the content and the links of hypermedia course pages to the learner, called *Adaptive educational hypermedia systems (AEHSs)*. Recommender systems for classifying learners and contents in order to recommend optimum resources and pathways are the second category of e-learning systems.

The existing Mag system will be extended according to proposed architecture. New learning objects will be added to system in form of the educational ontolologies. Appropriate clustering of both learning material and learners will be implemented in order to introduce recommendation of appropriate material to specific types of learners. We plan to track characteristics of the learner and collect a variety of useful information about the learner's perspectives on the content itself.

References

1. Bednarik R., Moreno A., Myller N.: Program Visualization for Programming Education – Case of Jeliot 3. Association for Computing Machinery New Zealand Bulletin, p.8 (2006)
2. Bieg C., Diehl S.: Education and tehnical design of a Web-based interactive tutorial on programming in Java. Science of Computer Programming, pp. 25-36 (2004)
3. Chen C.M.: Ontology-based concept map for planning a personalized learning path. British Journal of Educational Technology, Blackwell publishing, pp. 1-31 (2008)
4. De Bra P., Aroyo L., Chepegin V.: The next big thing: adaptive Web-based systems. Journal od Digital Information 5, Article No. 247, p.12 (2004)
5. Devedžić V.: Education and the Semantic Web. International Journal of Artificial Intelligence in Education 14, IOS Press, pp 39-65 (2004)
6. Dunn R., Dunn K.: Teaching Students Through Their Individual Learning Styles: A Practical Approach. Reston Publishing: Reston, VA, (1978)
7. Farzan, R., Brusilovsky, P.: Social Navigation Support in a Course Recommendation System. In proceedings of 4th International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, Dublin, pp. 91-100 (2006)
8. Glenn B., Shahida P., Wei F. Moritz S.: A Web-based ITS for OO Design. 12th International Conference on Artificial Intelligence in Education, Amsterdam, pp. 59-64 (2004)
9. Ivanović M., Pribela I., Vesin B., Budimac Z..: Multifunctional Environment For E-Learning Purposes. Novi Sad Journal of Mathematics, NSJOM Vol. 38, No. 2, pp. 153-170 (2008)
10. Jones N., Macasek M., Walonoski J., Rasmussen K., Heffernan N.: Common Tutor Object Platform – an e-Learning Software Development Strategy. Proceedings of the 15th international conference on World Wide Web, Edinburgh, Scotland, pp. 307-316. (2006)

11. Ksristofic, A.: Recommender System for Adaptive Hypermedia Applications. In Proceeding of Informatics and Information Technology Student Research Conference, Bratislava, pp. 229-234 (2005)
12. Martinez, M.: Designing learning objects to mass customize and personalize learning. in: The Instructional Use of Learning Objects, D. A. Wiley, ed. Article 3.1, p. 29 (2000)
13. Merceron A., Yacef K.: A Web-Based Tutoring Tool with Mining Facilities to Improve Learning and Teaching. Artificial Intelligence in Education, IOS Press, pp. 201-208 (2003)
14. Merino P.J.M., Kloos C.D.: An Architecture for Combining Semantic Web Tehniques with Intelligent Tutoring Systems. ITS 2008, Springer-Verlag Berlin Heideberg, pp. 540-550 (2008)
15. Mobasher, B.: Data Mining for Personalization. In The Adaptive Web: Methods and Strategies of Web Personalization. Springer-Verlag, Berlin Heidelberg, pp. 1-46 (2007)
16. Romero C., Ventura S., Delgado J.A., De Bra P.: Personalized Links Recommendation Based on Data Mining in Adaptive Educational Hypermedia Systems. Creating New Learning Experiences on a Global Scale pp. 292-306 (2007)
17. Romero, C., Ventura, S.: Educational Data Mining: a Survey from 1995 to 2005. Expert Systems with Applications. Elsevier 1:33, pp. 135-146 (2007)
18. Schafer, J.B.: The application of data-mining to recommender systems, Encyclopedia of data warehousing and mining. Hershey, PA. Idea Group, pp. 44-48 (2005)
19. SCORM - http://www.adlnet.gov/scorm/
20. Šimić G.: The Multi-courses Tutoring System Design. ComSIS Vol. 1, No. 1, pp. 141-155 (2004)
21. Soonthornphisaj N., Rojsattarat E., Yim-ngam S.: Smart E-Learning Using Recommender System. ICIC Springer-Verlag Berlin Heidelberg, pp 518-523 (2006)
22. Suarez M., Sison R.: Automatic Construction of a Bug Library for Objected-Oriented Novice Java Programmer Errors. ITS 2008, Springer-Verlag Berlin Heideberg, pp. 184-193 (2008)
23. Sykes E.R., Franek F.: An Intelligent Tutoring System Prototype for Learning to Program Java. The third IEEE International Conference on Advanced Learning Technologies (ICALT'03) Athens, Greece, pp 485-492 (2003)
24. Sykes E.R., Franek F.: Presenting JECA: A Java Error Correcting Algorithm for the Java Intelligent Tutoring System. Advances in Computer Science and Technology - 2004, St. Thomas, US Virgin Islands (2004)
25. Vesin B. Ivanović M., Budimac Z., Pribela I.: Tutoring System for Distance Learning of Java Programming Language. 10th Symposium on Programming Languages and Software Tools SPLST, Dobogókő, Hungary, pp. 310-320. (2007)
26. Vesin B., Ivanović M., Budimac Z.: Learning Management System for Programming in Java. Annales Universitatis Scientiarum De Rolando Eötvös Nominatae, Sectio - Computatorica, vol. 31., pp. 75-92 (2009)
27. Wei F., Moritz S.H., Parvez S.M., Blank G.D.: A Student Model for Object-Oriented Design and Programming. Journal of Computing Sciences in Colleges, pp. 260 - 273 (2005)
28. Wolf C.: iWeaver: Towards 'Learning Style' - based e-Learning in Computer Science Education. Australasian Computing Education Conference, Adelaide, Australia, vol. 20 (2003)
29. Zaiane O.R.: Recommender systems for e-learning: toward non-intrusive web mining. Data mining in e-learning, WIT press, pp. 79-96 (2006)

Aleksandra Klašnja-Milićević is a Lecturer of Computer Science in Higher School of Professional Business Studies at University of Novi Sad, Serbia. She received her Diploma in Electrical and Computer Engineering from Faculty of Technical Sciences at the University of Novi Sad in 2002. She joined the graduate program in Computer Sciences at Faculty of Science, Department of Mathematics and Informatics, University of Novi Sad in 2003, where she received her M.Sc. degree (2007). Her research interests include recommender systems, information retrieval, user modeling and personalization, and electronic commerce. She has published more than 20 referred papers on her work in national and international conferences and journals.

Boban Vesin has graduated at the Faculty of Science, University of Novi Sad, in 2002. He got his master degree at the same Faculty in 2007 and is working on his PhD thesis. Currently he is a lecturer at Higher School of Professional Business Studies, University of Novi Sad. His major research interests are e-Learning and personalization in intelligent tutoring systems. He has published a number of scientific papers in the area.

Mirjana Ivanović is a Professor at Faculty of Science, Department of Mathematics and Informatics, University of Novi Sad, Serbia. Professor Ivanović's research addresses the issues of multi-agent systems, e-learning and web-based learning, data mining, case-based reasoning, programming languages and tools. She actively participates in more than 10 international and several national projects. She has been a program committee member of more the 50 international conferences. Prof. Ivanović is Editor-in-Chief of ComSIS (Computer Science and Information Systems) Journal. She has published over 200 scientific papers in proceedings of international conferences and journals, has written more than 10 university textbooks in the field of informatics and ICT.

Zoran Budimac is a professor at Faculty of Science, Department of Mathematics and Informatics, University of Novi Sad. He graduated in 1983 (informatics), received master's degree (computer science) in 1991 and doctor's degree (computer science) in 1994. His research interests include: mobile agents, e-learning, software engineering, case-based reasoning, implementation of programming languages. He has been project leader for several international and several national projects. He has published over 180 scientific papers in proceedings of international conferences and journals, has written more than 12 university textbooks in different fields of informatics.

Received: June 08, 2009; Accepted: February 17, 2010.