Learner Modelling for Enhanced Teaching of Java Programming Basics

Boban Vesin, Aleksandra Klašnja-Milićević, Zoran Marošan, Ninoslava Savić, Borislav Jošanov

{vesinboban, aklasnja}@yahoo.com, {zomar, ninas, borjos}@uns.ac.rs
Higher School of Professional Business Studies,
Vladimira Perića-Valtera 4, Novi Sad, Serbia

Abstract. Personalisation of the e-learning systems represents the adjustment of the available content and form of teaching materials that are presented to the learner in the learning process. There are different types of personalisation in the form of text adjustment, page selection or choice of different forms of lessons. Personalisation in e-learning systems is performed based on data from the learner model, which is collected through user sessions. In this paper we presented an adaptive and intelligent web-based PRogramming TUtoring System – Protus 2.1. One of the most important features of Protus 2.1 is the adaptation of the presentation and navigation through a course material based on particular learner knowledge. System aims at automatically guiding the learner's activities and recommend relevant actions during the learning process. This paper describes the functionality, structure and implementation of a learner model used in Protus 2.1 as well as performed actions for on-the-fly update of learner model in order to perform personalisation options.

Keywords. Tutoring system, learner model, personalisation, Java programming, Protus

1. INTRODUCTION

E-learning systems experience the rapid growth in the last decades. Also, e-learning has a great potential to continue fast growing in the future. The advantages of learning through a global network are manifold and obvious: the independence from time and space, learners can learn at their own pace, learning materials can be placed in one place and used-processed from around the world (Chatti et al., 2013). E-learning is therefore proved to be efficient, flexible and affordable. Development of e-learning is much more demanding and more expensive than the development of a static system since it is desirable to create different views of the same course content to achieve a better system. Different forms of the same teaching materials are needed for the successful implementation of personalised learning options in the system. However, the personalisation options increase efficiency of e-learning, thus justifying the higher initial cost of their construction.

One of the most important segments in today's development and use of the Internet is the personalisation of content and building user profiles based on the behaviour of each individual user (Weerasinghe & Mitrovic, 2012). The formed profile represents the learner at a given time and it will assist system in the choice of new content and information that is presented to the learner. In order to personalise the learning process and to adapt the content to each learner, e-learning systems must use strategies that will meet the needs of learners. Also, these systems must use different technologies to change the environment and to perform an adaptation of teaching materials based on the needs of learners. The process of adaptation can be in the form of adaptation of content, learning process, feedback or navigation.

In order to provide individual adaptation in the tutoring systems it is necessary to store the information about the learners (goals, preferences, knowledge, etc.) to be used for adaptation purposes (Wang & Chen, 2008). This information constitutes the learner model. To achieve the goal of personalised adaptive learning, prior knowledge helps to distinguish what learners already know and what they do not know (Nguyen & Do, 2008). The learner model represents the state of knowledge of the learner in the concerned subject and helps in deciding the correct teaching strategy to be used for the learner.

In the previous works, we implemented an adaptive and intelligent web-based Programming TUtoring System – Protus 2.1 that applies recommendation and adaptive hypermedia to perform personalisation (Vesin et al., 2012). This system is realized as a general tutoring system for learning different programming languages, but it is completely tested for an introductory Java programming course. The implemented system aims at automatically guiding the learner’s activities and recommends relevant links and actions to him/her during the learning process. The learner knowledge base in Protus 2.1 is represented by an overlay model in which the current state of a learner’s knowledge level is described as a subset of the overall architecture (Vesin et al., 2011). The learner model includes learner’s personal information, background, goals, and learning style as well as his/her competence levels for each concept node and each unit in the content tree, and an overall subject competence level.

The rest of the paper is organized as follows. In the second section appropriate related work is discussed. Section 3 describes personalisation and adaptation options used in Protus 2.1. Details about learner modelling in Protus 2.1 are presented in Section 4. Section 5 presents system's user interface and its basic functionalities. Section 6 brings conclusions and indicates directions of further research.

2. RELATED WORK

Learner model is a collection of static and dynamic data about the learner (Nguyen & Do, 2008). These data are stored within the appropriate data structure. The system uses this information to predict the behaviour of learners as a basis for adapting teaching material. Many researchers presented various suggestions for developing learner models and some of them will be presented in this paper.

Model of a web-based personalised intelligent tutoring system with a learner model is presented in (Li et al., 2008). That model makes use of learners' knowledge levels, psychological characteristics and learning styles in order to construct and update learner details (Yu & Zhiping, 2008).

The learner model of adaptive learning system based on Semantic Web has been presented in (Baishuang & Wei, 2009). This model considers several factors including: learner study style, cognition level, interest and hobby and so on. The authors used Protégé to set up learner model ontology and used data mining technology to update learner model.

Other example of learner model and description of the adaptation mechanisms is presented in (Liu et al., 2008). Authors show that the most relevant difficulty in the knowledge modelling in e-learning systems is related to database structures. Their systems concentrate more on advancing a learner's state of knowledge than on analysing and improving the learner's cognitive state. Besides, system does not facilitate the definition and execution of actions that provide constant updating of learner model.

Other examples of implemented learner models are presented in (Desmarais & Baker, 2012; Li et al., 2012; Wei & Yan, 2009; Nguyen & Do, 2008)

Architecture of implemented learner model in programming tutoring system, described in this paper is based on gathered experiences from previously mentioned systems. Our approach brings acceptable solution for the personalisation process. Further we present a way of linking semantics and content, within learner model in Protus 2.1. Appropriate MySQL tables are defined and populated in order to accomplish effective and scalable learner modelling in Protus 2.1.

3. PERSONALISATION AND ADAPTATION IN E-LEARNING SYSTEMS

Personalisation of the e-learning systems represents the adjustment of the available content and form of teaching materials that are presented to the learner in the learning process based on data from the learner model. There are different types of personalisation in the form of text adjustment, page selection or choice of different forms of lessons. Content can be customized by providing additional material, highlighting important information and with comparative review of different kinds and types of explanation (Devedzic, 2006; Klasnja-Milicevic et al., 2010).

There are different categories of users in the e-learning systems. They vary in age, individual abilities, cultural characteristics, occupation, etc. Some are university students, teachers or employees in organizations from various fields. Some of the users are employees in companies responsible for staff training and they use such systems to increase the efficiency of personnel. Often educational content on the Internet are searched by employees of governmental organizations, researchers and administrative staff. Regardless of their differences they were all interested in a quick and effective learning.

Satisfaction and learning effectiveness largely depends on the content being taught, the way in which it is presented, efficiency, relevance, accessibility of such content, and so on. Since all learners are different, different personalisation must be involved in the development of e-learning systems in order to support the individual and unique characteristics of the learners. Personalisation usually takes into account the different levels of user knowledge and application of various forms of assistance and additional explanations, recommendation of different paths through the course, different principles and techniques of adaptive hypermedia, and so on. In addition, the system for e-learning must provide appropriate security information and reports on the results of learning, performance, ranking, and various statistics about teaching process.

3.1 Learner Modelling

Building of the learner model and tracking related cognitive processes are important aspects in providing personalisation. The learner model is a representation of data about an individual learner that is essential for an adaptive system. The system uses data from learner model in order to predict the learner's behaviour, and thereby adapt to his/her individual needs. Learner model is a collection of static and dynamic data about the learner (Nguyen & Do, 2008). Static data include personal data, specific course objectives, etc. Dynamic data include scores, time spent on specific lesson, marks, etc. Also, learner model contains data about his/her performance and learning history.

Data from learner model in Protus 2.1 is classified along three layers as it is presented in figure 1 (Devedzic, 2006):

- *Objective information*: personal data, previous knowledge, preferences, etc. The learner edits these data during his/her registration on the system.
- *Learner's performance:* data about level of knowledge of the subject domain, his/her misconceptions, progress and the overall performance for particular learner.
- *Learning history*: data about lessons and tests learner has already studied, his/her interaction with system, the assessments he/she underwent, etc.

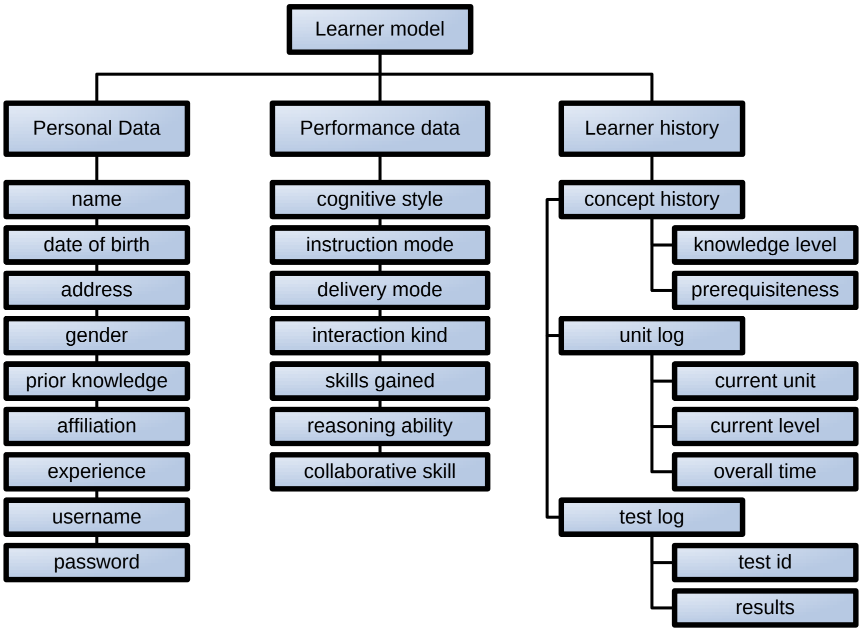


Figure 1. Layers in learner model

Learner model represents the data structure shown in Figure 1. Personal data are basic information about the learners that is used for their identification (name, address, date of birth, etc.). In addition, these data may include details about the initial knowledge of learners, information about the initial learning objectives and interests of learners, their favourite categories of media for presenting information, information about the actions that the learner initiated at individual situations and information about the desired reaction system generated as a result for specific actions of learners. Most of this information is collected directly from learners, therefore system can ask them to fill in the questionnaires and conduct initial testing.

Performance data in the learner model represent knowledge level of particular learner about appropriate topic in the field, the errors learner made, progress and overall performance in the course, etc. Most of these properties are presented quantitatively and the system automatically measures and updates them. Recording and regular updating of the information in the learner model allows the system to predict the learner's responses to various learning activities, suggest next steps, set tasks, and so on.

Learning history includes information about the courses and subjects that the learner has already processed in the system, learning objects which he interacted with, testing that has passed, and so on. Along data on learner progress, the system also monitors and records information about the history of learning.

Some data in the learner model are updated regularly, each time learner log on to the system, while some are updated infrequently or only once. Details of modelling learners will be presented in subsequent chapters.

The quality of the learner model may not be proportional to the amount of data stored in it. Usually, as the amount of data in the model is bigger, the personalisation possibilities are larger and precision of models is greater. However, as the complexity of the model is increased, frequent update becomes a problem.

3.2 Learner Modelling in Protus 2.1

In any educational system, learner model represents objective and subjective information about the learner, his/her interactions, as well as view of the learners’ progress and the history of their activities. Environment or system uses this information in order to maximize the ability to predict the behaviour of learners and adapt to his/her individual needs (Felder & Silverman, 1988; Parvez & Blank, 2008; Vesin et al. 2011).

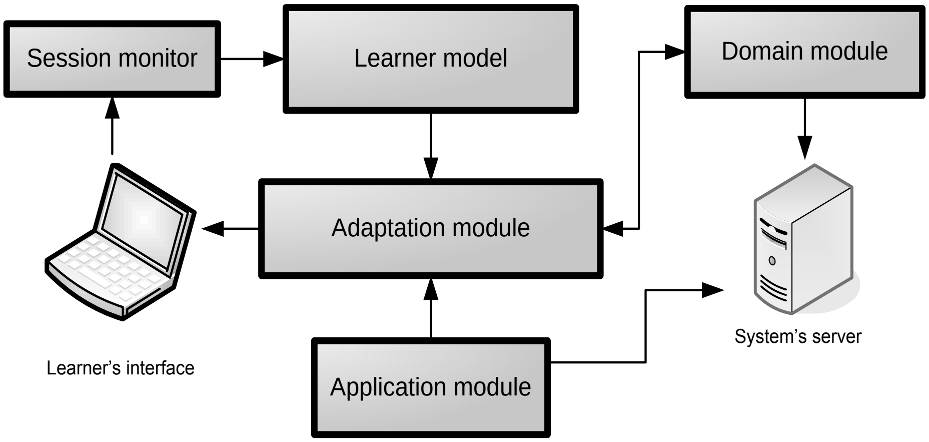


Figure 2. Global architecture of Protus 2.1

Learner model in Protus 2.1 presents crucial element of system's architecture (figure 2). It is a result of monitoring user's sessions and it provides data for implemented adaptation.

Learner model in Protus 2.1 helps in collecting information about learners' interaction with the system: the user name of the learner, the date and time of access and the numerical results of interaction (accessed resources, test results, etc.). This model is automatically updated each time a request or action reaches the web server (for example, data on changing lessons, resources, assessment test, etc.).

The learner model is initialized by a simple but carefully designed questionnaire which is presented to the learner in first session (Klasnja-Milicevic et al., 2010). The initial overall competence level is decided by checking the learner grades of prerequisite courses and previous experience data, if available. The learning styles are assessed by tracking learning behaviour. The learners are allowed to set and modify their learning preferences and goals (Klasnja-Milicevic et al., 2012).

Protus 2.1 system gradually rebuilds the learner model during the session in order to keep track of the learner's actions and his/her progress, to detect and correct his/her errors and possibly to redirect the session accordingly. At the end of the session, all of learners' preferences are recorded. The learner model is then used along with other information and knowledge to initialize the next session for the same learner (Vesin et al., 2012).

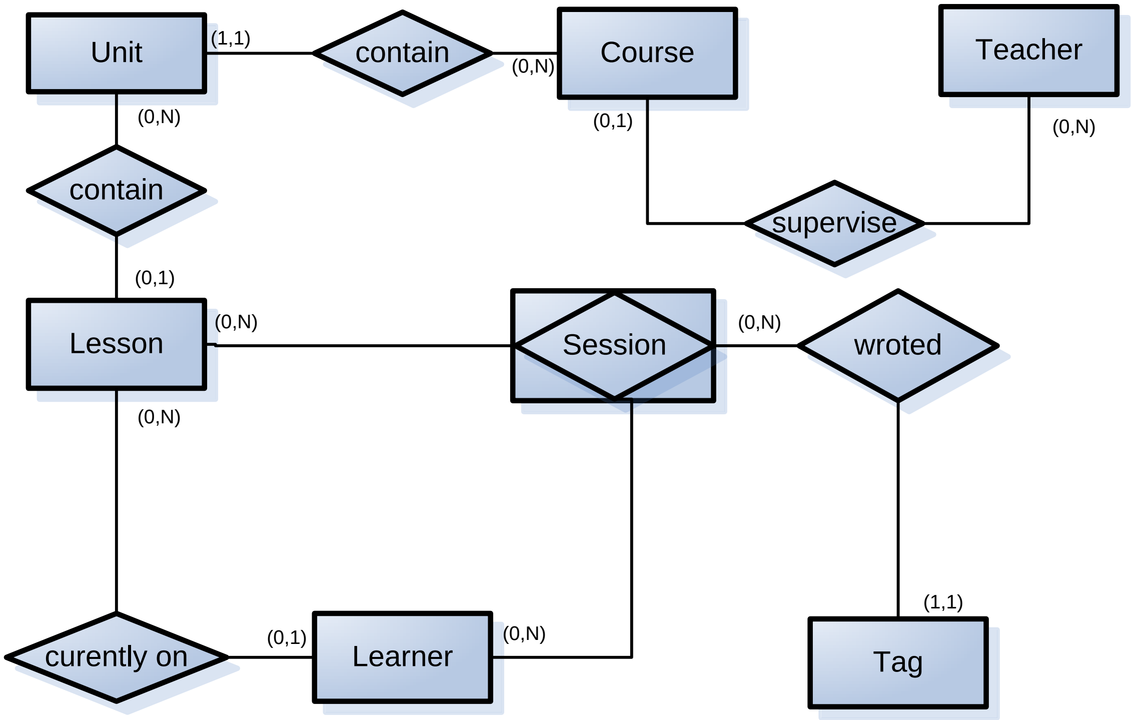


Figure 3. ER diagram of the Protus 2.1 system database

The Protus 2.1 system uses and updates the database with data about learners, teachers, course, unit, lessons, tagging and evaluation process. The ER diagram of the database is shown in Figure 3. The database consists of seven tables:

1. *Learner*. It contains basic information about the learner as well as some information about the learning styles and learner's progress.
2. *Teacher*. It contains basic information about the teacher.
3. *Lesson*. It contains information about the lessons.
4. *Unit*. It contains information about the unit, lesson and learning objects (resources) from which lesson is consisted.
5. *Course*. It contains information about the course, units, lessons, number of learners attending the course and duration of the course.
6. *Session*. It contains information about learner sessions that the learner has completed during the course and the grades he/she earned for them.

7. *Tag*. It contains information about tags and information about lessons and learning objects for which the tag is placed.

The system uses that information in order to predict the learner's behaviour, and thereby adapt course to his/her individual needs.

Figure 4 shows the monitoring and modelling of the learners within tutoring system. The interaction involves all relevant actions undertaken by the learner during a particular session. Interaction always involves some concepts, resources or tests. Interaction may contain some numerical values (score on the test, duration of the visits, the serial number of the visited resources or concepts, etc.).

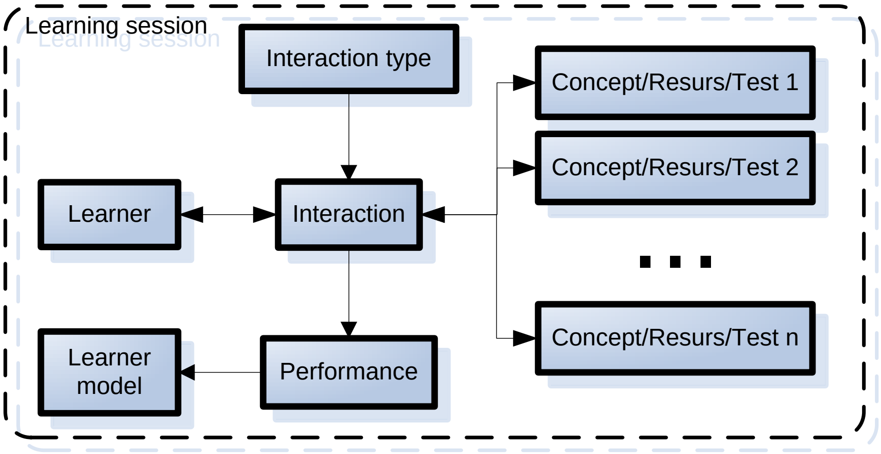
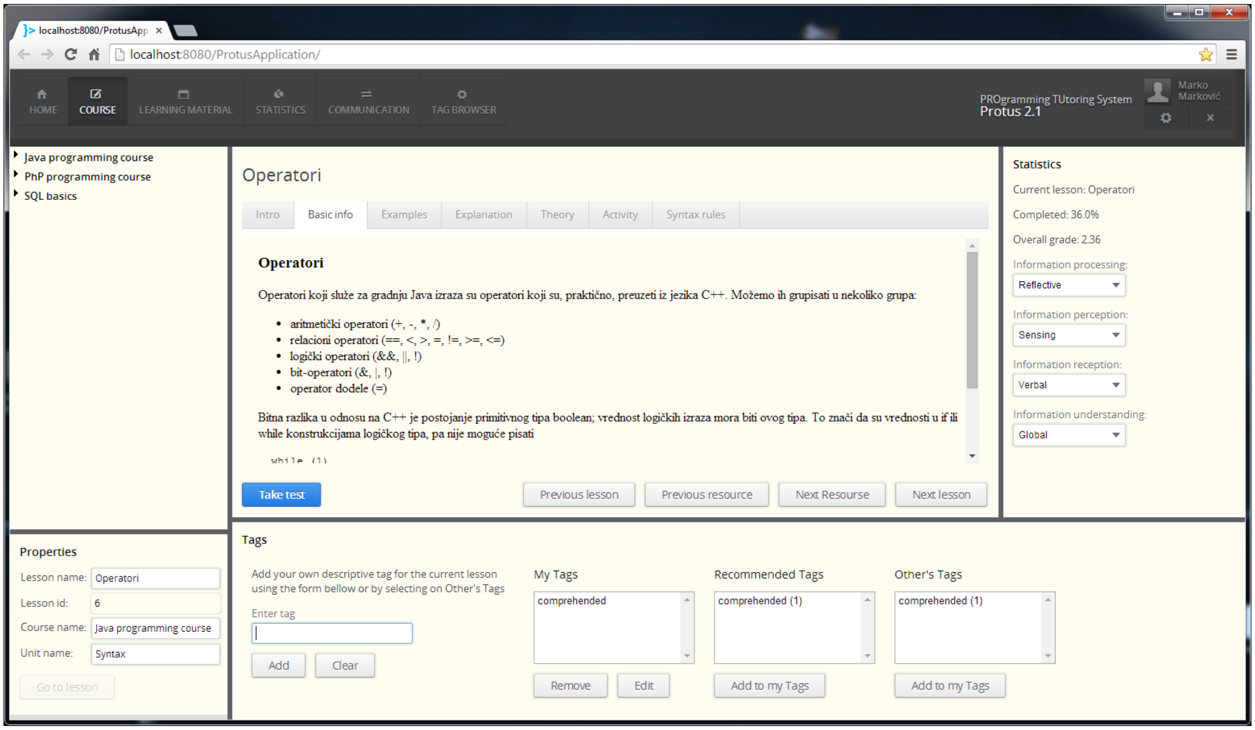


Figure 4. Learner modelling in tutoring system.

4. USER INTERFACE OF PROTUS 2.1

Protus 2.1 is a tutoring system designed to provide learners with personalised courses from various domains. It is an interactive system that allows learners to use teaching material prepared for appropriate courses and also includes parts for testing acquired knowledge. In spite of the fact that this system is designed and implemented as a general tutoring system, the first completely implemented and tested version of the system was for introductory Java programming course (Klasnja-Milicevic et al., 2010). Java is chosen because it is a clear example of an object-oriented language and is therefore suitable for the teaching of the concepts of object-orientation. The course is designed for learning programming basics for learners with no object-oriented programming experience.



Operatori

Operatori

Operatori koji služe za gradnju Java izraza su operatori koji su, praktično, preuzeti iz jezika C++. Možemo ih grupisati u nekoliko grupa:

- aritmetički operatori (+, -, *, /)
- relacioni operatori (==, <, >, =, !=, >=, <=)
- logički operatori (&&, ||, !)
- bit-operatori (&, |, !)
- operator dodele (=)

Bitna razlika u odnosu na C++ je postojanje primitivnog tipa boolean; vrednost logičkih izraza mora biti ovog tipa. To znači da su vrednosti u if ili while konstrukcijama logičkog tipa, pa nije moguće pisati

```
while (1)
```

Statistics

Current lesson: Operatori

Completed: 36.0%

Overall grade: 2.36

Information processing: Reflective

Information perception: Sensing

Information reception: Verbal

Information understanding: Global

Properties

Lesson name: Operatori

Lesson id: 6

Course name: Java programming course

Unit name: Syntax

Tags

Add your own descriptive tag for the current lesson using the form bellow or by selecting on Other's Tags

Enter tag

My Tags: comprehended

Recommended Tags: comprehended (1)

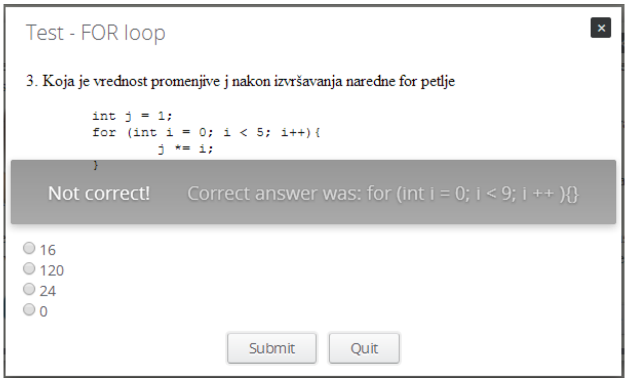
Other's Tags: comprehended (1)

Figure 5. Course options within Protus 2.1

The learning content is divided into units, each of which consists of several lessons (*Concepts*). Every lesson contains several resources (presented in different tabs – Figure 5): *Introduction*, *Basic info*, *Theory*, *Explanation*, *Examples*, *Syntax rules*, *Activity*, etc. To every lesson an unlimited number of resources and tests can be attached. Their number can be increased by teachers using an appropriate authoring tool. Protus 2.1 administrator's module contains additional functionalities for adding new learning material: lessons, resources and tests. This module will be presented in Section 4.1.

4.1 Testing in Protus 2.1

During sessions, learners visit certain resources and solve various tasks. When the learner completes the sequence of learning materials, the Protus 2.1 system evaluates the learner’s acquired knowledge. Tests, designed for every lesson, contain several multiple-choice questions. Protus 2.1 provides feedback on learners’ answers and gives the correct solutions after every question (figure 6). The learners’ ratings are interpreted according to the percentage of correct answers (figure 7).



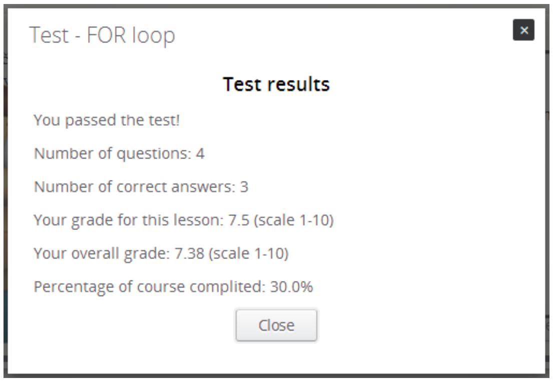
Test - FOR loop

3. Koja je vrednost promenjive j nakon izvršavanja naredne for petlje

```
int j = 1;
for (int i = 0; i < 5; i++){
        j *= i;
}
```

Not correct! Correct answer was: for (int i = 0; i < 9; i ++){}

○ 16
○ 120
○ 24
○ 0



Test - FOR loop

Test results

You passed the test!

Number of questions: 4

Number of correct answers: 3

Your grade for this lesson: 7.5 (scale 1-10)

Your overall grade: 7.38 (scale 1-10)

Percentage of course complited: 30.0%

Figure 6. Feedback after incorrect answer

Figure 7. Test summary

4.2 Evaluation Process

When the learner completes the sequence of learning materials, the Protus 2.1 system evaluates the learner’s acquired knowledge. The learners’ ratings can be interpreted according to the percentage of correct answers, as follows:

- 5 (excellent) - (80–100%)
- 4 (very good) - (70–79%)
- 3 (average) - (60–69%)
- 2 (passing) - (50–59%)
- 1 (marginal) - (0–49%)

This five-point grading scale is based on our secondary school grading system. Consequently, learners have a better sense of having mastered the material using this system of evaluation. The system can be easily transformed and adapted to other standards of grading.

5. CONCLUSION

Although some systems take a learner’s characteristics (e.g. knowledge levels, learning styles, etc.) and needs into account, choice of right learning material or presentation method to the specific learner is especially important in order to reach the desired teaching effects. Hence, various intelligent techniques are introduced in tutoring systems to perform personalised teaching. These techniques produce personalisation based on data from learner models, teaching material, teaching methods etc. In this paper, we presented functionality, structure and implementation of a learner model used in Protus 2.1 as well as performed actions for on-the-fly update of learner model in order to perform personalisation options.

In order to accomplish successful categorization of learners we tracked characteristics of the learner and collected a variety of useful information:

- about the learner, including cognitive, affective and social characteristics,
- about the learner's perspectives on the content itself: feedback on the content, knowledge of the content (as determined, for example, by a test administered during the learner's interactions with the system),
- about the technical context of use: characteristics of the learner's environment,
- about learner's interaction with content: observed metrics such as dwell time,
- number of keystrokes, patterns of access, etc.

We proposed a design and architecture of learner model that can be used in tutoring systems from various domains. This approach allows implementing adaptation customized to different requirements of learners. The learner demand is derived from the knowledge contained in the learner model. For a future work, we consider testing Protus 2.1 and influence of various modelling options and personalisation on progress of the learner.

References

Baishuang, Q., & Wei, Z. (2009). Student Model in Adaptive Learning System Based on Semantic Web. *First International Workshop on Education Technology and Computer Science, 2009. ETCS '09.* Vol. 2, pp. 909–913.

Chatti, M. A., Dakova, S., Thus, H., & Schroeder, U. (2013). Tag-Based Collaborative Filtering Recommendation in Personal Learning Environments. *IEEE Transactions on Learning Technologies*, pp. 337 - 349.

Desmarais, M. C., & Baker, R. S. J. D. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modelling and User-Adapted Interaction*, Vol. 22, Issue 1-2 , pp. 9-38.

Devedzic, V. (2006). *Semantic Web and Education*. Springer science, New York.

Felder, R., & Silverman, L. (1988). Learning and teaching styles in engineering education. *Engineering Education*, *78*, pp. 674–681.

Klasnja-Milicevic, A., Vesin, B., Ivanovic, M., & Budimac, Z. (2010). E-Learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education*, *56*(3), pp. 885–899.

Klasnja-Milicevic, A., Vesin, B., Ivanovic, M., & Budimac, Z. (2012). Personalisation of Programming Tutoring System Using Tag-Based Recommender Systems. In *Advanced Learning Technologies ICALT 2012 IEEE 12th International Conference on*, pp. 666–667.

Li, Z., Sun, Y., Xu, T., & Liu, Y. (2012). The research of classical learner models in intelligent tutoring systems. In *ICCSE 2012 - Proceedings of 2012 7th International Conference on Computer Science and Education*, pp. 1204–1207.

Li, Z., Xu, T., & Sun, Y. (2008). A web-based personalized intelligent tutoring system. In *Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*, Vol. 5, pp. 446–449.

Liu, H., Tang, S., & Ma, L. (2008). A rule-based approach for student modeling. In *Proceedings - 5th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2008*, Vol. 4, pp. 526–530.

Nguyen, L., & Do, P. (2008). Learner Model in Adaptive Learning. *World Academy of Science Engineering and Technology*, *45*, pp. 395–400.

Parvez, S. M., & Blank, G. D. (2008). Individualizing tutoring with learning style based feedback. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* Vol. 5091 LNCS, pp. 291–301.

Vesin, B., Ivanovic, M., Klasnja-Milicevic, A., & Budimac, Z. (2011). Rule-based reasoning for altering pattern navigation in Programming Tutoring System. *International Conference on System Theory Control and Computing ICSTCC 2011 15th*, pp. 644-649.

Vesin, B., Ivanovic, M., Klasnja-Milicevic, A., & Budimac, Z. (2012). Protus 2.0: Ontology-based semantic recommendation in programming tutoring system. *Expert Systems with Applications*, *39*, pp. 12229–12246.

Wang, F.-H., & Chen, D.-Y. (2008). A knowledge integration framework for adaptive learning systems based on Semantic Web languages. In *Proceedings - The 8th IEEE International Conference on Advanced Learning Technologies, ICALT 2008*, pp. 64–68.

Weerasinghe, A., & Mitrovic, A. (2012). Supporting Self-Directed Learning Skills in Learning Management Systems. *Workshop Proceedings of the 20th Int. Conf. on Computers in Education ICCE 2012*, pp. 2–7.

Wei, X., & Yan, J. (2009). Learner profile design for personalized E-learning systems. In *Proceedings - 2009 International Conference on Computational Intelligence and Software Engineering, CiSE 2009*, pp. 1-4.

Yu, S., & Zhiping, L. (2008). Intelligent Pedagogical Agents for Intelligent Tutoring Systems. *2008 International Conference on Computer Science and Software Engineering*, pp. 516–519.