DOI:10.2298/CSIS111231001V

Ontology-Based Architecture with Recommendation Strategy in Java Tutoring System

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

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

Abstract. The aim of Semantic Web is to provide distributed information with well-defined meaning, understandable for humans as well as machines. E-learning is an important domain which can be benefited from the Semantic Web technology. Ontologies, as a building structure of Semantic Web, will fundamentally change the way in which e-learning systems are constructed. The explicit conceptualization of system components in a form of ontology facilitates knowledge sharing, knowledge reuse, communication and collaboration among system components, and construction of intensive and expressive systems. In previous research, we implemented tutoring system named Protus (PRogramming TUtoring System) that is used for learning basic concepts of Java programming language. Protus uses principles of learner style identification and content recommendation for course personalization. The new version of the system called Protus 2.0, supported by several ontologies, as well as examples of its usage for performing personalization are presented in this paper. Architecture of new system extends the usage of Semantic Web concepts, where the representation of each Protus 2.0 component is made by a specific ontology, making possible a clear separation of the tutoring system components and explicit communication among them.

Keywords. Semantic Web, ontology, tutoring system, recommendation systems, personalization

1. Introduction

Semantic Web technologies seem to be a promising technological foundation for the next generation of e-learning systems [9]. *Ontology*, generally defined as a representation of a shared conceptualization of a particular domain, is a major component of the Semantic Web. The initial work on implementing ontologies as the backbone of e-learning systems is presented in [25]. Since that time, many authors have proposed the usage of ontologies in different

aspects of e-learning, such as adaptive hypermedia, personalization, and learner modelling [19].

Interest in ontologies has also grown as researchers and system developers have become more interested in reusing and sharing knowledge across systems [34]. Currently, one key obstacle to sharing knowledge is that different systems use different concepts and terms for describing domains [2]. If we could develop ontologies that might be used as the basis for multiple systems, they would facilitate sharing, reuse and common terminology.

In our previous work, we presented a general tutoring system named Protus (PRogramming TUtoring System) that is used and tested for learning basic concepts of Java programming language [40]. Personalization in Protus is based on principles of learning style identification, content recommendation and navigational sequencing [20], [41]. Learners with different learning styles have different sets of navigation sequence. Hence, learners were clustered based on their learning styles and then behavioural patterns were discovered for each learner by AprioriAll algorithm [22]. Two learners are said to be similar to each other if they are evaluated by the system with the same ratings for a similar navigational sequence. Recommendation process can be carried out according to these learning sequences based on the collaborative filtering (CF) approach [22]. The main objective of this paper is to present the improved version of the system, called Protus 2.0, that implements new, ontology based architecture of the system. This architecture will implement same recommendation techniques for performing personalization that were part of the previous version of the system, but with benefits of Semantic web technologies.

The major goal of learning systems is to support an intended pedagogical strategy [8]. In this scope, pedagogical ontologies can be associated with reasoning mechanisms and rules to enforce a personalization strategy. Often this strategy consists of selecting or computing a specific navigation sequence among the learning resources. Thus, formal semantics are required in this case to enable such computation.

Ontologies in Protus 2.0 are written in OWL [27]. To support the development of the ontologies and the translation in OWL, we use the open-source tool Protégé 4.1 [28]. The use of reasoning mechanisms and rules for knowledge representation and inference engines for reasoning are presented in our previous work [42].

The rest of the paper is organized as follows. In the second section, appropriate related work is analysed and discussed. In Section 3 the representation of components according to Semantic Web technologies within Protus 2.0 will be presented. In addition, we discuss use of ontologies in recommendation process in Protus 2.0. Performed personalization and effects of implemented ontologies are presented in Section 4. Section 5 concludes the paper and indicates directions of possible further research.

2. Related Work

Recently many researches have been focused on applying Semantic Web technologies to different aspects of e-learning [19]. Most of the developed systems use ontologies only for representation of concepts, knowledge or learners' data [13], [14].

The prototype system named SMARTIES is a totally ontology-aware system, which fully utilizes the qualities of ontology, computationally, as well as conceptually [26]. In this moment, that ontology focuses only on the abstract design of learning contents and has not been yet related to domain knowledge or learning objects to concretize the abstract design. We are going to further enhance this work by not only adapting the content modelling but also showing a way how to link semantics and content with implementing rule-based reasoning for adaptation process in Protus 2.0.

The Personal Reader [10] brings another important result in the e-learning field. This system also uses the Semantic Web to personalize and enrich e-learning contents. It presents a service architecture relying on RDF (*Resource Description Framework*) and ontologies to exchange information about learning resources, the domain, and learners. Architecture for personalized e-learning based on Semantic Web technologies was proposed in [15]. The authors propose usage of several ontologies for building adaptive educational hypermedia systems. Unfortunately, in this system ontologies do not represent the teaching strategy's functionality of a resource. A teaching strategy ontology has been implemented in the Protus 2.0 system for clear separation of performed activities in recommendation process.

Our research is closely related to the essences of QBLS system [8]. It is a web-based intelligent learning system that completely relies on Semantic Web technologies and standards. It reuses a large set of learning resources taken from the web and has been used as an online support system for lab sessions of Java programming course. On the other hand, Protus 2.0 implements a complete Java course rather than collecting unstructured learning resources from the web.

Other proposals of ontologies and their usage for several aspects of the e-learning systems, such as learner model and preferences, domain ontology, task ontology, and others, can be found in [1], [23], [32]. In the structure of these systems, the usage of ontologies focuses mainly on learning objects and their related aspects. Besides, that does not facilitate the definition and communication between the other components of the system's architecture. All elements of Protus 2.0 architecture are implemented as Semantic Web components.

In the above-mentioned papers, ontology architecture is presented but specific examples of personalization activities are omitted. The architecture for a tutoring system supported by several ontologies is presented in this paper. The implemented ontology-based architecture offers acceptable solution for the mentioned problems. Personalization actions are presented through concrete examples later in this paper.

In the structure of previously mentioned systems, the use of ontology focuses mainly on learning objects and their related aspects. Besides, that does not facilitate the definition and communication between the other components of the system's architecture. Architecture for tutoring system supported by several ontologies is described in this paper as a way of addressing and offering acceptable solution for the mentioned problems.

3. Protus 2.0 Architecture

Protus 2.0 is a tutoring system designed to support learning processes in different courses and domains. In spite of the fact that this system is designed as a general tutoring system for different programming languages, the first completely implemented and tested version was used for learning in the context of an introductory Java programming course [21]. It is an interactive system with primary goal to allow learners to use teaching material prepared within an appropriate introductory programming course but also includes a part for testing learner's acquired knowledge.

During the last decade, increasing attention has been focused on ontologies and their usage in applications related to areas such as knowledge management, intelligent information integration, education and so on. Ontology engineering, as a set of tasks related to the development of ontologies for a particular domain, offers a direction towards solving a wide range of problems brought by semantic obstacles. According to that, ontology engineering could be a key aspect for improvements of our already developed in traditional manner, tutoring system.

Ontologies allow specifying formally and explicitly the concepts, their properties and relationships [13]. Educational ontologies such as: for presenting a domain (*domain ontology*), building learner model (*learner model ontology*), presenting of activities in the system (*task ontology*), specifying pedagogical actions and behaviours (*teaching strategy ontology*), defining the semantics of messages sent among components (*communication ontology*) and specifying behaviours and techniques at the learner interface level (*interface ontology*), must be included in the system [9]. A repository of ontologies must be built to achieve easier knowledge sharing and reuse, more effective learner modelling and easier extension of whole system. Ontologies are structured following the SCORM (*Sharable Content Object Reference Model*) e-learning standard [30] using the formal ontology language OWL [32]. This ontological representation enables not only to represent meta-data but also reasoning in order to provide the best solution for each individual learner [42]. General ideas, of redefined ontology-based architecture for Protus 2.0 have been presented in [20]. This architecture is based on experiences gained from similar web-based learning systems [5], [24], [31] and architecture for ontology-supported adaptive web-based education systems suggested in [7], [25], [37]. All the proposed architectures are highly modular with four central components: the application module, the adaptation module,

the learner model and domain module. In all proposed architectures, the adaptation module is explicitly separated from the domain module, but another component is introduced in Protus 2.0 as in [25] – the application module. This module is used for storing adaptation rules used in further personalization process. Figure 1 depicts the general architecture of the redesigned and extended version of Protus 2.0 system.

It is important to note that the original architecture of Protus did not bring any kind of homogenous representation of components. Each one was represented by different formats, using a variety of tools. The purpose, of our current research activities, was to represent each component of the system in form of the ontology. According to that, the level of abstraction of this architecture will be higher [18].

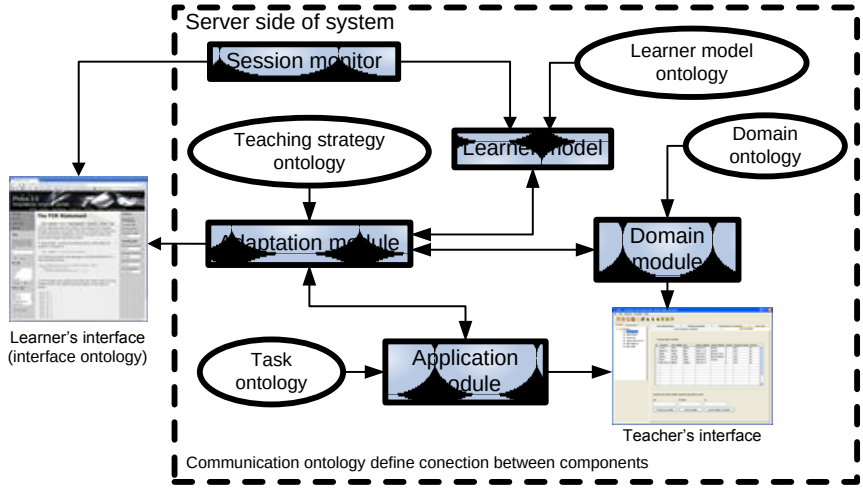

Fig. 1. Protus 2.0 Architecture

This approach will make it easier to understand the role of each component and, consequently, to promote interoperability among the components of the architecture. The developed system is modular, which allows better flexibility and future replacement of various components as long as they comply with the current interface. In the rest of the section, details of construction of different kinds of ontologies and their use for performing personalization employed in our system will be separately addressed.

Concrete examples of the ontologies and their use for the purpose of personalization within Java tutorial will be presented.

3.1. Domain Ontology

One of the main goals of the learning process is to understand and to acquire a body of knowledge for a given domain [29]. Domain model presents storage for all essential learning material, tutorials and tests. It describes how the

content intended for learning has to be structured [3]. Often the domain model can be structured as a taxonomy of concepts, with attributes and relations connecting them with other concepts, which naturally leads to the idea of using ontologies to represent this knowledge.

The complete Java course contains several concepts (lessons) [16]. Therefore, the Java course in Protus 2.0 contains: an introductory lesson, syntax, loop statements, execution control, etc. (Figure 2). To each concept any number of different resources (text files, images, animations, etc.) can be assigned. All resources are assigned depending on their *Resource type*: theory, examples, assignments, exercises, syntax rules, and so on.

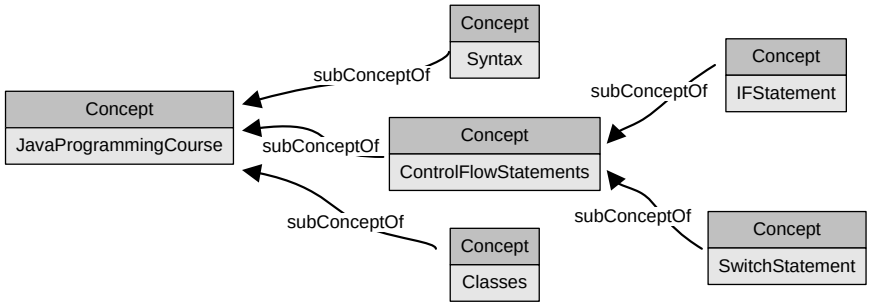

Fig. 2. An excerpt of ontology as domain topology of Protus 2.0

Concepts. In Figure 2, an excerpt of a domain ontology covering basics of Java programming concepts with *subConceptOf* relationships between these concepts has been shown. This figure depicts the root concept with some of its sub concepts: *Syntax*, *ControlFlowStatements* and *Classes*. The *ControlFlowStatements* concept is further specialized and fine-grained into *IFStatement* and Switch*Statement*. Clear specification of other relations between concepts will be useful for further personalization purposes.

An example of instance of *Concept* class that is used to collect information about the *For Statement* concept is presented in Table 1.

Table 1. Example of instance of *Concept* class

Property description	Property name	Property value	Property type
Concept's id	hasId	C009	Datatype property
Concept's name	hasName	ForStatement	Datatype property
Resource's type	hasResource	R017	Object property
Superclass	subConceptOf	LoopStatements	Object property
Prerequisite	hasPrerequisite	ExecutionControl	Object property
Prerequisite	hasPrerequisite	Syntax	Object property

A *Property* is a directed binary relation that specifies class characteristics. They represent attributes of instances and sometimes act as data values (*Datatype property*) or link to other instances (*Object property*). OWL also has a third type of property - *Annotation properties*. Annotation properties can be used to add information (metadata) to classes, individuals and object/datatype properties. OWL allows classes, properties, individuals and the ontology itself to be annotated with various pieces of information/meta-data. These pieces of information may take the form of auditing or editorial information. For example, it could be details about creation date, author, comments or references to resources such as web pages etc.

This particular instance of *Concept* class (Table 1.) has unique id: C009. It has been used for defining a lesson named *ForStatement* and it contains data about its superclass (it is *subConcept* of *loopStatements* concept) and concepts that are prerequisite for it (*ExecutionControl* and *Syntax*).

Resources. All concepts must be supported with various types of resources. When the learner accesses a course, Protus 2.0 can infer which resources could be suitable for presentation to the learner.

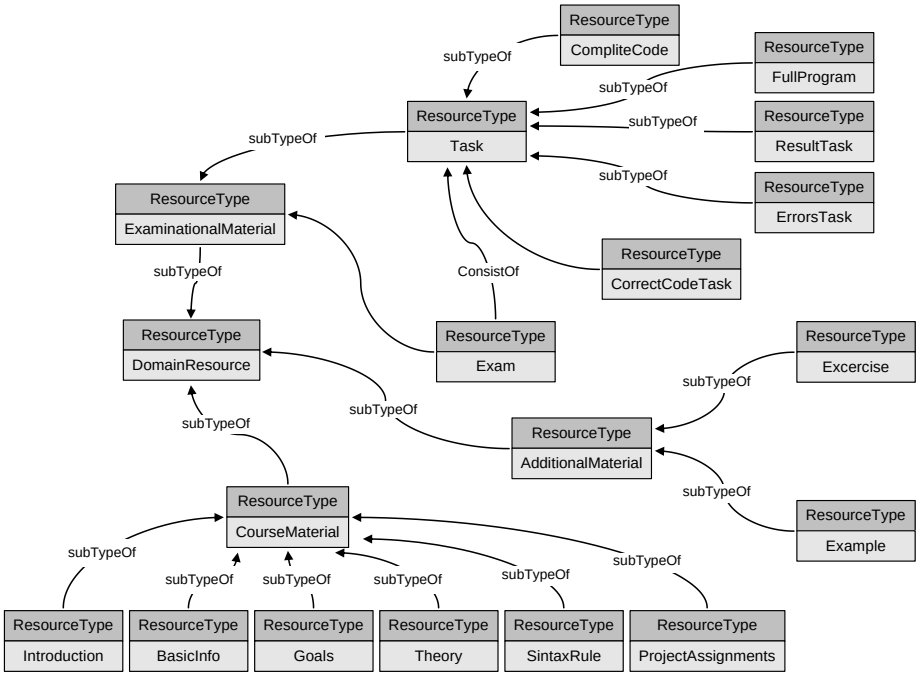

Fig. 3. An excerpt of ontology as resource topology of Protus 2.0

An excerpt of ontology as *Resource type* topology is depicted in Figure 3. The ontology depicts resource types in the programming domain. The most general resource type is *DomainResource*. *DomainResource* has three subtypes: *CourseMaterial, AdditionalMaterial* and *ExaminationMaterial*.

Classes *CourseMaterial* and *AdditionalMaterial* represent the theoretical and practical explanations, respectively, that are displayed to the learners. *ExaminationMaterial* can be further specialized to *Task* and *Exam*. The *Exam* is consisted of various *Tasks* [17]. Tasks could include code completion, code correction, listing errors, etc.

CourseMaterial can be further specialized into *Introduction*, *BasicInfo*, *Goals*, *Theory*, *SintaxRule* and *ProjectAssignements,* which corresponds to the essential elements of a programming language course.

Ontology presented in the Figure 3 further provides information for the *Task ontology* and the *Teaching strategy ontology* which will be explained in more details in the rest of the section.

Details about resources are kept in *Resource* class instances. Each instance of the *Resource* class contains basic information on individual resources, which will later be used for the subsequent selection of appropriate resources in the process of personalization. Specific type and role are determined for every resource.

An example of instance of the *Resource* class that is used to display the syntax rules of *for* statement is presented in Table 2.

Table 2. Example of instance of the resource class

Property description	Property name	Property value	Property type
Resource's id	hasId	R017	Datatype property
Resource's name	hasName	forLoop017	Datatype property
Resource's type	isTypeOf	Syntax rule	Object property
Concept's type	isResourceFor	For Statement	Object property
Resource's role	supports	Visual style	Datatype property
Is resource visited?	isVisited	yes	Datatype property
Is resource recommended	isRecommended	no	Datatype property
File Type	hasFileType	jpg	Datatype property
Concept's role	hasRole	definition	Datatype property
Link to used figure	hasFigure	Figure6.jpg	Annotation properties

This particular instance of *Resource* class has unique id: R017. It is used for presenting a *syntax rule* for a lesson (concept) named *ForStatement* and it contains a link to a certain jpg file (Figure 4) that will be presented to the learner if the system chooses this resource during personalization activities.
All resources are grouped by their type, role and the concept they support and that present a basis for successful recommendation during the personalization process.

3.2. Task Ontology

Task ontology is a system of vocabulary for describing problem-solving structure of all existing types of tasks domain independently. It complements the domain ontology by representing semantic features of the problem solving [9]. Task ontology specifies domain knowledge by giving roles to each object and relations between them. For instance, what role a paragraph in a textbook, can play.

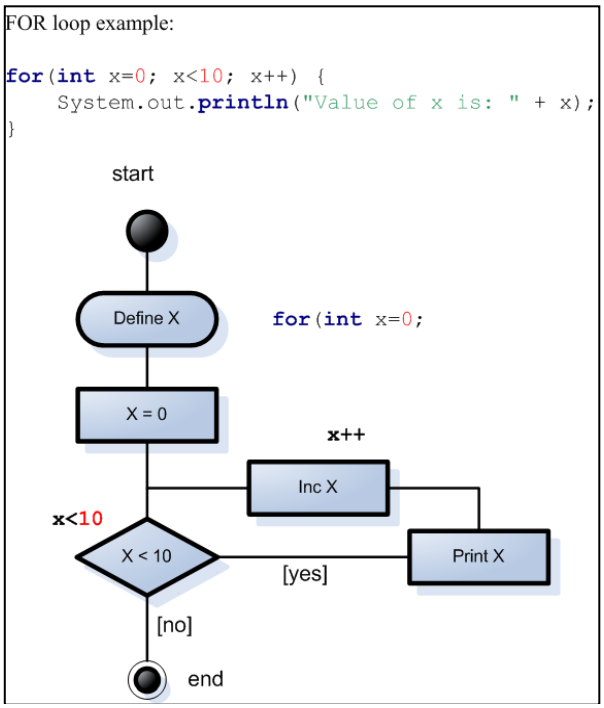

Fig. 4. Figure resource

Task ontology shows the role of a specific resource from the domain ontology. For example, if a resource has *fact* or *definition* role it is used to increase basic knowledge and if its role is *example*, then it is used to increase learner‘s practical skills.

An excerpt of task ontology of resources in Protus 2.0 is depicted in Figure 5. The ontology represents learning material grouped by the resources. The class *Concept* is used to annotate a unit of knowledge, which is represented by some *Resource*.

Like in [8], concepts and resources are related by the *hasResource* property. Concepts can be arranged by the *hasPrerequisite* property. The *hasPrerequisite* property is proposed for navigational purposes. It allows pointing out concepts that must be known before starting to study a concept,

and the concepts for which it is a prerequisite. Concept will not be covered unless that the prerequisite condition is satisfied. There can be a different sequence of resources that depends on the navigational sequence determined for a particular learner.

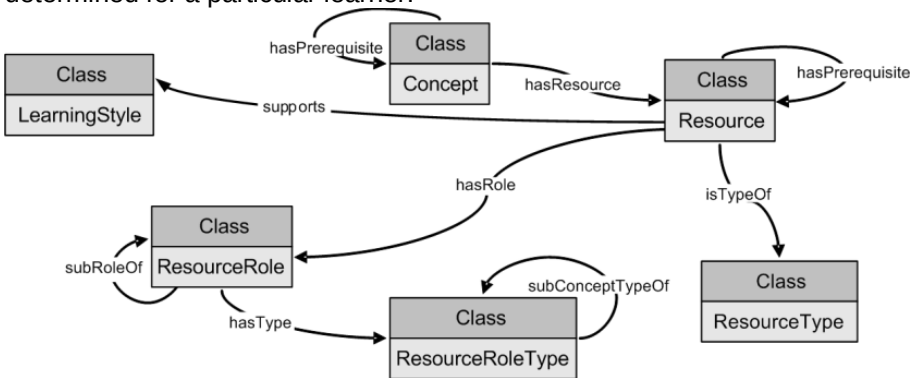

Fig. 5. An excerpt of task ontology of resources

Resources play certain roles in particular concept fragments. For example, some resources represent the crucial information, while the others just represent a mean to provide additional information or a comparison. In the proposed ontology, we represent these facts by instances of the *ResourceRole* class and its two properties: *hasRole* and *supports*. For example, resources like *BasicInfo* and *Example* have different roles. The role of the first is to represent introductory information for lesson and the role of the former is to provide additional information. On the other hand, both concepts support adaptation to learner with *Reflective* style of learning [12]. Resource properties can be further extended by assigning a *ResouceType*. Similarly, the resources roles can be further extended by specifying their types. Concepts, their types and resources form the task ontology of Protus 2.0 system.

3.3. Learner Model Ontology

The learner model stores personal preferences and information about the learner's mastery of domain concepts [36]. The information is regularly updated according to the learner's interactions with the content and is used by the *Teaching strategy ontology* to draw conclusions and decisions. This ontology (Figure 6.) offers the opportunity to map all information about the learner, from confidential data, like password, to a knowledge evolution history.

The class *Learner* is built from three components: *Performance*, *PersonalInfo*, and *LearningStyle*. These three classes are related to association through *hasPerformance*, *hasInfo*, and *hasLearningStyle* properties.

Class *LearningStyle* represents the preferred learning style for particular learner. This class offers four categories to the dimensions of the Felder-Silverman Learning Style Model (sequential/global, active/reflective, visual/verbal and sensing/intuitive) [12].

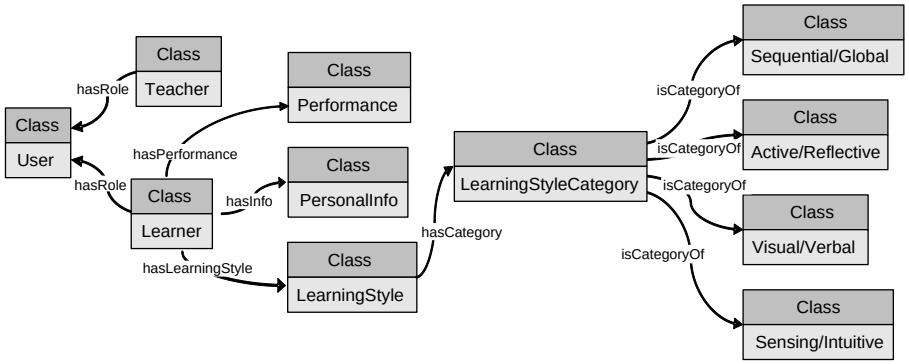

Fig. 6. Learner model ontology of Protus 2.0

During a learning session, the learner interacts with a tutoring system. Learner interactions can be used to draw conclusions about his/her possible interests, goals, tasks, knowledge, etc. These conclusions can be used later for providing personalization. Ontology for learner observations should therefore provide a structure of information about possible learner interaction. Figure 7 depicts such ontology as a part of *Learner model ontology*. Learner performance is maintained according to a class *Interaction*. *Interaction* is based on actions taken by a specific learner, during a specific *Session*. Interaction implies a *Concept* learned from the experience, which is represented by the *conceptUsed* property. *Interaction* has a certain value for *Performance*, which is in this context defined as a floating-point number and restricted to the interval from 1 to 5. This ontology is responsible for updating the *Learner model ontology*.

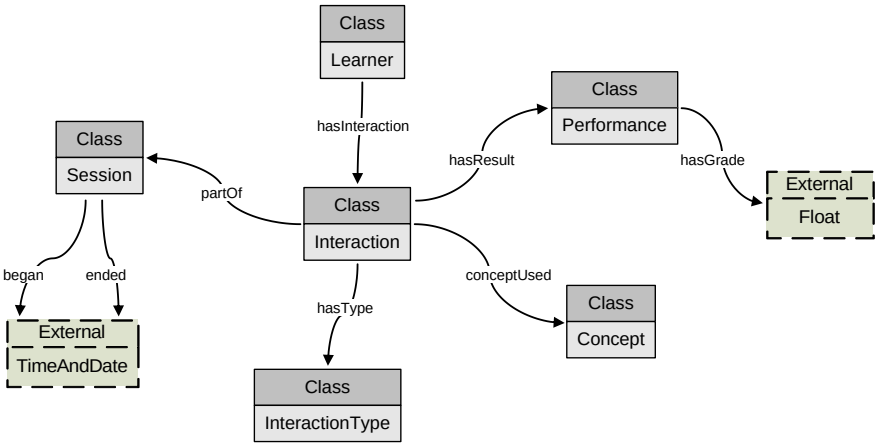

Fig. 7. Ontology for learner observation and modelling

An example of instance of *Interaction* class that is used to keep track of the learner's interaction during one session is presented in Table 3.

Table 3. Example of instance of *Interaction* class

Property description	Property name	Property value	Property Type
Interaction's id	hasId	I007	Datatype property
Session	partOf	S1012	Object property
Interaction's type	hasType	T02 (test)	Object property
Used concept	conceptUsed	C032	Object property
Learner	whoInteracted	L01	Object property
Performance	hasResult	P01	Object property

This particular instance of *Interaction* class has unique id: I007. It is formed during session S1012 when learner with id L01 took test and gained results, which are all collected in instance of *Performance* class with id P01. Instances of class *Performance* contain, among other, data about grade that learner earned during current testing. Based on that *Performance* data, system makes decision in further personalization within *Teaching strategy ontology* described in next section.

3.4. Teaching Strategy Ontology

Authoring of adaptation and personalization is actually authoring of learner models and applying different adaptation strategies and techniques to ensure efficient tailoring of the learning content to the individual learners and their learning styles and navigation sequencing [1].

Figure 8. shows how the adaptation is carried out by the *Teaching strategy ontology*. The decisions are drawn on the basis of the information contained in the *Condition* class (that is generated by the information about learning style and performance of the learner) as well as teaching goals and previous behaviour patterns. These conditions are composed of data coming from several other components such as *Learner model ontology*, *Task ontology* and *Domain ontology*.

Personalization presents the choice of the most appropriate learning pattern or resource that will be recommended to the learner. This action depends of many conditions but it implies only one decision. The decision determines what concept and resource the system is going to present for the learner.

An instance of the *Condition* class that was formed based on the *Performance* data of every learner is presented in Table 4. This particular

instance of *Condition* class has unique id I006. It contains data collected based on learner's learning style and performance.

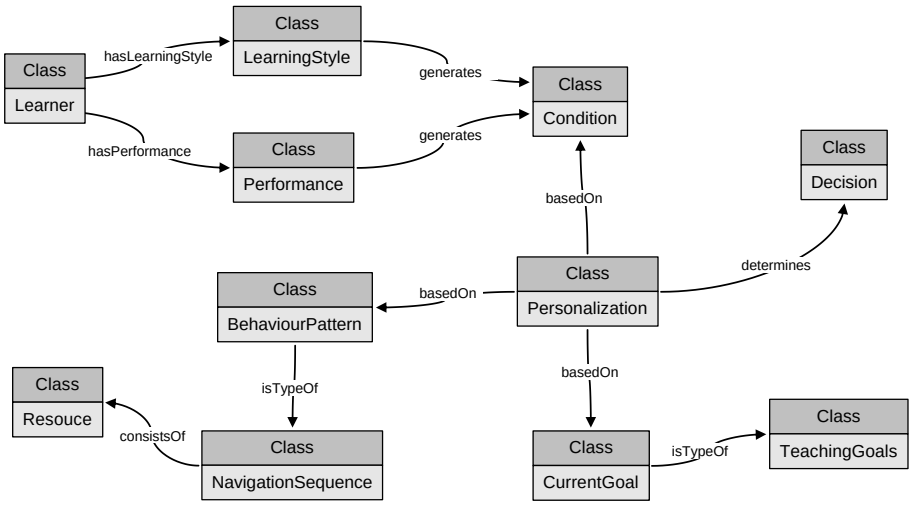

Fig. 8. Teaching Strategy ontology of Protus 2.0

An instance of the *BehaviourPattern* class presents specific type of *NavigationalSequence* class. Former consists of series of resources that learner had interacted with.

Table 4. Example of instance of *Condition* class

Property description	Property Name	Property value	Property type
Condition's id	hasId	I006	Datatype property
Learning style of learner	generatedBy	LS03	Object property
Learning style Category	hasLearningStyleCategory	visual	Datatype property
Learning style domain	hasLearningStyleDomain	Information Reception	Datatype property
Learner's performance	generatedBy	P01	Object property
Personalization	Generates	PR09	Object property

For example, an instance of the class *BehaviourPattern* contains the information presented in Table 5.

This particular instance of *BehaviourPattern* class is type of Navigational sequence marked as NS02, that learner made during session S1012, with

rating of 0.37 and it generates instance of *Personalization* class marked as PR09.

All personalization activities within Protus 2.0 are performed based on previously mentioned data in instances of *Condition* and *BehaviourPattern* data. Section 4 will present a few examples of performed personalization based on all collected data about learner's interaction with the system.

Table 5. Example of instance of *BehaviourPattern* class

Property description	Property name	Property value	Property type
BehaviourPattern's id	hasId	BP0016	Datatype property
Navigational Sequence	isTypeOf	NS02	Object property
Session	partOf	S1012	Datatype property
Personalization	generate	PR09	Object property
Ratings of navigational sequence	hasRate	0,37	Datatype property

4. Personalization in Protus 2.0

Protus 2.0 system offers two types of personalization to each individual learner, personalization based on learning styles of learners and personalization based on mining the frequent sequencing.

When learners start their interaction with Protus 2.0, the first step is to cluster learners based on their learning style. Behavioural patterns are discovered for each learner by AprioriAll algorithm, based on chosen options during learning (details about performed algorithm in Protus are presented in [20]). Recommendation list of material has to be presented to learner is created according to the data from the *Learner model ontology* and ratings of the frequent sequences, provided by the Protus 2.0 system. All recommendation actions are done in real-time during learning sessions. The provided recommendation is expected to have a higher accuracy in matching learners' requirements to learning material and thus a higher level of acceptance by the learners.

Queries over ontology data has been performed using Sparql – query language for RDF [33] and Semantic Web rule language - SWRL has been used for rule-based reasoning [35]. Details about implemented adaptation rules in Protus 2.0 are presented in [38], [39]

4.1. Learner's interaction with Protus 2.0

In this section, we will explain recommendation procedures for a new learner. The new learner signs up by using the registration form in order to create an initial personal profile and update the *Learning model ontology*. Each profile stores personal information supplied directly by the learner, i.e.: last name, first name, login, previous knowledge, preferences, etc. (known as static information), and information about learning style, current progress, and behaviour (known as dynamic information). When learners are already registered to the system, their learning styles need to be tested. The learner has to fill-in a questionnaire that is used to determine his/her preferred learning style [21]. The learning style of the student indicates a preference for some presentation methods over others. These results are stored in the *Learner model ontology* (Figure 6), which will be used for the initial adaptation in Protus 2.0.

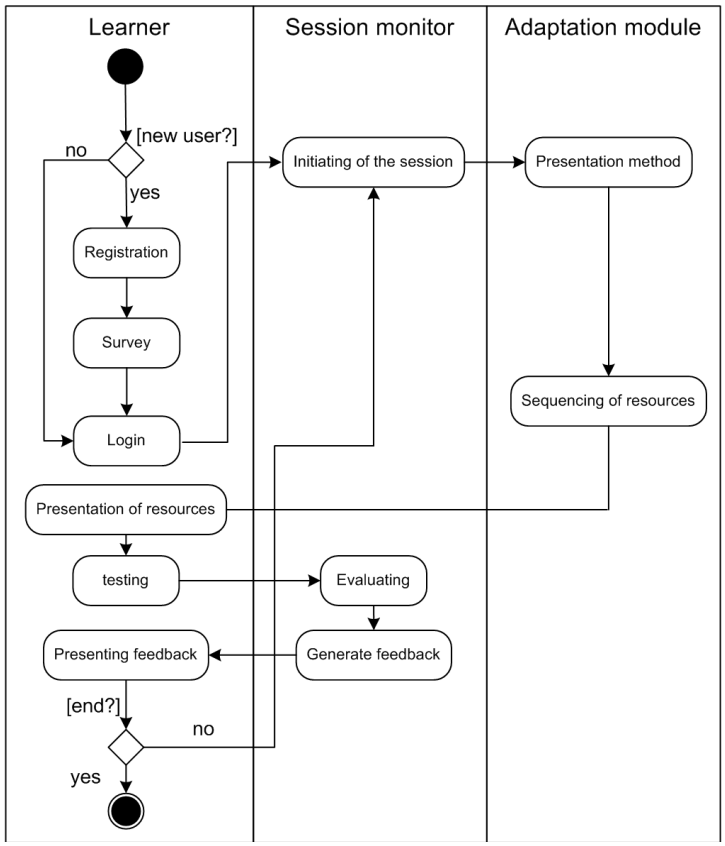

Fig. 9. Learning process in Protus 2.0

When a learner (nevertheless, he/she is new or the old one) is logged in, a session is initiated based on the learner's specific learning style already

stored in *Learning model ontology*, and sequence of lessons is recommended to him/her (Figure 9). The learner can freely change the order of lessons he/she is attending. After selecting a lesson, from the collection of lessons available in Protus 2.0, the system chooses a presentation method for the lesson based on the learners' preferred learning style. For the rest of the lesson, learners can freely switch among presentation methods using the media experience bar. That option was provided because initial assessment of the user's learning styles can be misleading in overall process of personalisation. When the learner completes the sequence of learning materials, the system evaluates the learner's knowledge degree for each lesson. The test contains several multiple-choice questions and code completion tasks. Protus 2.0 then provides feedback to the learner on his/her answers and gives the correct solutions after the learner finished the test.

Recommendations cannot be made for the whole pool of learners, because even for learners with similar learning interests, their ability to solve a task can vary due to variations in their knowledge level. In this approach, we perform a data clustering technique as a first step to cluster learners based on their learning styles. These clusters are used to identify coherent choices in frequent sequences of learning activities. Then, a recommendation list can be created according to the ratings of these frequent sequences provided by the Protus 2.0 system. The details of the whole process are presented in the rest of the paper.

4.2. Learning styles

It is obvious that different learners have different preferences, needs and approaches to learning. Psychologists call these individual differences learning styles. Therefore, it is very important to accommodate for the different styles of learners through learning environments that they prefer and find more efficient. Learning styles can be defined as unique manners in which learners begin to concentrate on, process, absorb, and retain new and difficult information [11].

There are over seventy identifiable approaches to investigate and/or describe learning style preferences. We used one such data collection instrument, called Index of Learning Styles (ILS) [12]. The ILS is a 44-question, freely available, multiple-choice learning styles instrument, which assesses variations in individual learning style preferences across four dimensions or domains [22]. These are *Information Processing*, *Information Reception*, *Information Perception* and *Information Understanding*. Within each of the four domains of the ILS there are two categories:

- *Information Processing*: Active and Reflective learners,
- *Information Perception*: Sensing and Intuitive learners,
- *Information Reception*: Visual and Verbal learners,
- *Information Understanding*: Sequential and Global learners.

The preferred learning style can be investigated by offering the learner a free choice between an example, an activity or an explanation at first, and by observing a pattern in the choices, he/she makes [22].

At the beginning of the session Protus 2.0 requests information about the status of the course from the *Learner model ontology* for the particular learner (Figure 10). This data includes information about the current lesson and the learning style category of the learner within one of the four domains of the ILS (Figure 6). Request for appropriate resources which will be presented to the learner, based on this data, is sent to the *Application module*. Further, all activities of learners are monitored, as well as all requests he/she send to the system.

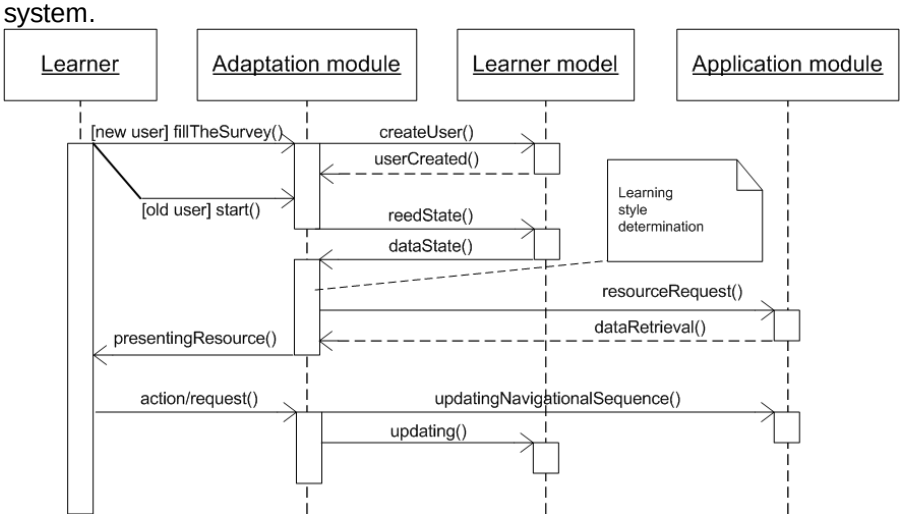

Fig. 10. Adaptation based on the learning styles

If the learner successfully learned a specific concept (Table 1), which is supported with a specific resource (Table 2), then the system should add that resource and details about performance to the successful navigational pattern (Figure 8). In addition, if the learner interacts with a specific resource and during that interaction he/she took the test and earned a specific grade, then the system should memorize that learner's performance (Table 3) and mark that resource as learned (Table 2).

If the learner does not provide the required level of performance results within a session based on the presentation method used for his/her certain learning style category, his/her current learning style category will be modified. In those cases, the system changes the current learning style of the learner to its alternative, from the same domain. Learning styles are grouped in pairs (active and reflective, sensing and intuitive, visual and verbal, sequential and global), therefore every learning style has only one alternative within the domain.

For example, if a learner with *Verbal* learning style interacts with the system and during that interaction he/she had accessed appropriate concept but not earned sufficient grade (required grade level is kept in global value

required), then, the learning style of that learner should be changed to its alternative from Information reception domain: *Visual* learning style. That implies that in the next session, the learner will be presented with resources that are defined to support a particular learning style category.

4.3. Navigational patterns

Resource sequencing is a well-established technology in the field of application of intelligent tutoring systems in educational processes [4]. The idea of resource sequencing is to generate a personalized course for each learner by dynamically selecting the most optimal teaching actions, presentation, examples, task or problems at any given moment. By optimal teaching action, it is considered an operation that in the context of other available operations brings the learner closest to the ultimate learning goal. Most often, the goal is to learn and acquire some knowledge up to a specific level in an optimal amount of time. Learners could follow different paths based on their preferences and generate a variety of learning activities. All these variations in series of learning activities are recorded by the Protus 2.0 system.

In order to monitor learner's performance during the session, Protus 2.0 records results of learner's interaction, earned grades and data about used concepts (navigation through resources). These results are used in building a global database of navigational patterns.

When the learner completes the sequence of learning materials, the Protus 2.0 system evaluates the learner's acquired knowledge [21]. The learners' grades can be interpreted according to the percentage of correct answers. Two learners are said to be similar to each other if they are evaluated by the system with the same ratings for a similar navigational sequence. Ratings of frequent sequences are not calculated only by followed sequences itself but earned grades throughout session are also included in calculation. Therefore, every system-imposed path still counts towards placing the learner in a particular cluster. Recommendation process can be carried out according to these learning sequences based on the collaborative filtering approach that is described in our previous work [22], [40]. Here, one practical example will be presented.

After each request or action of learner, the system examines the current resource sequence and makes comparison with navigational patterns of previous users (Figure 11). Protus 2.0 finds similar users and creates a recommendation list according to the ratings of the frequent sequences.

For example if a specific navigational sequence of resources has been recommended to the learner, then the system should recommend to him/her the next specific resource that belongs to that navigational sequence. Recommendation status of that resource is set to true (Table 2), therefore one of the several changes in user interface must be made (depending on the included resource type):

- link to that resource is annotated or highlighted (Figure 12a).

- the interface elements for sequential navigation will be hidden, giving the learner possibility to freely jump through the courseware (in a case of sequential learning style category) or presented (in a case of sequential learning style category, Figure 12b).
- additional tabbed pane elements will be added to related or more complex content to help situate the learnt subject and contribute in creating clear overall view on the subject being thought (Figure 12c).

Details of SWRL rules that retrieve data from Protus 2.0 ontology and make appropriate decisions for performed adaptation are presented in [38], [39].

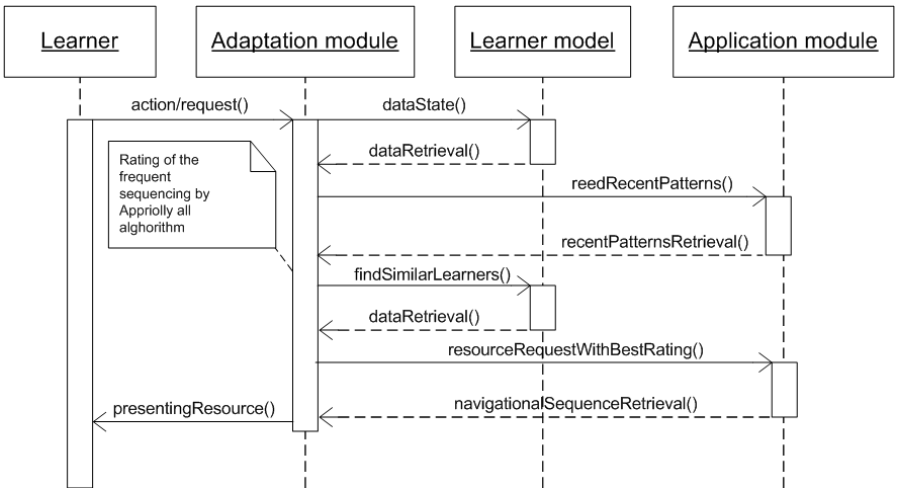

Fig. 11. Adaptation based on the navigation pattern

a. Annotation b. Presented interface elements c. Adding elements

Fig. 12. User interface adaptation

4.4. Final Remarks

The architecture for a tutoring system supported by several ontologies was presented as a way of addressing the problems of maintenance and reuse of components of the learning system. The implemented architecture extends

the use of ontology, where the representation of each component is made by a specific ontology. This way, it is possible to:

- promote a clear separation of concerns of the tutoring system components
- make explicit the communication among the components
- specify the interface for easy updating of the components
- emphasize the gains of the use of Semantic Web in the development of tutoring systems.

Ontology-based architecture also provides better interoperability and reusability of systems components in the future that will allow implementing additional programming courses as well as courses from other domains.

Preparation stage for building ontology-based architecture of Protus 2.0 utilized the main disciplines of knowledge engineering [6]. The four preparation steps were:

1. developing an ontological knowledge model by gathering expertise related to learning objects and content sequencing
2. representing the domain model using OWL ontology
3. establishing individual relationships and adaptation rules using SWRL and
4. asserting factual knowledge based on the ontology schema.

The above processes provide the basis for building an initial ontological knowledge base. Both description logic reasoning and the rules engine were further applied to extend the inference power to establish a runtime knowledge base. In our previous works, we presented SWRL rules for altering pattern navigation [38, 42] and rules for building learner model [39, 42] in Protus 2.0.

These ontologies may serve as a foundation that can be extended and modified to present the ontology for other adaptive systems.

The use of ontologies to model the knowledge in Protus 2.0 system represents a key aspect for the integration of various information that are presented to the learner, for supporting collaboration within learning community, for improving information retrieval, and more generally, it is important for reasoning on available knowledge. In Protus 2.0, ontologies are used to model educational domains and to build, organize and update specific learning resources (i.e. learning objects, learner profiles, learning paths, etc.). Ontology represents the terminology of a programming domain, defining its essential knowledge. Ontologies in Protus 2.0 will also be used to support semantic search, making possible to query multiple repositories and discover associations between learning objects that are not directly understandable.

This Semantic web approach allows implementation of adaptation customized to different requirements. The learner demand is derived from the knowledge contained in the ontology. Various conditions in Protus 2.1 are captured in the body of SWRL rules that are described in [42]. As a result of the firing of rules, recommendations in the form of various content presentations are generated, which can be used to implement the concept of adapted content and adapted navigation.

Practically, Protus 2.0 do not offer any kind of improvements regarded to personalization process. This process is the same as in the previous version of Protus. Architecture supported by the Semantic Web Technologies has been proposed in this paper in order to implement new, ontology based architecture, while evaluation of the performed personalization process has been described in detail in [42]. Learners were basically satisfied with previous version, therefore similar results are expected with this new version of the system. Protus 2.0 is in the process of implementation and results of the use of the system in the classroom will be the subject of our future work.

5. Conclusion

In this paper we proposed how Semantic Web technologies and in particular ontologies can be used for improving functionalities of an existing Java tutoring system. The architecture for such an adaptive and personalized tutoring system that completely relies on Semantic Web standards and technologies has been presented. The form of several ontologies has been proposed which correspond to the components of a tutoring system.

Ontologies will fundamentally change the way in which systems are constructed. Today, knowledge bases are still built with little sharing or reusing - almost each one is built from scratch. In the future, intelligent systems developers will have libraries of ontologies at their disposals. Rather than building from scratch, they will assemble knowledge bases from the libraries. This should greatly decrease development time while improving the robustness and reliability of the resulting knowledge bases.

The explicit conceptualization of system components in a form of ontologies facilitates knowledge sharing, knowledge reusing, communication and collaboration among system components, and construction of intensive and expressive systems. Ontologies are being more and more widely used for constructing systems that require an explicitly encoded knowledge. Therefore, those systems are able to interoperate better providing learners with an extended access to information resources.

The proposed architecture is modular, which allows higher flexibility and future replacement of various components as long as they comply with the current interface.

Previous version of Protus system was intensively tested and several experiments were carried out in order to show suitability of using recommendation technics for suggesting online learning activities to learners based on their learning style, knowledge and preferences. The plan is to incorporate Protus 2.0 in traditional programming courses. With this new Semantic web, supported tutoring system higher scalability and easier maintenance of the system are expected.

Although ontologies have a set of basic implicit reasoning mechanisms derived from the description logic, which they are typically based on (such as classification, instance checking, etc.), they need rules to make further

inferences and to express relations that cannot be represented by ontological reasoning. Thus, ontologies require a rule system to derive/use further information that cannot be captured by them, and rule systems require ontologies in order to have a shared definition of the concepts and relations mentioned in the rules. For the future work, we plan to present complete set of personalization rules that will allow reasoning on the instances of ontologies.

Acknowledgments. This paper is part of the research project Infrastructure for Technology enhanced Learning in Serbia supported by the Ministry of Science, Technologies, and Development of the Republic of Serbia [Project No. III47003].

References

1. Aroyo, L., Mizoguchi, R.: Process-aware Authoring of Web-based Educational Systems. In Proceedings of the International Workshop on Semantic Web and Web-based Education SWWL, Velden, Austria, 212-221. (2003)
2. Bodroža-Pantić, O., Matić-Kekić S., Jakovljev, B., Marković, Đ.: On MTE-model of mathematics Teaching: Studying the Problem Related to a Plane Division Using the MTE-model, *International Journal of Mathematical Education in Science and Technology* 39(2):197-213. (2008)
3. Bork, A.: Tutorial Learning for the New Century. Journal of Science Education and Technology, Vol. 10, No. 1, 57-71. (2001)
4. Brusilovsky, P., Vassileva, J.: Course sequencing techniques for large-scale webbased Education. International Journal of Continuing Engineering Education and Lifelong Learning 2003 - Vol. 13, No.1/2, 75-94. (2003)
5. Chen, C. M.: Ontology-Based Concept Map for Planning a Personalized Learning Path. British Journal of Educational Technology, Blackwell publishing, 1-31. (2008)
6. Chi, Y.L.: Ontology-based curriculum content sequencing system with semantic rules. Expert Systems with Applications Vol. 36, 7838–7847. (2009)
7. De Bra, P., Aroyo, L., Chepegin, V.: The next big thing: adaptive Web-based systems. Journal of Digital Information 5, Article No. 247. (2004)
8. Dehors, S., Faron-Zucker, C.: QBLS: A semantic Web Based Learning System. In Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications, 795-802. (2006)
9. Devedžić, V.: Semantic Web and Education. Springer Science, New York; (2006)
10. Dolog, P., Nejdl, W.: SemanticWeb Technologies for the Adaptive Web. The Adaptive Web, LNCS 4321, Springer-Verlag Berlin Heidelberg, 697–719. (2007)
11. Dunn, R., Dunn, K., Freeley, M. E.: Practical applications of the research: responding to students' learning styles-step one. Illinois State Research and Development Journal, Vol. 21, No. 1, 1–21. (1984)
12. Felder, R. M., Soloman, B.A.: Index of learning styles questionnaire. Retrieved 17 December 2009, from http://www.engr.ncsu.edu/learningstyles/ilsweb.html, (1996)
13. Gascueña, J. M., Fernández-Caballero, A., González, P.: Domain Ontology for Personalized E-Learning in Educational Systems. In Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies, 456-458. (2006)

14. Hee Lee, C., Hyun Seu, J., Evens, M. W.: Building an Ontology for CIRCSIM-Tutor. In Proceedings of the 13th Midwest AI and Cognitive Science Society Conference, Chicago, 161–168. (2002)
15. Henze, N., Dolog, P. Hejdl, W.: Reasoning and Ontologies for Personalized E-Learning in the Semantic Web. Educational Technology & Society, Vol. 7, No. 4, 82-97. (2004)
16. Ivanović, M., Bađonski, M., Budimac, M., Pešović, D., Programski jezik Java, PMF, Depatrman za matematiku i informatiku, Novi Sad, 2005
17. Ivanović, M., Pribela, I., Vesin, B., Budimac, Z.: Multifunctional Environment For E-Learning Purposes. Novi Sad Journal of Mathematics, NSJOM, Vol. 38, No. 2, 153-170. (2008)
18. Jacinto, A.S., Parente de Oliveira, J. M.: An ontology-based architecture for Intelligent Tutoring System. Interdisciplinary Studies in Computer Science Vol. 19, No. 1, 25-35. (2008)
19. Jovanović, J., Rao, R., Gašević, D., Devedžić V., Hatala, M.: Ontological Framework for Educational Feedback. In Proceedings of the SWEL Workshop of Ontologies and Semantic Web Services for IES, AIED 54-64. (2007)
20. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: Integration of Recommendations into Java Tutoring System. In Proceedings of the 4th International Conference on Information Technology ICIT 2009 Jordan, Paper no 154. (2009)
21. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: Integration of recommendations and adaptive hypermedia into Java tutoring system. Computer Science and Information Systems, Vol. 8, No. 1, 211-224. (2011)
22. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: E-Learning Personalization Based on Hybrid Recommendation Strategy and Learning Style identification. Computers & Education Vol. 56, 885-899. (2011)
23. Lee, M.C., Yen Ye, D., Wang, T.I.: Java Learning Object Ontology. In Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies, 538-542. (2005)
24. Merino, P. J. M., Kloos, C.D.: An Architecture for Combining Semantic Web Tehniques with Intelligent Tutoring Systems. ITS, Springer-Verlag Berlin Heideberg, 540-550. (2008)
25. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome AI-ED Problems. International Journal of Artificial Intelligence in Education, Vol. 11, No. 2, 107-121. (2000)
26. Mizoguchi, R., Hayashi, Y. Bourdeau, J.: Inside Theory-Aware and Standards-Compliant Authoring System. SWEL Workshop of Ontologies and Semantic Web Services for IES, AIED, 1-18. (2007)
27. OWL Web Ontology Language Reference. In: Dean, M., Schreiber, G. (eds.) W3C Recommendation, Retrieved February 10th 2004, http://www.w3.org/TR/2004/REC-owl-ref-20040210/
28. Protégé http://protege.stanford.edu/, Retrieved December 19th 2011
29. Razmerita, L., Nabeth, T., Angehrn, A., Roda, C.: InCA: An Intelligent Cognitive Agent-Based Framework For Adaptive And Interactive Learning, Cognition and Exploratory Learning in Digital Age. In Proceedings of the IADIS International Conference, Lisbon, Portugal, 373-382. (2004)
30. SCORM 2004 3rd Edition - Overview, 16 November 2006, Advanced Distributed Learning. http://www.adlnet.gov.
31. Rodríguez-González, A., Torres-Niño, J., Jimenez-Domingo, E., Gomez-Berbis, J. M., Alor-Hernandez, G.: AKNOBAS: A Knowledge-based Segmentation

Recommender System based on Intelligent Data Mining Techniques. Computer Science and Information Systems, Vol. 9, No. 2, 713-740. (2012)
32. Sosnovsky, S., Mitrovic, A., Lee, D.H., Brusilovsky, P., Yudelson, M., Brusilovsky, V.: Towards Integration of Adaptive Educational Systems: Mapping Domain Models to Ontologies. In Proceedings of the 6th International Workshop on Ontologies and Semantic Web for E-Learning at ITS 2008, Montreal, Canada, 60-64. (2008)
33. Sparql – query language for RDF, www.w3.org/TR/rdf-sparql-query/
34. Swartout, W., Tate, A.: Ontologies. Intelligent Systems and their Applications, IEEE, Vol. 14, No. 1, 18-19. (1999)
35. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Retrieved 19.12.08 from http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/
36. Ullrich, C.: Description of an Instructional Ontology and its Application in Web Services for education. In Poster Proceedings of the 3rd International Semantic Web Conference, 93-94. (2004)
37. Vesin, B., Ivanović, M.: Modern Educational Tools. In Proceedings of the PRIM2004, 16th Conference on Applied Mathematics, Budva, Montenegro, 293-302. (2004)
38. Vesin, B., Ivanović, M., Klašnja-Milićević, A., Budimac, Z.: Rule-based Reasoning for Altering Pattern Navigation in Programming Tutoring System. In the proceedings of the 15th international conference on System theory, control and computing, October 14-16, Sinaia, Romania, 644-649. (2011)
39. Vesin B., Ivanović M., Klašnja-Milićević A., Budimac Z.: Rule-based Reasoning for Building Learner Model in Programming Tutoring System, *H. Leung et al. (Eds.): ICWL 2011, LNCS 7048,* Springer, Heidelberg, 154--163. (2011)
40. Vesin, B., Ivanović, M., Budimac, Z.: Learning Management System for Programming in Java. Annales Universitatis Scientiarum De Rolando Eötvös Nominatae, Sectio Computatorica, vol. 31, 75-92. (2009)
41. Vesin, B., Ivanović, M., Budimac, Z., Pribela, I.: MILE - Multifunctional Integrated Learning Environment. In the proceedings of the IADIS Multi Conference on Computer Science and Information Systems MCCSIS'2008, Amsterdam, Netherlands, 104-108. (2008)
42. Vesin B., Ivanović M., Klašnja-Milićević A.,Budimac Z., Protus 2.0: Ontology-Based Semantic Recommendation in Programming Tutoring System*, Experts systems with application, 2012,* DOI: 10.1016/j.eswa.2012.04.052

MSc 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, Serbia. His major research interests are e-Learning and personalization in intelligent tutoring systems. He has published a number of scientific papers in the area.

MSc 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 25 referred papers on her work in national and international conferences and journals.

PhD Mirjana Ivanović since 2002 holds position of full professor at Faculty of Sciences, University of Novi Sad, Serbia. She is head of Chair of Computer Science and member of University Council for informatics. Author or co-author is, of 13 textbooks and of more than 235 research papers on multi-agent systems, e-learning and web-based learning, software engineering education, intelligent techniques (CBR, data and web mining), most of which are published in international journals and international conferences. She is/was a member of Program Committees of more than 120 international Conferences and is Editor-in-Chief of Computer Science and Information Systems Journal.

PhD Zoran Budimac since 2004 holds position of full professor at Faculty of Sciences, University of Novi Sad, Serbia. Currently, he is head of Computing laboratory. His fields of research interests involve: Educational Technologies, Agents and WFMS, Case-Based Reasoning, Programming Languages. He was principal investigator of more than 20 projects. He is author of 13 textbooks and more than 225 research papers most of which are published in international journals and international conferences. He is/was a member of Program Committees of more than 100 international Conferences and is member of Editorial Board of Computer Science and Information Systems Journal.

Received: December 31, 2011; Accepted: July 25, 2012.