IP Addressing: Problem-Based Learning Approach on Computer Networks

Aleksandar Jevremovic, Goran Shimic, Mladen Veinovic, and Nenad Ristic

Abstract—The case study presented in this paper describes the pedagogical aspects and experience gathered while using an e-learning tool named IPA-PBL. Its main purpose is to provide additional motivation for adopting theoretical principles and procedures in a computer networks course. In the proposed model, the sequencing of activities of the learning process is grouped into three phases based on educational goals. In this way, the same tool is used on several courses with different curricula. In IPA-PBL, problem-based learning (PBL) is applied as a pedagogical strategy, as well as a set of concrete methods implemented in the software. Together with the pedagogical model, specific domain ontology is designed. In this way, the learner's knowledge can be analyzed in order to collect data necessary for the dynamic adaptation of system behavior. The results collected while using IPA-PBL are compared to those obtained without using the system. Statistical analysis, together with pertaining considerations and conclusions, are also presented in the paper.

Index Terms—Computer networks, intelligent tutoring, IP addressing, ontology modelling, problem-based learning

1 Introduction

IP addressing represents one of the most important topics in computer network (CN) courses due to its fundamental role in organizing and establishing reliable computer networks. This paper describes the pedagogical aspects and experience gathered while using an e-learning tool named IPA-PBL – Web based hybrid system (problembasedlearning.org/Computer+Networks), designed for the problem-based learning (PBL) of IP addressing. In addition to knowledge of network equipment and related procedures for connecting and controlling the equipment, IP addressing also requires creativity, especially in network administration of complex and distributed organizations.

Therefore, the level of student knowledge and skills related to the CN domain strongly depends on understanding and implementing IP addressing principles and rules. In an environment with a growing need for CN designers and administrators [18] and with rapid changes in CN technology, one would expect that in contemporary CN courses transfer of knowledge is carried out in an efficient and effective way. For this purpose, varieties of software tools are designed in order to improve the clarity and interactivity of the learning process [7], [13], [23]. Dragging and dropping symbols of network equipment from the palette to the working panel (virtual networking space), making connections between equipment units (by simple movement of cursor between symbols), running the simulation and tracking the transfers of data between routers, switches and computers, as well as watching how the equipment changes and reassembles the content through a relaying process, represent some of the many scenarios that are used in such tools. Regardless of the software support, students still lack theoretical knowledge such as IP addressing concepts. Moreover, they lose motivation due to many Internet sites offering the services for calculating IP addresses based on entry data [14], [15], [29]. Unfortunately, when they face a concrete problem or task which is not stereotypical and requires creative skills and practical knowledge [17], they are incapable of tackling the problem and finding the optimal solution.

This was the main reason for developing the software tool with the purpose of providing additional motivation for learning the theoretical principles and procedures of IP addressing. The results of these efforts are presented in this paper. IPA-PBL hybrid nature is reflected in different supporting services offered to the learner: Tutoring through practical exercises, recommendations regarding to achieved results and content presented as explanations, arguments and learning material profiled in accordance with knowledge level of individual learner. Related works and the theoretical foundation of the research represent the content of the following section. The proposed solution is based on implementation of a specific pedagogical strategy. These issues are explained in the section titled Pedagogical Model. Detection of user misunderstandings depends on specially designed domain ontology. Details of its design and application are described in Section 4. The experience and results collected during several years of usage are presented in the subsequent section. The last section covers conclusions and future work.

2 Related Works

PBL has been used as a pedagogy method implemented in curricula since 1969 [24]. Over such a long period of time,

- *A. Jevremovic and M. Veinovic are with the Informatics and Computing Department, University Singidunum, Belgrade 160622, Serbia. E-mail: {ajevremovic, mveinovic}@singidunum.ac.rs.*
- *G. Shimic is with the Center for Simulations and Distance Learning, Military Academy, University of Defense, Belgrade 160622, Serbia. E-mail: gshimic@gmail.com.*
- *N. Ristic is with the Informatics and Computing Department, University Sinergija, Bijeljina 76300, Bosnia and Herzegovina. E-mail: nristic@sinergija.edu.ba.*

Manuscript received 7 July 2015; revised 27 May 2016; accepted 4 June 2016. Date of publication 21 June 2016; date of current version 20 Sept. 2017. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TLT.2016.2583432

numerous definitions and descriptions of PBL have been published in books and journals. As a method, PBL is founded by subject matter experts (SMEs) without educational psychology or cognitive science backgrounds. Therefore, in contrast to the traditional transfer of knowledge from the teacher to the learner, where facts and concepts represent the main part of the transferred content, PBL is an instructional approach in which learning happens through the process of making hypotheses and exercising deductive thinking in order to find the solution for the assigned task (problem) [4]. PBL is concerned with solving different kinds of problems under the domain at hand [5].

Initially, PBL was used in medicine long before the appearance of its software implementations. Nevertheless, PBL is as current today as it was more than 40 years ago due to its sophisticated methods and demonstrated practical results. It is not possible for the learner to become the subject matter expert without having the experience in solving realistic problems. Unfortunately, this can be expensive and sometimes even impossible. Therefore, implementations of PBL can be found in numerous intelligent tutoring systems (ITSs): SlideTutor [8] designed for PBL in dermatopathology helps the students to prove their diagnoses (hypothesis) by choosing and combining the appropriate concepts from the domain ontology; CIRCSIM-Tutor [20] represents another medical ITS designed for PBL by performing Socratic dialogs with the students to solve the problems in the domain of cardiovascular physiology. Creating meaningful problems represents one of the most important aspects of PBL pedagogy [31]. Problems appropriate to the learner's knowledge level challenge him to research accessible resources in order to find the pieces of the puzzle needed to complete the task. Mostly used in medicine, PBL is proposed to be performed as team work – combined with collaborative learning [3].

In addition to implementations in medicine, PBL is adopted in computer science (CS) as one of the most appropriate methods in teaching different types of knowledge and skills. It is used in the classrooms as a pure pedagogical method with the instructor playing the roles of facilitator and SME, while the learners split into groups and try to solve the problem by using methodologically correct approaches [26]. In such scenarios, the focus is on the collaborative work of the learners. Unfortunately, PBL takes place only during group sessions and there is no individual learning support. For this reason, various CS ITSs designed for this purpose exist. They can be considered as PBL tools because they are mostly designed for the learning of a specific matter [27]. Some are created with the purpose of learning software design. Representatives of such systems are KERMIT, used for individual learning of database conceptual design [30], and COLLECT-UML [2], used for learning UML class diagrams, which supports both individual and collaborative learning. Others are mainly focused on implementation issues – these are ITSs designed for learning programming in some targeted language. ELM ART – a Web based ITS [34] for learning the LISP programming language – represents one of the most often references thanks to its ability to adapt problems based on an episodic learner model. SQL Tutor represents an example of an ITS designed for learning SQL queries [22], which is interesting because it employs a domain model based on constraints, instead of concepts and their relations. Regardless of the differences in design and scope of the applications, all ITSs mentioned above share some common characteristics, such as dependency on well-defined domain ontology and problem types.

Apart from commercial CN courses offered by CN equipment makers (e.g., Cisco courses: CCNA, CCNP, CCIP, etc.), PBL seems to be used in university level education mainly for teaching CN topics in the second half of undergraduate studies and at the master level [19]. Judging from the publications, PBL in CN courses is more often applied as a pedagogical strategy than as a method implemented in software tools, regardless of the numerous applications mainly designed for simulating CN. It is considered as a part of the curriculum, in a blended learning approach, with group-assigned tasks and is strongly supported by a variety of software tools [11]. However, these approaches are interesting in describing the ways in which PBL should be implemented. In relation to the organization of a course, it is important to decide in which parts of the course (e.g., topics or weeks, lessons) PBL should be used. It is also significant that the balance between PBL and other learning approaches should be specified for every particular course activity. Traditionally, the complexity (scope) of problems should be increased gradually from the beginning to the end of the course.

Recommender systems are particulary mentioned as technology-enhanced learning (TEL) tools that offer specific support to learners, based on content, knowledge (ontology models), collaboration between them or hybrid solutions, implemented in different learning contexts: formal and informal learning, mobile learning, lifelong learning, etc. [35].

IPA-PBL represents a hybrid solution for supporting blended learning, and providing a user-friendly environment and useful feedback information in order to increase learning efficiency.

3 Brief Description of the Most Important Aspects of IPA-PBL

The system presented herein is a complex architecture and consists of numerous components that form the Web application which aggregates different learning resources to act as topics of the CN course. The most important topics are presented in this section.

3.1 System Architecture

IPA-PBL architecture (Fig. 1) is based on both Web-based applications and intelligent tutoring system architecture [37]. The core of the system comprises Content Composer, IPv4 Problem Generator and Pedagogy Module. Session Manager represents the system front end, which uses Session Tracker to record all of the user actions during the learning sessions (LRD – learner's row data) on the one hand and deliver the learning content on demand to the learner on the other. Therefore the interaction between learner and system is considered as two separate streams. Different mutual synchronized processes are used for these two purposes: Single-direction LRD streaming and bidirectional request–response-based content deliveery (see Section 3.5 for details).

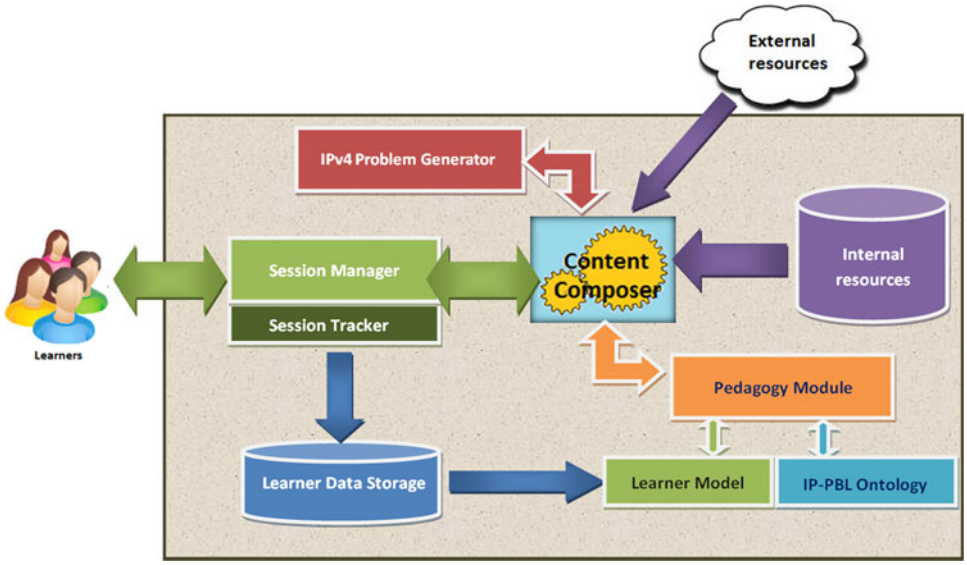

Fig. 1. Overall system architecture.

During exploitation, the IPA-PBL system stores LRDs sent from clients in the Learner Data Store (LDS). These data are used for making conclusions about a particular learner. This information is used for updating the Learner Model and further for adapting the learning content for delivering to the learner (see Section 3.4). Content Composer performs the dynamic content composition based on instructions given by Pedagogy Module. In this way the system improves interactivity with the learner while the session data are better fragmented; this represents a main difference regarding the way the learner models are commonly used in ITS. Each time the learner requests a resource the system checks whether there is any change in LRD, i.e., is there any new event that the learner generated from the last content delivery?

3.2 Pedagogical Model

PBL as a learner-centred approach is strongly based on the implemented pedagogical model [9]. The facts mentioned in previous sections lead to the conclusion that students have a great opportunity to master the subject matter easily and become CN experts in a reasonable time. Nevertheless, the achievement of such an objective depends also on their motivation. Only a well-designed pedagogical model can attract students to use PBL resources in the right way from the very beginning of the course and keep them interested through the sequence of lessons as well as motivate them to give their maximum in the exams. Therefore, the pedagogical model represents one of the most important parts of the proposed solution.

Based on previous experience gathered from teaching CN courses over several years, as well as using PBL as a pedagogical approach in a domain similar to CN [28], the IPA-PBL pedagogical model is designed to provide self-paced individual learning based on a blended approach. 'Blended' refers to the ability of the learner to use IPA-PBL simultaneously with other learning content represented in the form of combined hyper-linked documents and external resources (e.g., CN simulators or multimedia content from YouTube). In the proposed model, the sequencing of activities of the learning process is grouped into three phases based on educational goals. In this way, the same tool is used on several courses with different curricula. Moreover, IPA-PBL is used as an additional tool in the Cisco Academy courses that are offered separately from regular faculty studies at Singidunum University and Military Academy in Serbia. Starting with recognition, reproduction and simple calculations in the first phase, the system leads the learner through activities progressively enabling him to be more creative and to solve more complex problems.

IPA-PBL provides the learner with the opportunity to attend the IP addressing course through as many sessions as he wants. Moreover, thanks to the blended mode of learning, it enables the motivated learner who has already passed the exam to improve his particular skills by making new attempts and using some of the resources offered by IPA-PBL. Motivated by game theory and game-based learning, in which competition between the players (learners) is used as a motivation factor [6], in IPA-PBL the learner's results are visualized and put together with the average and maximum scores of others. In this way, the learner can make comparisons and become motivated to invest more effort into improving his skills and knowledge. The section Considerations of Results presents the details of the implemented pedagogical model.

As a main pedagogical approach and motivation factor, PBL is carefully implemented in the system (Fig. 1). Problem solving represents the most interactive part in communication between the system and the learner. The problems are graded on seven levels based on complexity (e.g., BinaryDecimalConversions represents 3 level 0 activity, while SolvingIPAddressDistributionInGivenNetwork is the most complex – level 6). Apart from this categorization, problems are also grouped according to educational goals: In the basic phase there are four levels of problems (0–3), the Intermediate phase covers the next two levels (4, 5) and in the advanced phase the learners have to solve the highest level problems (6).

The basic phase starts with the very basic pre-test whose purpose is to determine whether or not the learner knows

how to perform decimal– binary conversion and vice versa. Such a skill is not part of the IP addressing course and it is expected that learners have already mastered these conversions. Nevertheless, BinaryDecimalConversions represents the first learning activity which could help the learners to refresh their knowledge and become reacquainted with the conversion procedures. By reading the text explanation, observing the example and performing a few test tasks afterwards, the learner gradually masters conversions. On the other hand, this activity is optional for those learners who already have such skills (those who have passed the pre-test). Other activities in this phase cover basic knowledge about IP addressing: IP address classes and basic network parameters. Regardless of specialization, IT students should know how to set up their computers in order to connect them to the network, or to recognize to which class the concrete IP address belongs. The tasks that are delivered to the learners are adapted to the activities. After the self-test activity, the system evaluates the learner's results and updates his profile. If some misunderstanding is identified, the learner is navigated back to the appropriate activity. The IPA-PBL problem generator automatically creates a new problem every time the user tries to improve his actual result. In this way, it is impossible for the learner to remember the correct answers and use them in the next attempt.

The Intermediate phase is designed to train the learners in the fast checking of IP addresses and mutual visibility among the computers in the same network (level 4 and five tasks). By achieving such goals, the learners can supervise existing networks independently. The misunderstandings registered in activities of this phase do not depend on activities of the previous phase. Therefore, loop-backs are possible only within this phase.

The purpose of the Advanced phase is to make learners capable of solving the distribution of IP addresses in the given network. There is only one activity which includes solving the most complex tasks that aggregate all of the previous knowledge delivered by IPA-PBL. Therefore, the learner solution is decomposed and analysed part by part depending on the IP addressing conceptual model specially designed for this purpose. Depending on the detected misunderstandings (if any), the learner is navigated back to some of the previous activities in the first or second phase. If the learner passes the final test, he is capable of organizing the computer network according to the infrastructure constraints and organizational structure of a hypothetical company.

3.3 Ontology Design and Usage

Despite the first appearances of the term "ontology" dating back to old century (Aristotle's Organon), its usage only intensified from the early nineties up to now due to the rapid development of computer science. Among many appropriate definitions, one is "Computational ontologies are a means to formally model the structure of a system, i.e., relevant entities and relations that emerge from its observation, and which are useful to our purposes" [36].

In the presented system, the purpose of ontology is to provide a flexible way of analysing the learner's knowledge of particular IP addressing concepts and their relations. More precisely, it should support making diagnoses (through finding misunderstandings) and updating the learner's profile in order to provide appropriate tutoring progressively through the learning sessions. Therefore, the student model is built in accordance with a used ontology design and represents its enhancement [10]. Although this is an interesting part of the system, considerations relating to it are beyond the scope of this paper.

Even though there are many CN ontology models described in the literature, the appropriate one has not yet been formulated. Most models focus on network security issues [32], [33], extended to categorization and a description of the network equipment [1], or just the architecture of the basic concepts [16]. Also, there are complex CN models that cover several CN aspects. They are represented as collections of particular models designed for specific demands. The common information model (CIM) proposed by the Distributed Management Task Force (http://www.dmtf.org) is one of the most complete [25]. Although the CIM model contains the concepts related to IP addressing, they are distributed in several sub-models and some concepts necessary for learning IP addressing are missing. Regardless of their complexity and completeness, a common characteristic of existing CN ontology models is that they focus on particular matters mentioned above.

Therefore, a new ontology model is designed for IPA-PBL (Fig. 2). This model contains exactly the concepts that are necessary for using the system. The relations among the concepts reflect the way in which they are used for diagnostic purposes. In other words, the IPA-PBL ontology model is also strongly related to the pedagogy model. More precisely, the concepts are used in the process of automatic problem generation. They are considered to be the key terms of the learning content and are visualized by using different colors and level labels. Some concepts are used in just one task level, whereas others are used in two or more levels (for clarity, the zones of levels 4–6 are not shown). Next, we provide brief descriptions of the concepts and their relations.

IPNetwork is the concept sitting on top of the model and represents the main purpose of IP addressing. Nevertheless, the basic ontology concept is the IPAddress. Both concepts, as well as the NetworkProperty, belong to a very abstract level. Through them, more specialized concepts are introduced. ClassfullNetwork and ClasslessNetwork represent the specializations of IPNetwork, while NetworkCapacity, NetworkAddress and BroadcastAddress are the properties (NetworkProperty) of interest.

3.4 Making Decisions

The ontology we present is designed to support the decision making in the learner's subsequent actions that depend on his actual knowledge. To provide this, the pedagogical model strongly depends on ontology concepts and their relations. More precisely, the rules for drawing conclusions about the learner compare his results with expected results, which, on the other hand, come from objects that represent concept instances. As a result, conclusions are produced in the form of a status description or decision about the action, depending on the rule type (working or finalizing rule). The higher the level of the task (under consideration), the more concepts (rules) are included in it.

Reasoning is not only focused on learner actions related to resources stored in the system. IPA-PBL is about more than

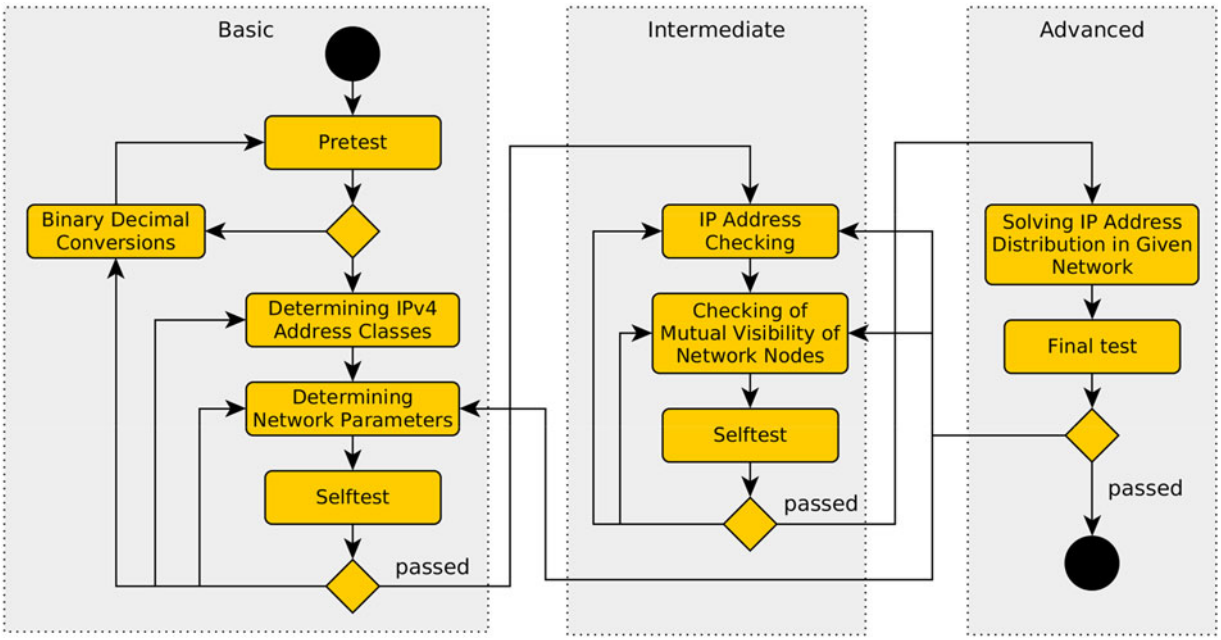

Fig. 2. Step by step learning according to Bloom's revised knowledge model.

drawing conclusions about attempts to complete concrete lesson tasks. Session Tracker provides more detailed information about learning sessions. For instance, if the learner uses other resources when completing the self-test the weight of the result will be weakened and in some circumstances the learner has to attempt this self-test again. Collecting LRDs makes this possible and the system as a whole strongly supports a learner-centric approach. Therefore, each concept is represented as a collection of parameters

$$C_{k=}\{n, d, \{(p_1^+, w_1), \ldots, (p_n^+, w_n)\}, \{(p_1^-, w_1), \ldots, (p_m^-, w_m)\}, C_r\}, \quad (1)$$

where n is the name of a concept, d represents its description, followed by an arbitrary number of parameter value–weight pairs with positive and negative effects ($p+$ and p-) and C_r denotes a collection of related concepts – relation-type pairs. Examples of positive parameters include self-test results, the number of resources used in the learning process and the number of messages exchanged with others. Negative parameters include, for instance, the number of self-test attempts, the self-test time and using other resources during the self-test. Collection C_r is represented as

$$C_{r=}\{(c_1, r\}, \ldots, \{c_q, r\}\}|r \in \{ascn, gen\}, \quad (2)$$

where c_1 and c_q are related concepts and r is a type of relation (it can be an association or a generalization). Next is an example that describes how these relationships are used for reasoning: the concepts BroadcastAddress and NetworkAddress are in is_a (gen in Eq. (2)) relationship with NetworkProperty. They are in a kind of (ascn in Eq. (2)) relationship with SpecialAddress at the same time (see Fig. 3). This means that if the learner masters both the broadcast and network addresses the system concludes that he/she has mastered all network properties. On the other hand, if the learner masters the two most important special addresses then he/she obtains the necessary preconditions to learn other ones.

IPA-PBL calculates the learner status regarding mastering every single concept as a difference between parameter–weight pair (see Eq. (1)) sums:

$$Stat = \sum_{i=1}^{n} p_i^+ w_i - \sum_{j=1}^{m} p_j^- w_j. \quad (3)$$

Every parameter p is simply represented by 1 or 0 (indicating whether it is included in the calculation or not). Parameters' weights are predefined in order to express their importance. Their values are in [0, 1]. For instance, the initial weight of the *self-test result* is 1 and it decreases with the rising number of attempts (i.e., $f = e^{-n}|n \geq 0$). The behaviour of time parameters (negative ones) are expressed by $1 - g(t)$ where $g(t)$ is the Gaussian function shifted right on the x-axis (t is a time value). In this way, if self-testing or learning is performed in 'reasonable' time there are no negative effects on the learner's status. On the other hand, an extremely short or long time maximizes the weight and, as a final consequence, makes the learner's status worse.

After learner's logging IPA-PBL is firing start rule for each concept in the domain model forming a set of mastered concepts:

$$(\exists c_k)(Stat(l, c_k) \geq Thrs_k) \rightarrow c_k \cup M_c, \quad (4)$$

where *Stat* stands for the calculated status of learner l regarding mastering the concept c_k; $Thrs_k$ represents a threshold – the minimum satisfied knowledge level of concept c_k; while M_c denotes a set of mastered concepts. The threshold is initially half the maximum theoretical score. After forming M_c, IPA-PBL evaluates the accessibility of related concepts forming a new set of concepts that should be learned

$$(\forall c_k \in M_c)(\forall c_q \in C_r)(Stat(l, c_q) < Thrs_q) \rightarrow c_q \cup L_c, \quad (5)$$

where c_q denotes the concept and L_c denotes a set of concepts that should be learned. For each new learner request

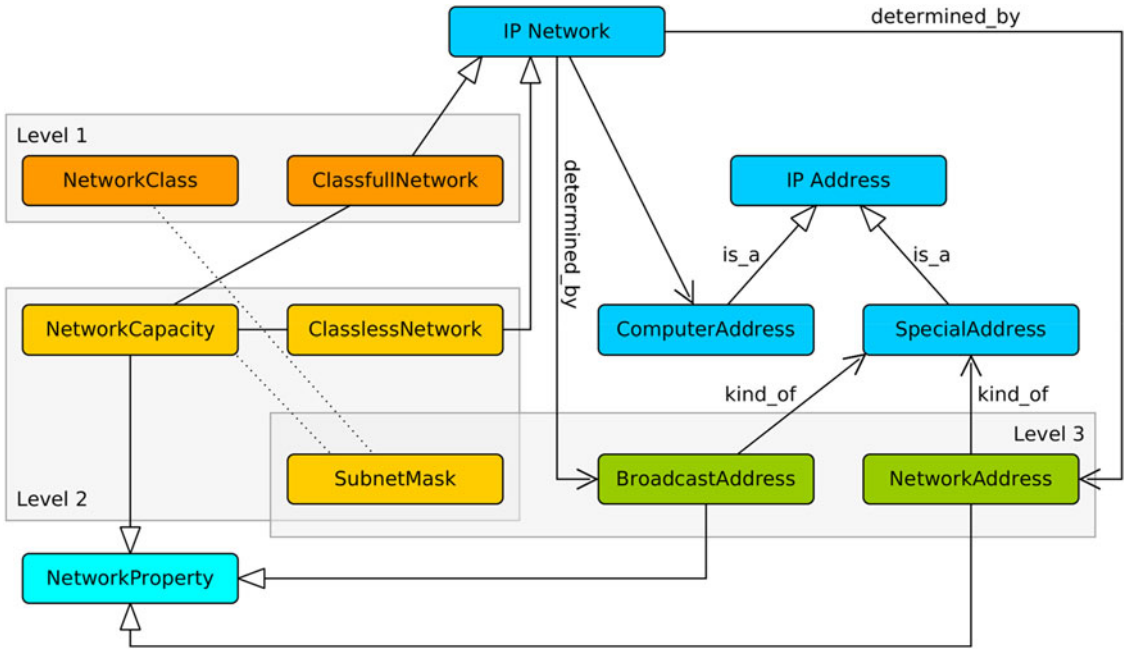

Fig. 3. Ontology model and relations of its concepts to the 'Basic' phase tasks.

IPA-PBL checks newly generated LRD(s) in order to update his/her status in the way described in Eqs. (1), (2), (3).

The expected results are defined according to the learning goals (see previous section). Binary – decimal conversion is not included as a concept (level 0 tasks) because this skill represents a precondition for using the IPA-PBL system. Due to the concrete objectives of the tasks at levels 1–3 (basic phase, Fig. 1), their relations with appropriate concepts in the ontology are presented. In this way, the simplest tasks (level 1) are related to the concepts such as NetworkClass and ClassfullNetwork, while the level 2 tasks focus on the network properties – they are related to NetworkAddress and BroadcastAddress. As mentioned, based on relations between tasks and concepts, IPA-PBL is capable of reasoning with regard to the learners' solutions. For example, the level 3 tasks are related to three concepts: NetworkCapacity, SubnetMask and ClasslessNetwork. The tasks that belong to the 'Intermediate' and 'Advanced' phases are more complex. While for level 1 tasks neither procedural knowledge nor calculations are required (the learner just needs to know the IP address classes), to perform the upper level tasks the learner needs to have knowledge of some specific NetworkProperties and IP addresses, as well as to master decimal–binary conversion based on calculations and manipulation with particular address bits.

3.5 Client Side LRD Generator

LRDs are generated on the client side by a LRD generator. It is delivered as a Java Script with the content to the Web client and consists of the following methods: event listeners (hook methods for learning events) and a transceiver, which first generates XML formatted LRDs on event listeners' demands and second sends them to the IPA-PBL server application. There are no limitations to the events that could be listened to: Clicking on any control on the Web page (e.g., command or radio buttons, check boxes, tabs, list options, links etc.), typing in text boxes, losing or setting the focus on some control, or on the whole page. The next example is of an LRD message generated during a self-test (Fig. 4). Each message has a clrd root element followed by an actions element whose attributes define a context (self-test) and identification details (lesson, question and attempt IDs).

The example shows the information about the learner's actions collected during a self-test: He/she tried to answer the multiple choice question. The first and third action elements represent clicking on the chosen answers; the second is a click on the link (the learner used other resources during the self-test). The LRD generator composes messages implementing Decorator design patterns (the actions unit is decorated with a sequence of fragments – event details in the form of action elements).

As a result, the IPA-PBL system records more details about learning sessions providing more control over the system – learner interaction. The second advantage is that learner behaviour data are fully separated from e-content;

```
<?xml version="1.0" encoding="UTF-8"?>
<clrd>
    <actions type="quanda" aid="45stdquad4sc12" >
      <action type="mouse_click" lid="45gsh4714d" >
          <object id="27451" type="list">
            <![CDATA[<option value="82.216.45.143">82.216.45.143</option>]]>
          </object>
      </action>
      <action type="mouse_click" lid="45gsh4714d" >
                <object type="link"><![CDATA[http://jodies.de/ipcalc]]></object>
      </action>
      <action type="mouse_click" lid="45gsh4714d" >
        <object id="27451" type="list">
          <![CDATA[<option value="82.216.45.0">82.216.45.0</option>]]>
        </object>
      </action>
      <action  type="mouse_click" lid="45gsh4714d" >
        <object type="submit"></object>
      </action>
    </actions>
</clrd>
```

Fig. 4. LRD message.

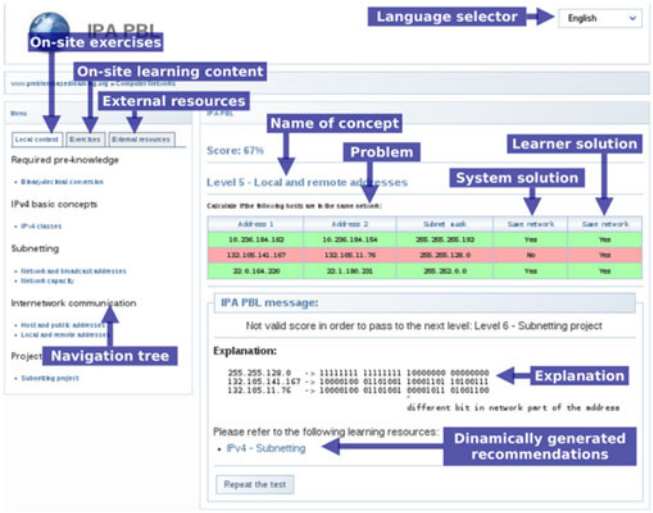

Fig. 5. Elements of user interface.

in other words, e-content can be managed as a part of particular e-course deployed on LMS (Learning Management System) or more general – as a resource hosted on CMS (Content Management System) while user data are stored in a separate system (e.g., HR system). The improved flexibility of the system represents the third advantage—The system behaviour is adaptive regarding the type of client side platform (desktop/laptop/tablet/smart phone).

4 Details of Experience Collected During IPA-PBL Exploitation and Related Improvements

The IPA-PBL user interface is designed to support a blended learning approach, as well as easy navigation through various learning resources (Fig. 5). In this way, the proposed pedagogical model (Section 3.2) is fully implemented. Exercises, local content and external resources are offered as menu tabs on the left-hand side. They are synchronized – the content shown in the main (right-hand side) panel is appropriate to the selected item (active concept) on the left-hand side. For instance, during an exercise the learner can gain easy access to corresponding internal or external learning resources just by switching tabs.

IPA-PBL feedback information has been found to be very useful to learners and it represents an additional learning resource. It consists of three parts: commented on and marked learner solution, correct (system) solution with arguments and explanation, and links to recom-mended learning resources.

Based on the analysis of the data collected from the learners' sessions (more details in the next section), we draw numerous conclusions about the learners' behaviour and implemented didactics. This information is used for system improvement. The pedagogical model remains the same, while numerous minor changes are made for this purpose. We now describe the most interesting conclusions and related improvements.

4.1 Level 0

We observed unexpected results in the level 0 tasks. In comparison to all other task levels, the results of BinaryDecimalConversion came out with the lowest score. Although performing these tasks requires strictly procedural knowledge, which is well described, their simplicity is offset by time limitations. Another reason for poor performance is that 100 percent completion was required (partially completed tasks were not accepted). In this way, many random errors were made by learners who achieved very good and excellent overall results at the end of the course.

Two improvements wereade for level 0 activities:

1. The number of tasks assigned to a particular learner progressively increases in the event of wrong answers with the purpose of providing additional practice.
2. At the same time, the tasks are delivered one by one (one per page) so as to provide better learner focus and concentration (initially, all tasks were delivered at once).

4.2 Level 1

These tasks cause minor obstructions to learners owing to the possibility of learning the exact intervals of each class (depends on the prefix bits of the left hand side byte). Nevertheless, there were noteworthy mistakes about the D (multicast) and E (for future or experimental purpose) address classes, while more than 80 percent of learners gave correct answers about the A–C address classes. Therefore, in addition to repeating the task until acceptable correctness is reached, the new improvements made for level 1 tasks include task adaptation in every new attempt to include more questions about the address classes that were determined to be problematic for a particular learner. This adaptation is made at the expense of classes with no problems detected (e.g., instead of the questions about A and B, there are questions about D and E address classes).

4.3 Levels 2 and 3

Regardless of the straightforward difference between tasks at these two levels, the mistakes made by learners were similar and based on misunderstanding of a common concept – SubnetMask (Fig. 2). Although full octet mask answers were error-free (/8, /16, or /24), the sub-net masks caused a number of wrong answers due to sub-net bits being split between the octets. To account for this, IPA-PBL is improved to provide better help for the learner; in such a case it delivers to him both the correct answer and a detailed 'how it's done' explanation. Moreover, the tasks delivered in the next attempt are adapted according to the mistakes the learner made previously. Besides the correctness of the solution, including the time spent, assessment motivates the learner to achieve not only better results, but also to perform in a shorter time.

4.4 Levels 4 and 5

Checking given IP addresses to see if they can be used for host addressing (level 4 tasks) requires various skills and knowledge from the learner. Tricky questions are also included in these tasks: Recognition of class-full addresses, network and broadcast addresses as well as knowing the ranges of special addresses. Skilful manipulation with subnet masks is required as well.

TABLE 1
T-Tests Results

	IPv4 scores				Final scores			
Generation	2010/11	2011/12	2011/12	2012/13	2010/11	2011/12	2011/12	2012/13
N	84	93	93	92	84	93	93	92
Mean	23.75	26.58	26.58	26.78	72.40	71.11	71.11	72.52
SD	7.11	4.83	4.83	3.58	14.57	10.41	10.41	8.37
SEM	0.78	0.50	0.50	0.37	1.59	1.08	1.08	0.87
SED	0.096		0.626		1.890		1.389	
DF	175		183		175		183	
t	3.1232		0.3228		0.6865		1.0180	
P value	0.0021		0.7472		0.4933		0.3100	
Significant	very		not		not		not	

As mentioned in Section 3, the level 5 tasks are about checking whether the computers belong to the same network. The pairs of concrete IP addresses are given as well as the address mask related to them. Before answering, learners need to perform calculations on every pair of host addresses.

At both levels (4 and 5) the tasks require a 'Yes/No' answer. This gave rise to certain problems in the results. In many cases (~23 percent), they are worse than those of the level 6 tasks. After a detailed analysis we concluded that 50 percent of learners overestimate their ability to answer correctly. The underlying reason is the use of approximate estimations instead of detailed calculations in order to reduce the time spent during self-tests. Such an approach produced worse results than expected.

Certain improvement in IPA-PBL has been made: If the correctness on answering the 'Yes/No' questions is less than 90 percent (a high score is required for reasons of reliability, to prevent guessing the answer) the question type is changed: a 'short answer' at level 5 and 'fill in the blanks' at level 4 are used instead of the 'Yes/No'. In this way, the learner has to perform calculations if he wants to achieve the correct answer.

4.5 Level 6

The tasks at this level are the most complex. They include all the knowledge delivered through the IPA-PBL and the results should reflect the maturity of the learner in dealing with IP addressing. The organizational structure of the company, number of computers in individual departments and IP address range are given; the learner's task is to define sub-ranges for these items which include network address, broadcast address and sub-network mask.

Learners made two types of mistakes. The minor one was missing the exact boundaries of the IP addressing ranges, usually for one or two addresses. For example, for a given network mask x.x.x.0/25 the range is x.x.x.0 – x.x.x.127; however, the learner specifies x.x.x.127 or x.x.x.128 as the IP addresses that could be used for the computers. This is wrong because 127 is reserved for broadcasting, while 128 is out of range. When such omissions occur, IPA-PBL redirects the learner back to the previous level. The more serious errors happen when the learner defines sub-ranges of IP addresses that are completely wrong. This is when IPA-PBL redirects the learner two levels back (level 4) because he is obviously confused about sub-netting by using both network masks and special addresses. If he continues to supply wrong answers at this level, IPA-PBL redirects him again – this time to level 3 activities.

In both minor and major mistake types, the main cause is incomplete understanding of the SubnetMask concept. Therefore, this concept is represented in different ways in activities at all levels except 0 and 1.

5 Analysis of Result

Generally, evaluating the effects achieved by PBL implementation in a specific course is difficult because it is used in combination with other learning approaches. Hence, it is hard to isolate the factors that influence student results and their attitudes towards PBL (O'Grady, 2012). Therefore, the results of assessments of different generations of students are used as representative data for PBL evaluation (Mitchell and Delaney, 2004).

In order to evaluate the efficiency of our IPv4 PBL solution, we compared the results of three student generations:

- generation 2010/2011 that had not used the system at all,
- generation 2011/2012 that had used the basic version of the solution (with no additional '+' tasks) and
- generation 2012/2013 that used the final version of the proposed solution.

The results of all three generations are structured in the same way: the final exam is divided into three parts. Two are based on the theory (test), while one is based on solving IPv4 problems. Each part has a maximum score of 30 points. The maximum total score is 100 points, with three portions of 30 points, and 10 points awarded for student activity during the semester. In this section, we compare the scores of different generations. More precisely, we compare the scores on the IPv4 part of the exam and also the final (total) score. The unpaired t-test with a confidence interval of 95 percent was used for comparison.

The t-test results, presented in Table 1, show a statistically significant improvement in the IPv4 results of the students using our system (generation 2011/12), compared to the results of the students who had not used the system at all (generation 2010/11). However, there is no statistically significant improvement in the results of the students who used an improved version of our system (generation 2012/13) when compared to the results of the students who have used the basic variant of our system (generation 2011/12), although the results are slightly better. An important

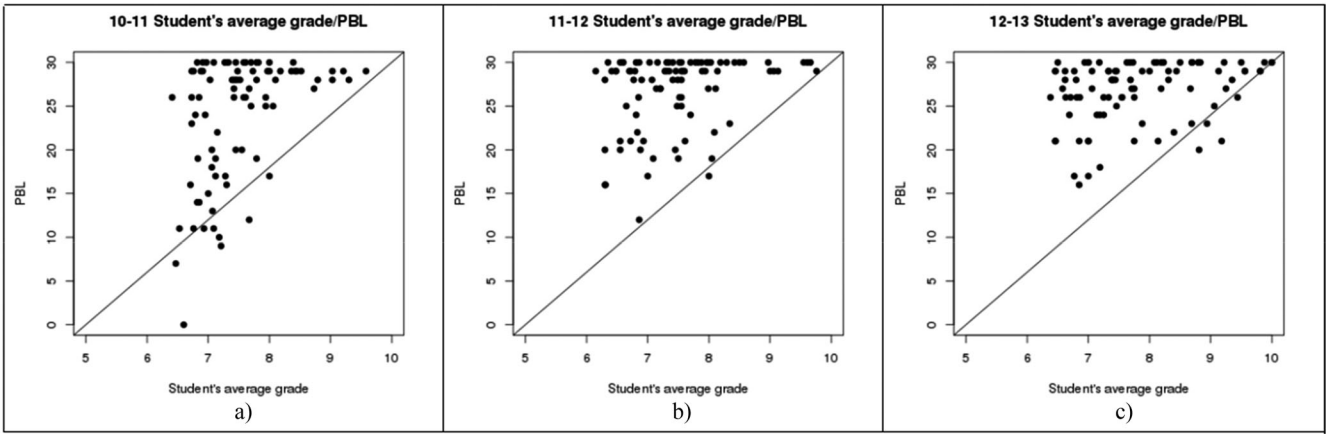

Fig. 6. Overall average grade versus PBL results.

change in all the results is the reduction in the standard deviation of the score which we interpret as a more equable distribution of knowledge and skills attributed to our system.

It is also important to notice an anomaly in the final scores of generations 2010/11 and 2011/12. Although students who have used our system (generation 2011/12) achieved a better score in the IPv4 part of the exam, the final scores of this generation were lower than the scores of the previous generation (average score in generation 2010/11 was 72.4 while the average score in generation 2011/12 was 71.11). Our understanding of this anomaly is that by using our system to prepare for the IPv4 part of the exam, students gained more confidence in their knowledge in this area, so they put less effort into preparing for other topics (that require more time-consuming preparation). However, no change in the final results was statistically significant.

To better describe the effects of using IPA-PBL, we provide plot charts (Fig. 6) that show the correlation between the learner results achieved in PBL correlated with an average grade obtained in previous courses. We observed the weakest correlation for the 2010/11 school year (case a). This was the last year in which the course was offered without IPA-PBL and the results presented are similar (+/-5 percent) to the preceding years in which the course had been taught. Interpreting that, the results were unexpected for the majority of learners. We observed a large deviation in student population with average results between 6.5 and 8. This reflects the main motive for implementing IPA-PBL in this particular course. In the first year of IPA-PBL use, the results improved significantly (case b). Less variation in the results, as well as their grouping in the desired area (upper right part of the chart) was achieved. In the 2012/13 school year (case c), better correlation between the PBL results and the average grade was achieved thanks to improving the tasks delivered to learners. In the previous year (case b), there was a significant share of high scores (>25pts) that is not complemented by medium range grades (from 6.51 – 'good' – to 8.1 – 'very good'). In both of the last two school years the presented plots converge to the upper right corner which implies the desired trend.

The next three charts are about outputs – the correlation between the final course grades and the portion of the course taught by IPA-PBL (Fig. 4) in the last three school years. These trends differ from the previous analysis. As explained above, in year 2010/11 (case a) the PBL approach without using IPA-PBL was employed and the correlation was weak since there was no dependence between the PBL results and the final scores of the learners. Despite the less scattered and more positive results after the first year of IPA-PBL use (case b), the relationship between those results and the learners' final scores is still quite weak. Similar to

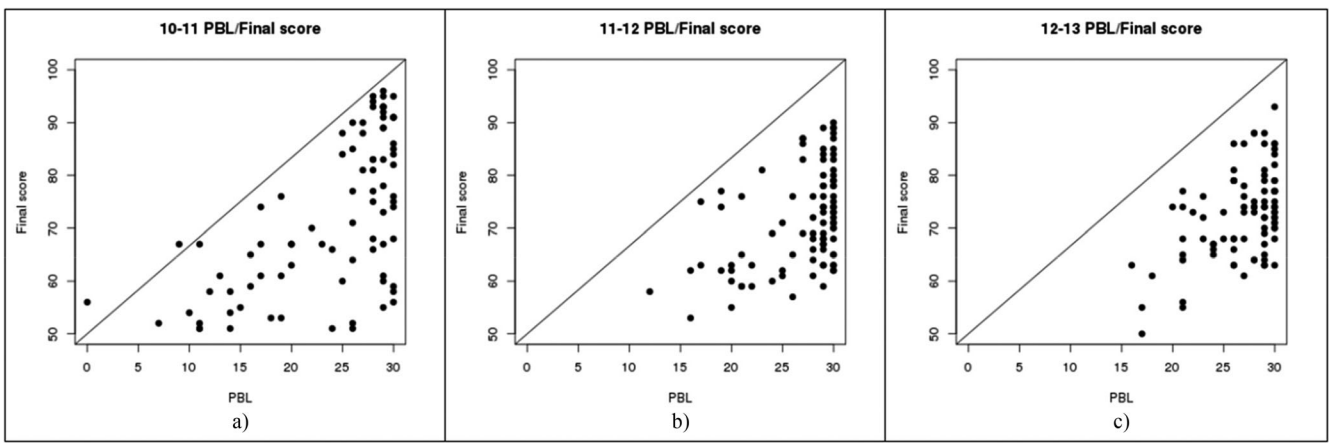

Fig. 7. Overall average grade versus PBL results.

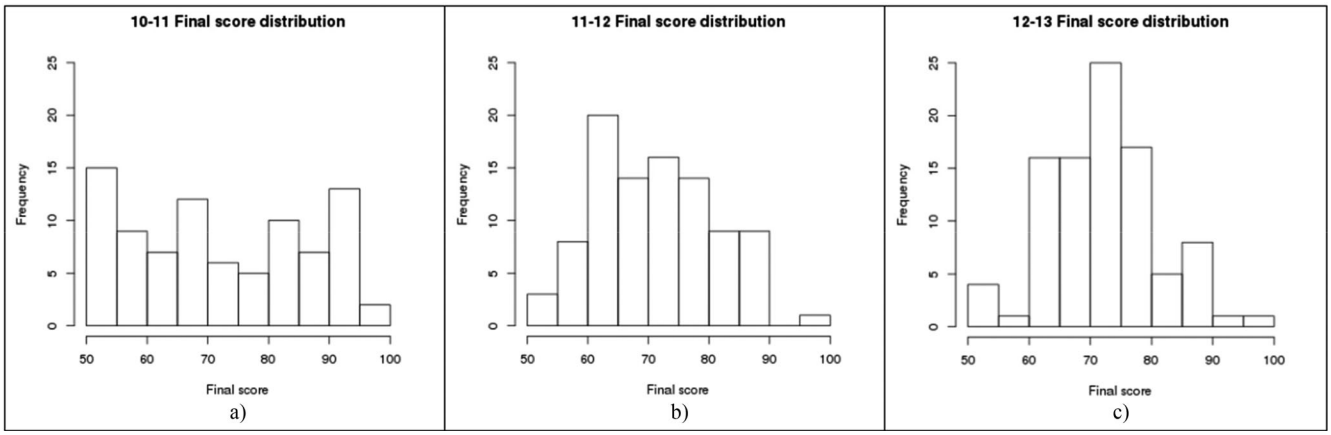

Fig. 8. Frequency distribution of results presented in Fig. 7.

the exhibit in Fig. 3, in the first year of IPA-PBL use (case b) the PBL results were much better than the final scores. The learners achieved approximately 60 percent as a final score and had PBL results that were uniformly distributed in the 20–30 range, with 30 being the maximum. Influenced by the improvement in level 5 and 6 tasks, the results in 2012/13 (case c) are less scattered, while correlation between them is better and the final scores are improved.

The next illustration (Fig. 8) shows the distribution of the final score frequencies. The x axis represents the score scale, while the y axis represents the number of learners. In 2010/11 (case a) the distribution of the results was almost uniform which was undesirable. This was largely due to the highest frequency of 50–55 percent scores, but also due to the lowest frequencies of 70–80 percent and 95–100 percent scores. In 2011/12 (case b), the distribution is significantly improved because the majority of the learners scored in the 60–80 percent range. In 2012/13, the distribution (case c) is obviously different from the previous one due to the highest frequency in the 70–75 percent range which is a change of almost 1/3 of frequencies in the 60–70 percent and 75–80 percent ranges.

Based on the considerations described above, the most important improvement is inclusion of the IPA-PBL system in 2011/12 which produced better scores in the IPv4 addressing course. The distribution diagrams (Fig. 5) reveal more conclusions. Undesirable uniform distribution in 2010/11 is significantly improved in 2011/12 owing to the use of IPA-PBL instead of just PBL. Regardless of almost the same average scores in the last two years (Table 1), the standard deviation of score distribution in 2012/13 is much better than the year before. Namely, the standard deviation is 14.567 for 2010/11, 9.322 for 2011/12, and 8.271 for 2012/13. The mode in the 70–75 percent range for the 2012/13 results, as well as the high frequencies of results in its proximity represent further evidence of the improvements achieved by using IPA-PBL.

6 Conclusions

Desire to improve learners' results in an existing CN course and the possibility of combining theory with practical problem solving during the learning process were the basic motivational factors in the design and development of the IPA-PBL system. The efficiency and effectiveness of PBL implementations described in a body of research work, as well as educational applications of PBL across a diverse set of domains, also encouraged our research efforts.

PBL, as a foundation for intelligent learner tutoring, should be supported by a sophisticated pedagogical model. The model we developed essentially represents a progressive sequence of activities complemented with loop-backs that provide dynamic adaptation of system behaviour according to the learner's actual state. To improve the estimation of preconditions, (theoretical) knowledge, and (problem solving) skills of an individual learner, the activities are separated into three phases and the pedagogical model is layered according to the tasks (problems) implemented. The learner has to undertake the self-test at the end of each phase and then the system behaviour is determined by a set of rules that navigate the learner back or forward, depending on the results. If the learner performance meets expectations, he is presented with less navigation features; otherwise a navigation-rich version is supplied.

Besides the pedagogical model, specific domain ontology is necessary for the desired intelligent behaviour. First, several already existing CN ontology models were considered. Most of them are of a pure hierarchical nature that supports only top-down concept decompositions; just a handful offer implementations of other types of concept relations. Unfortunately, such models are very complex due to their consisting of a large number of concepts. Moreover, concepts are further grouped into many sub-models. If the concepts of interest belong to different sub-models, their loading and processing can cause difficulties and a decline in software performance. Therefore, we designed a separate ontology model. It consists of three general and nine concrete domain concepts while also providing overall diagnostic support. The efficiency of such a simple and optimized model is based on relations between concepts on the one hand, and learning content and problems for solving, on the other.

As mentioned above, both models considered are necessary for the advanced behaviour of the system. They are tightly coupled regardless of different approaches in their design. The flexibility of the pedagogical model stems from the rules used to derive conclusions about the learner. Rules manipulate the objects represented by the learner results, mistakes he makes and related concepts, such as inputs and

learning resources as rule outputs. The concept relations are used to chain the rules. On the other hand, the separately developed ontology model is flexible due to the capability to add, change, and remove concepts and their relations. The changes in ontology have to be followed by updates in both rules and learning resources. Such details as well as the very process of inference are not presented herein because we believe they are beyond the scope of this paper.

The effects of using IPA-PBL are verified by comparing the course results of three generations of learners. The data of the unpaired t-test used for this purpose (Table 1) show statistically significant improvements in the results achieved by using the system. Further changes made in order to provide better interaction with the learner (Section 5) resulted in a more modest success (Figs. 3 and 4, cases b and c). The standard deviations of results and their frequency distribution (Fig. 5) indicate significant progress in both generations using the system. The uniform distribution of the results (Fig. 5, case a) registered in the first generation (without using the system) is replaced with approximately normal distribution in the last two generations (Fig. 5, cases b and c). Regardless of almost the same average results in the last two generations, the system improvements provided a more favourable grouping in the latter one: the standard deviation is reduced by more than 10 percent and the frequency distribution of the results shifted towards higher scores. The presented results verify the conclusion that using IPA-PBL enriched the learning process and provided better results in the form of the course's final output.

We do not expect the application of our system to provide the same level of progress in future generations. IPA-PBL represents a well-designed solution with an extensible conceptual and pedagogical model that acts as a whole. Therefore, future improvements should focus on the development of additional tools and modules. Some of them would focus on linking with the environment: providing collaborative learning support or enabling integration with other learning applications, such as simulators, learning management systems or mobile applications. However, the main progress would be achieved by a tool designed for the easy editing and testing of entities contained in these pedagogy and ontology models. In this way, the system could be constantly improved by making changes to existing types of problems as well as creating new ones.

Appendix

A more detailed description of scenario used in the system is available in the Appendix which is accessible at a link: www.problembasedlearning.org/_file/scenario.pdf.

Acknowledgments

This work has been supported by the Serbian Ministry of Education and Science (projects III44006, ON174008, and TR32054).

References

[1] S. Abar, Y. Iwaya, T. Abe, and T. Kinoshita, "Exploiting domain ontologies and intelligent agents: An automated network management support paradigm," in *Proc. Int. Conf. Inf. Netw. Adv. Data Commun. Wirel. Netw.*, 2006, pp. 823–832.

[2] N. Baghaei and A. Mitrovic, "COLLECT-UML: Supporting individual and collaborative learning of UML class diagrams in a constraint-based intelligent tutoring system," in *Knowledge Based Intelligent Information and Engineering Systems*. Berlin, Germany: Springer, 2005, pp. 458–464.

[3] T. Barrett, "Understanding problem-based learning," in *Handbook of Enquiry & Problem Based, Learning*, T. Barrett, I. MacLabhrainn, and H. Fallon, Eds. Dublin: AISHE, 2005, pp. 14–25.

[4] H. S. Barrows, "Problem-based learning in medicine and beyond: A brief overview," *New Direct. Teach. Learn.*, vol. 68, pp. 3–12, 1996.

[5] D. Boud and G. Feletti, *The Challenge of Problem-Based Learning*. London: Kogan Page, 1997.

[6] J. Burguillo, "Using game theory and competition-based learning to stimulate student motivation and performance," *Comput. Edu.*, vol. 55, no. 2, pp. 566–575, 2010.

[7] C. P. Tracer, "Available within Cisco Networking Academy program from." [Online]. Available: http://cisco.netacad.net/. Accessed on: Sep. 1, 2014.

[8] R. Crowley and O. Medvedeva, "An intelligent tutoring system for visual classification problem solving," *Artif. Intell. Med.*, vol. 36, no. 1, pp. 85–117, 2006.

[9] E. de Graaf and A. Kolmos, "Characteristics of problem-based learning," *Int. J. Eng. Edu.*, vol. 19, no. 5, pp. 657–662, 2003.

[10] V. Devedzic, *Semantic Web and Education*. Berlin, Germany: Springer, 2006, Art. no. 192.

[11] J. Dong and H. Guo, "Enhance computer network curriculum using collaborative project-based learning," in *Proc. Annu. Conf. Am. Soc. Eng. Edu.*, 2011, pp. 1151–1564.

[12] P. Gil, F. A. Candelas, and C. A. Jara, "Computer networks e-learning based on interactive simulations and SCORM," *Int. J. Online Eng.*, vol. 7, no. 2, pp. 15–23, 2011.

[13] Graphical Network Simulator 3. [Online]. Available: http://www.gns3.net. Accessed on: Sep. 1, 2014.

[14] IP Address Range Calculator. [Online]. Available: http://www.csgnetwork.com/ipinfocalc.html. Accessed on: Sep. 1, 2014.

[15] IP Calculator. [Online]. Available: http://jodies.de/ipcalc. Accessed on: Sep. 1, 2014.

[16] L. Jiang, C. Zhao, and H. Wei, "The development of ontology-based course for computer networks," *Int. Conf. Comput. Sci. Softw. Eng.*, vol. 5, pp. 487–490, 2008.

[17] D. Krathwohl, "A revision of Bloom's taxonomy: An overview, theory into practice," *Publishing Models Article Dates Explained*, vol. 41, no. 4, pp. 212–218, 2002.

[18] T. A. Lacely and B. Wright, "Occupational employment projections to 2018," *Monthly Labor Rev. U.S. Bureau Labor Stat.*, vol. 132, no. 11, pp. 82–123, Nov. 2009

[19] N. Linge and D. Parsons, "Problem-based learning as an effective tool for teaching computer network design," *IEEE Trans. Edu.*, vol. 49, no. 1, pp. 5–10, Feb. 2006.

[20] B. Mills, M. Even, and R. Freedman, "Implementing directed lines of reasoning in an intelligent tutoring system using the atlas planning environment," in *Proc. Int. Conf. Inf. Technol. Coding Comput.*, 2004, pp. 729–733.

[21] G. Mitchell and J. Delaney, "An assessment strategy to determine learning outcomes in a software, engineering problem-based learning course," *Int. J. Eng. Edu.*, vol. 20, no. 3, pp. 494–502, 2004.

[22] A. Mitrovic and S. Ohlsson, "Evaluation of a constraint-based tutor for a database language," *Artif. Intell. Edu.*, vol. 10, pp. 238–256, 1999.

[23] NetAnim. [Online]. Available: http://www.nsnam.org/wiki/NetAnim. Accessed on: Sep. 1, 2014.

[24] A. Neville, "Problem-based learning and medical education forty years on," *Med. Principles Practice*, vol. 18, no. 1, pp. 1–9, 2009.

[25] V. Nguyen, "Ontologies and information systems: A literature survey," *Defence Sci. Technol. Org.*, Salisbury, MD, Australia, Tech. Rep. DSTO-TN-1002, pp. 52–62, 2011.

[26] E. Nuutila, S. Torma, and L. Malmi, "PBL and computer programming – the seven steps method with adaptations," *Comput. Sci. Edu.*, vol. 15, no. 2, pp. 123–142, 2005.

[27] M. O'Grady, "Practical problem-based learning in computing education," *ACM Trans. Comput. Edu.*, vol. 12, no. 3, 2012, Art. no. 10.

[28] G. Shimic and A. Jevremovic, "Problem-based learning in formal and informal learning environments," *Interactive Learn. Environments*, vol. 20, no. 4, pp. 351–367, 2012.

[29] Subnet Calculator. [Online]. Available: http://www.subnet-calculator.com. Accessed on: Sep. 1, 2014.

[30] P. Suraweera and A. Mitrovic, "An intelligent tutoring system for entity relationship modelling," *Int. J. Artif. Intell. Edu.*, vol. 14, pp. 375–417, 2004.
[31] L. Torp and S. Sage, *Problems as Possibilities: Problem-Based Learning for K-16 Education*, 2nd ed. Alexandria, VA: Association for Supervision and Curriculum Development, 2002.
[32] J. Undercoffer, A. Joshi, and J. Pinkston, *Modeling Computer Attacks: An Ontology for Intrusion Detection*, G. Vigna, E. Jonsson, and C. Kruegel, Eds. New York, NY, USA: Springer, pp. 113–135, 2003.
[33] J. A. Wang, M. Guo, H. Wang, M. Xia, and L. Zhou, "Ontology-based security assessment for software products," in *Proc. 5th Annu. Workshop Cyber Secur. Inf. Intell. Res. Cyber Secur. Inf. Intell. Challenges Strategies*, 2009, Art. no. 15.
[34] G. Weber and P. Brusilovsky, "ELM-ART: An adaptive versatile system for web-based instruction," *Int. J. Artif. Intell. Edu.*, vol. 12, pp. 351–384, 2001.
[35] M. Erdt, A. Fernandez, and C. Rensing, "Evaluating recommender systems for technology enhanced learning: A quantitative survey," *IEEE Trans. Learn. Technol.*, vol. 8, no. 4, pp. 326–344, Dec. 2015
[36] N. Guarino, D. Oberle, and S.-F. Staab, "What is an ontology?," in *Handbook on Ontologies*, 2nd ed., S. Staab and R. Studer Eds. Berlin, Germany: Springer-Verlag, 2009, Art. no. 2.
[37] P. Brusilovsky and C. Peylo, "Adaptive and intelligent web-based educational systems," *Int. J. Artif. Intell. Edu. 13*, vol. 13, no. 2–4, pp. 156–169, 2003.

Aleksandar Jevremovic received the BSc, MSc and PhD degrees at Singidunum University, Serbia. He is an associate professor of computer network security at Singidunum University. So far, he has authored/co-authored a number of research papers and made contributions to three books about computer networks, computer networks security, and Web development. He is recognized as an Expert Level instructor at the Cisco Networking Academy program.

Goran Shimic received the BSc at University of Defense, MSc at University of Belgrade and PhD degree at Singidunum University, Serbia. He is the director of the Center for Simulations and Distance Learning, Military Academy, University of Defense, Belgrade, Serbia (http://www.va.mod.gov.rs/cms/view.php?id=21662). He is also an associate professor of computer science specialized in expert systems, object-oriented design and implementation, and business software applications. His current research interests include area of interoperability between different learning resources & systems, and their integration with diverse types of Web services and applications. So far, he has authored/co-authored about 50 research papers and made contributions to seven books. He is a member of the GOODOL-DAI research network (http://www.goodoldai.org/goran_simic).

Mladen Veinovic received the BSc, MSc, and PhD degrees from the Faculty of Electrical Engineering, University of Belgrade, Serbia, in 1986, 1990, and 1996, respectively. From 1987 to 2005, he worked in the Institute of Applied Mathematics and Electronics, Belgrade, where he was head of the department for speech processing. Since 2005, he has been working at Singidunum University, where he is currently a vice rector. He is the author of seven books and a number of journals and conference scientific papers. His current research interests include security, computer networks, data bases, speech analysis and compression, and digital signal processing.

Nenad Ristic received the bachelor's degree in the field of programing from Sinergija University in 2009, and the master's degree in the field of advanced information technologies from Singidunum University in 2010. He is currently working toward the PhD degree in the field of advanced security systems at Singidunum University. Since March 2007, he has been a head IT at Sinergija University, a graduate teaching assistant at Sinergija University since April 2009, and a program administrator at Microsoft IT Academy at Sinergija University since January 2011. His fields of interests include computer networks, Web development, software engineering, and computer science education.