

**Web-Based Intelligent Tutoring for Prolog
Following the Learning Theory of Constructivism**

Afroditi Stathaki

A THESIS

Submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

DEPARTMENT OF INFORMATICS ENGINEERING

SCHOOL OF APPLIED TECHNOLOGY

TECHNOLOGICAL EDUCATIONAL INSTITUTE OF CRETE

November 2017

Approved by:

Supervisor
Prof. Emmanouil Marakakis

Abstract

Intelligent tutoring systems (ITS) incorporate techniques for transferring knowledge and skills to students. These systems are based on a combination of computer-aided instructions and artificial intelligence techniques. This thesis studies and develops a web-based intelligent tutoring system for Prolog following the learning theory of constructivism. Our thesis studies among others the kind of programming knowledge, which is required in the learning process by learners (programming techniques, program schemata, etc.), how this knowledge is represented, and how this knowledge can be applied in an effective way. The constructivism theory is encapsulated into our system in order to have high quality learning outcomes. This theory is based on the fundamental principles of *interaction*, *construction*, *assimilation* and *knowledge consolidation*. These four principles have been taken into account in the development of our system. As such, the learning is performed through interaction of the students/learners with the system. The learning process is guided by iteratively constructing solutions on assigned problems. Our system instructs learners by giving guidelines on how they can construct a solution to their problem. In addition, each learner can try a set of exercises and the system checks the correctness of the provided answers. In case of erroneous ones, the system gives appropriate instructions to the student in order to reconstruct a correct answer. Many learning decisions are taken by the learner rather than by the system due to the constructivism theory. The practice and the assessment modules support the principles of *assimilation* and *knowledge consolidation*. In the practice module, the learner has to practice the theory he learnt on a set of exercises by choosing a practice method from the available ones. In the assessment module, the learner has to answer a set of exercises on what he has already learnt. The main novelty of our system is its flexibility to adapt to individual student choices and profile, offering a wide range of alternatives and trying to continuously retain the interest of learners. Finally, a preliminary evaluation of our system confirms its usability and its benefits of using it for learning Prolog.

Abstract

Τα ευφυή συστήματα διδασκαλίας (ITS) ενσωματώνουν τεχνικές για τη μεταφορά γνώσεων και δεξιοτήτων στους σπουδαστές. Αυτά τα συστήματα βασίζονται σε ένα συνδυασμό οδηγιών με τη βοήθεια του υπολογιστή και της τεχνητής νοημοσύνης. Αυτή η διπλωματική εργασία μελετά και αναπτύσσει ένα ευφύε διαδίκτυακό σύστημα διδασκαλίας βασισμένο και με σκοπό να διδάξει την γλώσσα προγραμματισμού Prolog, ακολουθώντας τη θεωρία μάθησης του κονστρουκτιβισμού. Η παρούσα διπλωματική εργασία μελετά, ποιό είναι το είδος της προγραμματιστικής γνώσης που απαιτείται από τη μαθησιακή διαδικασία από τους εκπαιδευόμενους (τεχνικές προγραμματισμού, προγράμματα, κλπ.), πώς αντιπροσωπεύεται αυτή η γνώση και πώς μπορεί να εφαρμοστεί αυτή η γνώση με αποτελεσματικό τρόπο.

Η θεωρία του κονστρουκτιβισμού είναι ενσωματωμένη στο σύστημά μας, προκειμένου να υπάρξουν μαθησιακά αποτελέσματα υψηλής ποιότητας. Αυτή η θεωρία βασίζεται στις θεμελιώδεις αρχές της αλληλεπίδρασης, της κατασκευής, της αφομοίωσης και της εδραίωσης της γνώσης. Αυτές οι τέσσερις αρχές έχουν ληφθεί υπόψη στην ανάπτυξη του συστήματός μας. Ως εκ τούτου, η εκμάθηση πραγματοποιείται μέσω της αλληλεπίδρασης των μαθητών / εκπαιδευόμενων με το σύστημα. Η διαδικασία εκμάθησης καθοδηγείται από την επαναληπτική κατασκευή λύσεων σε συγκεκριμένα προβλήματα. Το σύστημά μας καθοδηγεί τους εκπαιδευόμενους δίνοντας κατευθυντήριες γραμμές για το πώς μπορούν να δημιουργήσουν μια λύση στο πρόβλημά τους. Επιπλέον, κάθε εκπαιδευόμενος μπορεί να δοκιμάσει ένα σύνολο ασκήσεων και το σύστημα ελέγχει την ορθότητα των παρεχόμενων απαντήσεων. Σε περίπτωση λανθασμένων απαντήσεων, το σύστημα δίνει τις κατάλληλες οδηγίες στον σπουδαστή προκειμένου να ανακατασκευάσει μια σωστή απάντηση. Οι κύριες αποφάσεις για την πρόοδο της μάθησης λαμβάνονται από τον μαθητή και όχι από το σύστημα λόγω της θεωρίας του κονστρουκτιβισμού. Οι πρακτικές και οι ενότητες αξιολόγησης υποστηρίζουν τις αρχές της αφομοίωσης και της ενοποίησης της γνώσης. Στο μάθημα πρακτικής, ο εκπαιδευόμενος πρέπει να ασκηθεί πάνω στη θεωρία που έμαθε σε ένα σύνολο ασκήσεων επιλέγοντας μια πρακτική μέθοδο από τις διαθέσιμες. Στη μονάδα αξιολόγησης, ο εκπαιδευόμενος πρέπει να απαντήσει σε ένα σύνολο ασκήσεων σχετικά με όσα έχει ήδη μάθει.

Η κύρια καινοτομία του συστήματός μας είναι η ευελιξία του να προσαρμόζεται στις επιμέρους επιλογές των σπουδαστών, προσφέροντας ένα ευρύ φάσμα εναλλακτικών λύσεων και προσπαθώντας να διατηρήσει συνεχώς το ενδιαφέρον των μαθητών αμείωτο. Τέλος, μια προκαταρκτική αξιολόγηση του συστήματός μας επιβεβαιώνει τη χρηστικότητα και τα οφέλη της χρήσης του για την εκμάθηση του Prolog.

Dedication

To my grandparents Dimitris, Afroditi!

Acknowledgements

I would like to express my sincere thanks to my advisor Professor Emmanouil Marakakis. His patience, encouragement and immense knowledge were key motivation throughout this thesis. Also, I would like to thank Alex Kanta for supporting in web-design and Mr. Maniki for supporting in procedure of survey. Finally, I would like to thank my family and friends for their love and support.

Table of Contents

<i>Abstract</i>	<i>iii</i>
<i>Acknowledgements</i>	<i>vii</i>
<i>Table of Contents</i>	<i>ix</i>
<i>List of Tables</i>	<i>xi</i>
<i>List of Figures</i>	<i>xii</i>
<i>List of Program Schemata</i>	<i>xiii</i>
<i>List of Algorithms</i>	<i>xiv</i>
<i>List of Schema Instance Paradigms</i>	<i>xv</i>
1 Overview and Motivation	16
1.1 Introduction	16
1.2 Problem Definition	19
1.3 Thesis Structure	20
2 Background	20
2.1 Introduction to e-learning	20
2.2 Learning Theories	21
2.3 Teaching Methods	23
2.4 Intelligent Tutoring Systems	24
2.5 Programming in Prolog	26
2.6 Web 2.0 technology	27
3 Related Work, Overview of our Prolog Intelligent Tutoring System and its Comparison 29	
3.1 Intelligent Tutoring Systems for Prolog	29
3.2 I.T.S in domain other than Prolog	31
3.3 Overview of our Prolog Intelligent Tutoring System	32
3.4 Comparison of our PITS with Related Systems	35
4 Architecture of PITS	37
4.1 Tutoring or Pedagogical module	38
4.2 Student or Learner module	40
4.3 The Teacher module	40
4.4 Knowledge Base	41
4.4.1 The domain model	41
4.4.2 The learner's or student's model	45
4.4.3 The tutoring model	46
4.5 Interface	47
5 Web Features of PITS	48
6 Detailed Presentation of the PITS Components	50
6.1 The Lecture Module	51
6.2 The Practice Module.....	52
6.3 The Assessment Module.....	58
6.4 The Teacher Module.....	59
6.5 Our System and the Principles of Constructivism Theory	61
7 Sample Sessions of PITS	62
7.1 Sample Sessions of Lecture Section	62
7.2 Sample Sessions of Practice Section	65
7.3 Sample Sessions of Assessment Section	69
7.4 Sample Sessions of Teacher Module	71

8	<i>Evaluation of PITS</i>	73
8.1	Procedure of Survey	73
8.1.1	Questionnaire Survey	73
8.1.2	Sample of Questionnaire Survey	74
8.2	Analysis of the Collected Data	75
8.2.1	Discussion and Analysis of Questionnaire	75
8.2.2	Further Analysis of the Collected Data from Other Perspectives.	79
9	<i>Conclusions and Future Work</i>	84
9.1	Conclusions	84
9.2	Future Work	85
10	<i>Bibliography</i>	87
11	<i>Appendix A: Code of Modules</i>	91
11.1	A.1. Algorithm of Lecture, Practice and Assessment Modules	91
11.2	A.2. Algorithm of Teacher Module	93
12	<i>Appendix B</i>	95
12.1	B.1. On-line questionnaire of Survey	95

List of Tables

<i>Table 1:Teaching Methods which are used in this PITS.....</i>	<i>39</i>
--	-----------

List of Figures

Figure 1: Pressey's intelligent machine	17
Figure 2: Traditional architecture of ITS.	25
Figure 3: LMS vs PITS.	35
Figure 4: Architecture of our PITS.	37
Figure 5: Use case diagram of PITS.	50
Figure 6: Structure of Lecture Module in PITS.	51
Figure 7: Structure of Practice Module in PITS.	53
Figure 8: Structure of Assessment Module in PITS.	59
Figure 9: Structure of Teacher Module in PITS.	60
Figure 10: First page of lecture with theory.	62
Figure 11: Page of lecture with question.	63
Figure 12: Page of feedback when the answer of student is correct.	64
Figure 13: Page of feedback when the answer of student is not correct.	64
Figure 14: Page of selections: Lecture, Practice and Assessment.	65
Figure 15: Page of practice selections: Step by step, Schema and Prolog.	66
Figure 16: Page of selection practice with help: Step by step.	67
Figure 17: Page of selection practice with schema based help: Schema.	68
Figure 18: Page of selections practice without help: Prolog.	69
Figure 19: Page of selection: Assessment.	70
Figure 20: The demonstration of results.	70
Figure 21: Page of teacher.	71
Figure 22: The end of course with questionnaire.	74
Figure 28 : Overall "like" scale of the system by the learners.	77
Figure 29: "pleasant" scale of the system by the learners.	78
Figure 33: Diagram - "Do you understand at first glance of the first page (main interface) what this system is about?" VS "Which is your current registration semester?".	80
Figure 34: Diagram - "How easy is to find the theory of a lesson?" VS "Which is your current registration semester?".	80
Figure 35: Diagram - "Do the available practice methods satisfy your learning requirements?" VS "Which is your current registration semester?".	81
Figure 36: Diagram - "Is the working environment pleasant? (Give a number from 1 to 5)" VS "Evaluate the system in the scale 1 (I do not like) to 5 (like very much)."	82
Figure 37: Diagram - "Do you see any functionality problem in this system?	83
Figure 38: On-line questionnaire of Survey.	95

List of Program Schemata

<i>Program Schema 1: Structure construction in the body of a clause.....</i>	<i>55</i>
<i>Program Schema 2: Structure construction in the head of a clause.....</i>	<i>56</i>

List of Algorithms

Algorithm 1: Algorithm of Lecture, Practice and Assessment Modules.....93
Algorithm 2: Algorithm of TeacherModules.....94

List of Schema Instance Paradigms

<i>Schema Instance Paradigm 1: Delete all of repetitions with using technique “structure construction in the head of a clause”</i>	56
<i>Schema Instance Paradigm 2: Delete all of repetitions with using technique “structure construction in the body of a clause”</i>	56

1 Overview and Motivation

Nowadays there is an intense discussion and movement in the education field about learning models. Educators from all over the world are expressing the need of a new education paradigm. Many and different learning models have been adopted and tested by various education societies such as public schools, universities, colleges and others. The aim of this thesis is to introduce and analyze an e-learning model called *Intelligent Tutoring Systems (ITS)*.

This thesis studies the features needed by an intelligent tutoring system in order to be used from both students and teachers, in distance and traditional learning. On this premise, we constructed a system, which aims to teach Prolog to students in higher education. This system was tested by students and the results of this study are presented in this thesis.

1.1 Introduction

Many researchers have tried to define the term ITS. Graesser, in the chapter «*Application to Learning and teaching*» defines that «*ITSs are computerized learning environments that incorporate computational models from the cognitive sciences, learning sciences, computational linguistics, artificial intelligence, mathematics, and other fields.* ». Another similar definition of the term is given in Wikipedia: «*An intelligent tutoring system (ITS) is a computer system that aims to provide immediate and customized instruction or feedback to learners, usually without intervention from a human teacher. ITSs have the common goal of enabling learning in a meaningful and effective manner by using a variety of computing technologies* » (Graesser et al, 2010).

The idea of intelligent machines has been discussed centuries ago. The well-known Pascal's calculator is an intelligent machine of the 17th century that was created by Blaise Pascal and it was capable of calculating mathematical functions. In 1924 a psychology professor Sidney L. Pressey in Ohio University created a teaching machine that was mechanical and was used to instruct students without a human teacher. His machine resembled closely a typewriter with several keys and a window that provided

the learner with questions. The Pressey Machine allowed user input and provided immediate feedback by recording their score on a counter (*Figure 1*).



Figure 1: Pressey's intelligent machine

Pressey's teaching and testing machine would not be considered intelligent as it was mechanically run and it was based on one question and answer at a time, but it set an early model for future projects (Shute, 1994).

In the period following the Second World War, Alan Turing a mathematician, logician and computer scientist, linked computing systems to logical thinking. He created the so-called "Turing Test" where a person communicates with two other agents, a human and a computer asking questions to both recipients. The computer passes the intelligence test if it can respond in such a way that the human posing the questions cannot differentiate between the other human and the computer. The Turing test has been used in its essence for more than two decades as a model for current ITS development (Shute, 1994).

In the latter part of the 1960s and 1970s, many new CAI (Computer-Assisted instruction) projects were developed, built on advances in Computer Science. For many years, educational languages and machines were developed, based on logic and CAI such as Algol, Logo and Plato: an educational terminal featuring displays, animations, and touch controls that could store and deliver large amounts of course material.

In the late 1990s, there was a shift in Artificial Intelligence (AI) research as systems moved from the logic focus of the previous decade to knowledge based systems that could make intelligent decisions based on prior knowledge (Buchanan, 2006). Such a

program was created by Seymour Papert and Ira Goldstein who created Dendral, a system that predicted possible chemical structures from existing data.

The microcomputer revolution in the late 1970s and early 80s helped to revive CAI development and jumpstart development of ITS systems. Personal computers such as the Apple 2, Commodore PET, and TRS-80 reduced the resources required to own computers and by 1981, 50% of US schools were using computers (Chambers & Sprecher, 1983). Several CAI projects utilized the Apple 2 as a system to deliver CAI programs in high schools and universities including the British Columbia Project and California State University Project in 1981.

A key breakthrough in ITS research was the creation of LISPITS, a program that implemented ITS principles in a practical way and showed promising effects increasing student performance. LISPITS was developed and researched in 1983 as an ITS system for teaching students the LISP programming language. LISPITS could identify mistakes and provide constructive feedback to students while they were performing the exercise. The system was found to decrease the time required to complete the exercises while improving student test scores. Other ITS systems beginning to develop around this time include TUTOR created by Logica in 1984 as a general instructional tool and PARNASSUS created in Carnegie Mellon University in 1989 for language instruction (Buchanan, 2006).

After the implementation of initial ITS, more researchers created a number of ITS for different students. In the late 20th century, the Intelligent Tutoring Tools (ITTs) was developed by the Byzantium project, which involved six universities. In addition, there were three ITS projects that functioned based on conversational dialogue: AutoTutor, Atlas and Why2. The idea behind these projects was that since students learn best by constructing knowledge themselves, the programs would begin with leading questions for the students and would give out answers as a last resort. AutoTutor's students focused on answering questions about computer technology, Atlas's students focused on solving quantitative problems, and Why2's students focused on explaining physical systems qualitatively (Buchanan, 2006).

Advances in ITS have progressed to the point of being used in school systems. One noteworthy example is the Cognitive Tutors developed by the Pittsburgh Science of

Learning Center. The Cognitive Tutors help students learn algebra, geometry, and programming languages by applying learning principles.

In this thesis, we present an Intelligent Tutoring System which is based on the theory of constructivism. This system follows some teaching methods and it aims to help students to learn Prolog Programming. It is a web-based system with some characteristics common to other educational systems. Our system can be used in realtime learning or in distance learning. Initially, the system presents the theory of each lesson of Prolog and some examples for exercises in Prolog programming. Then, the practice module is used allowing students to use any of the three types of practice: *hint-based*, *schema-based* and *no guidelines practice*. The whole learning environment is personalized according to individual profiles. It provides examples and appropriate instructions according to the personal learning style, making the whole learning process as personalized as possible. Finally, evaluation of the student progress is performed in the assessment module. The progress evaluation results determine whether the student can proceed to the next lecture. A teacher is also allowed optionally, and if present she/he can supervise the whole process asynchronously and identify the progress of the individual students. The implemented system is a prototype. We have performed an initial evaluation of the system with very satisfactory results. These results are also discussed in Chapter 8 of the thesis.

1.2 Problem Definition

The primary goal of the ITS is to mimic human tutors by adapting instruction to an individual student's performance, strengths and weaknesses, and other attributes (Nguyen et al, 2009). In addition, such systems have the benefit of web-based education, which is independent of teaching and learning with respect to time and space. Courses are installed and maintained in one place may be used by a huge number of users all over the world. The World Wide Web offers an unprecedented opportunity for addressing large audiences with intelligent tutoring technology and thus giving rise to various endeavors in this field.

The main goal of this study is to develop a web-based intelligent tutoring system for Prolog with novel features compared with the ones that have already been developed. This thesis studies the development of a new I.T.S. which follows the learning theory

of constructivism and it shares some of the characteristics of a *Learning Management System (LMS)*. Our system aims to teach Prolog programming and apply it in education in real time and in distance learning. The system instructs learners on how to construct Prolog programs. Each learner tries his/her own examples and the system locates the learner's misconceptions on Prolog programming making the necessary corrections and giving appropriate instructions.

1.3 Thesis Structure

The rest of this thesis is organized in the following sections. In Chapter 2, we present the background and related work in Intelligent Tutoring System and e-learning. The applications of e-learning, educational theories/strategies and Prolog Programming are also discussed. In Chapter 3, we present related work in ITS for Prolog. We make an overview of our Prolog Intelligent Tutoring System (PITS) and make a comparison of our system with the ones in related work. In Chapter 4, we present the architecture of PITS. In Chapter 5, we present the web features of our system. In Chapter 6, details of the components of our system are presented. We also describe the operations of each component. In Chapter 7, some samples of PITS are presented. Chapter 8 presents and discusses the evaluation of the system and the results of this analysis. In Chapter 9, we discuss the conclusions of this research and future work.

2 Background

In this chapter, we present the basic concepts of e-learning, learning theories and educational strategies and techniques. We also make a short introduction to ITS and to programming in Prolog.

2.1 Introduction to e-learning

During the 20th century, new technologies in communication and computer science try to promote an alternative learning standard. Nowadays, the learning standard has evolved because new technologies are embedded in traditional learning (Mallon, 2010). According to this new model, the concept of e-learning has been created. Many researchers have tried to define e-learning. Still, there is no commonly accepted definition (Oye et al., 2012).

In fact, it is difficult for researchers to perform meaningful cross-study comparisons and build on the outcomes from the previous studies due to the large extend of

opinions, techniques and fields. This contributes to conflicting findings about distance learning, e-learning, and online learning environments. In addition, terms are often interchanged without meaningful definitions. Some authors explicitly define e-learning, others imply a specific definition or view of e-learning in their articles.

According to the opinion of Elis, “e-learning not only covers content and instructional methods delivered via CD-ROM, the Internet or an Intranet but also includes audio- and videotape, satellite broadcast and interactive TV” (Aktison, 2002). Another approach is that e-learning includes the constructivist theoretical model as a framework for its definition. Therefore, some researchers believe that e-learning is not only procedural but also shows some transformation of an individual's experience into the individual's knowledge through the knowledge construction process (Wang et al., 2008).

Both Aktison and Wang believed that some level of interactivity needs to be included to make the definition truly applicable in describing the learning experience (Aktison, 2002) (Wang et al., 2008). Some authors also provide either no clear definition or a very vague reference to other terms such as online course/learning, web-based learning, web-based training, learning objects or distance learning believing that these terms can be used synonymously.

Besides, many terms are used to describe learning that is delivered online, via the internet, ranging from Distance Education, to computerized electronic learning, online learning, internet learning and many others.

We define eLearning as *courses that are specifically delivered via the internet to somewhere other than the classroom where the professor is teaching*. It is not a course delivered via a DVD or CD-ROM, video tape or over a television channel. It is interactive in that you can also communicate with your teachers, professors or other students in your class. Sometimes it is delivered live, where you can “electronically” raise your hand and interact in real time and sometimes it is a lecture that has been prerecorded.

2.2 Learning Theories

Many e-learning systems have been constructed aiming to approach some learning theories. In most of the cases, the courses of e-learning systems are planned and

designed using learning theories. There are four basic types of learning theories: *behaviorism*, *cognitivism*, *constructivism*, and *social constructivism* (Park Y, 2011).

Behaviorism is the doctrine that regards psychology as a scientific study of behavior and explains learning as a system of behavioral responses to physical stimuli (Madden G., 2013). Psychologists working within this theory of learning are interested in the effect of reinforcement, practice, and external motivation on a network of associations and learned behaviors. Behaviorist theory may have implications for changing behavior, but it offers little in the way of explaining cognitive change a structural change in understanding.

Cognitivism, as a psychological theory, stems from the burgeoning field of cognitive science, particularly the later work of Jean Piaget just prior to his death. It also has roots in biology and in evolution of the logical outcome of the work of contemporary scientists (Madden G., 2002). In fact, behaviours or skills as the goal of instruction, cognitive development and deep understanding are the focal point.

Constructivism is a theory of learning that it has its roots in both philosophy and psychology. Constructivism acknowledges the learner's active role in the personal creation of knowledge (Scerri, 2003). There are four fundamental tenets, 1) *interaction*, 2) *construction*, 3) *assimilation* and 4) *knowledge consolidation*, which provide the foundation for the basic principles of *teaching*, *learning*, and *knowledge evolution* processes. This theory supports that the learners can discover the knowledge through the interaction with reality. In this knowledge creation process, the knowledge created will vary in its degree of validity as an accurate representation of reality (Meyer, 2009). The learners are actively involved in the learning process and they can construct their own understanding. Their readings have further consequences in their *assimilation* and *knowledge consolidation* processes. It is important to note that constructivism is not a particular method of pedagogy. In fact, constructivism is a theory describing how the learning process is achieved.

Social constructivism is a theory of learning in which, the terms by which people perceive and describe the world, including language, are social artifacts. In this theory, the reality is seen to be created through processes of social exchange, historically situated; social constructivists are interested in the collective generation of meaning among people (McKinley, 2015). Social constructivism includes the idea that there is

no objective basis for knowledge claims, because knowledge is always a human construction. In addition, backers of social constructivism support that the knowledge is constructed by the social group and the inter subjectivity is established through the interactions of the group.

2.3 Teaching Methods

A teaching method includes a set of principle and strategies, which are used by the teacher with the aim to enable learning. Teaching methods, which are used by the teacher in learning process, depend on a number of factors such as 1) the developmental level of students, 2) the goals, 3) the intent, and objectives of the teacher, 4) the content, and 5) the environment (Petrina, 2007). In the most general terms, there are four different models, which consist of a set of teaching methods. These models are the *Didactic*, *Modeling*, *Managerial*, and *Dialogic*.

- **Didactic:** Is a direct teaching where the teacher delivers the knowledge with some method such as *lecturing or demonstrating*. Both methods are used by our system.
- **Modeling** Is a direct teaching which includes visual tools and typically in the form of *demonstration* and *practice*. The *practice* method is used by our system.
- **Managerial:** Is an indirect or interactive teaching, which uses the *individualization*, and *group management*. Our system is characterized by *individualization*.
- **Dialogic:** Is an indirect or interactive teaching, which uses *the Socratic Technique of dialogue* and *questions*. The *questions* method is used by our system.

There are some other additional teaching methods, which are used very often and they do not belong to a specific model. These methods, which are supported by our system, are the following.

- *Activity:* A general teaching method (e.g., problem solving, design challenge, field trips, and role-playing) based on planned, purposeful involvement of students.
- *Direct instruction:* A term used to describe explicit, step-by-step instruction directed by the teacher. The format or regimen advocated is demonstration, guided practice, and independent practice.
- *Drill and practice:* A form of independent study whereby, after the teacher explains a task, learners practice it.

- *Feedback*: A semi-formal mode of communicating to student's constructive criticism regarding their performance during an activity.
- *Individualized instruction*: It is a teaching method whereby teaching and learning are tailored to meet a learner's unique characteristics.
- *Module*: A module is a self-contained and comprehensive instructional package, meaning that everything that a student needs is in the module. A form of individualized instruction whereby students use a self-contained package of learning activities that guides them to learn.
- *Programmed and automated instruction*: A form of individualized instruction whereby information is learned by way of reading programming direction in textual form or using computer-based programs.
- *Presentation and lecture*: Students listen to a person who talks about a topic. The lecturer at the same time displays material, which illustrate the topic of presentation.

2.4 Intelligent Tutoring Systems

An ITS includes a set of estimates of the students mastery of his/her knowledge and skills and ability to apply them based on the student's performance in scenarios (Samuelis, 2007).

An ITS consist of the following four basic components:

1. The *Domain model*, which is the cognitive model that contains the concepts, rules, and problem-solving strategies of the domain to be taught (Le Ngueyen, 2007).
2. The *Student model*, which is the core component of an ITS. It represents the learner's emerging knowledge and skills. Information such as learning preferences, past learning experiences and advancement may also be relevant in adapting the teaching process. It may also record the learner's errors and misconceptions. The student knowledge processes, known as diagnostics, analyze the behavior of the student (Padayachee, 2002).
3. The *Tutoring model*, which makes choices about tutoring strategies and actions (Padayachee, 2002).

4. the *User interface model*, which integrates three types of information that are needed in carrying out a dialogue: a) knowledge about patterns of interpretation (to understand a speaker) and action (to generate utterances) within dialogues, b) domain knowledge needed for communicating content and c) knowledge needed for communicating intent (Padayachee, 2002).

The interaction of these basic components is shown in *Figure 2*.

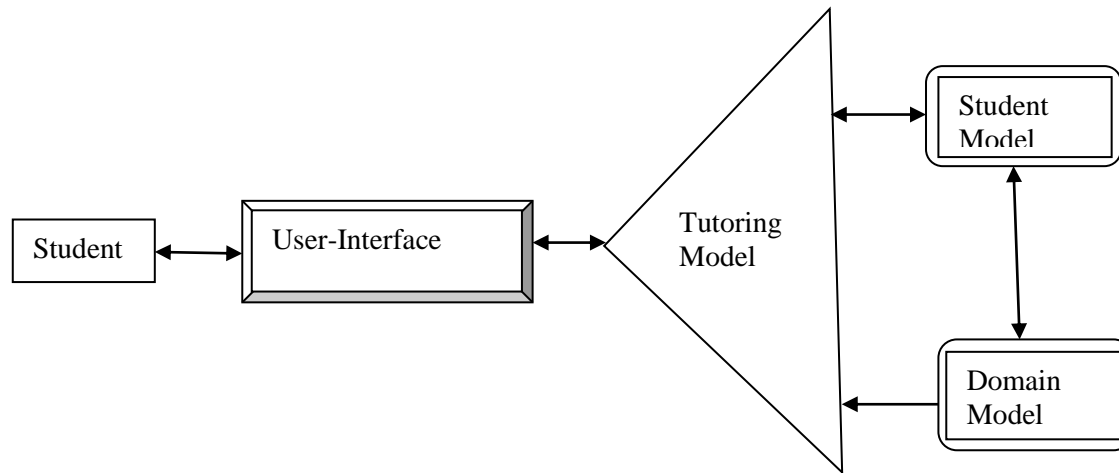


Figure 2:Traditional architecture of ITS.

Dede in 1986 added one more model in the above models, the *Pedagogical model*. This model determines which pedagogical means are most efficient in order to accomplish a given goal, alternative approaches to dialogue management (adjusting to different learning styles), and domain-dependent teaching heuristics (Elsom-cook, 1991).

The most recent ITS environments that have been tested on thousands of students, they can have been proven effective in helping students to learn. They include the *cognitive tutors*, *constraint-based tutors*, *case-based tutors*, and *tutors with animated conversational agents*. These systems also have scientific principles of learning that guide their design (Graesser et al, 2010).

Many ITSs have been constructed to help students learn geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, chemistry, etc.

Intelligent Language Tutoring Systems teach natural language to first or second language learners.

There are many benefits associated with ITSs. They train instructors and can replace instructors in situations where they may not be present. Training is tailored and customized to the individual student so more efficient student learning occurs. The knowledge incorporated into the ITS captures the expertise of best instructors and distributes it to all students.

Electronic tutors personalize learning much like a human tutor. When the course content presented is based on evaluation of the student's performance and knowledge, the resulting dynamic presentation differs for each student (Hong, 2004). In addition, computer tutors can rouse less anxiety in students than human tutors: Learners can practice as much as they want and make mistakes without anyone else knowing.

Another group of students who might benefit from ITSs are those that make in-person attendance difficult, those who are home-bound or do not have easy access to schools, such as students with severe disabilities or students living in rural areas. If distance-learning technology is available, learners can access the knowledge from wherever they are.

2.5 Programming in Prolog

Prolog is a declarative, general-purpose logic programming language associated with artificial intelligence and computational linguistics. Prolog has its root in first-order logic. Prolog's implementation is based on Horn Clause logic, which is a subset of first-order logic. This language is used for solving problems that involve objects and relationships between objects. Computer programming in Prolog consist of facts, rules and questions about the objects and their relationship. The approach in Prolog, is more about describing known facts and relationships about a problem and about prescribing the sequence of steps taken by a computer to solve the problem.

Prolog also provides a uniform data structure called *term*. All data are expressed as terms. A Prolog program consists of a set of clauses, where each clause is either a fact or a rule. A clause is of the form "Head :- Body." and is read as "Head is true if Body is true". Clauses with empty bodies are called facts. An example of a fact is "parent(john, nick)." which can be written in equivalent form as the clause "parent(john, nick) :- true.". A rule is a clause with non-empty body. A rule's body consists of calls to

predicates, which are called the rule's goals. For example, the next rule "child(X,Y) :- parent(Y,X)." states that "X is child of Y if Y is parent of X."

SWI-Prolog is a versatile implementation of the Prolog. It provides a free programming environment and it is an attractive language for realizing applications. SWI-Prolog also offers a variety of development environments, most of which may be combined at will. SWI-Prolog is equipped with an extensive web server (HTTP) framework that can be used both for providing (REST) services and for end-user applications based on HTML5+CSS+JavaScript. Pengines (Prolog engines) allow clients to run queries against a client-provided program on a remote server using a generic API. These features of SWI-Prolog are the reason to develop our system that will be presented below.

2.6 Web 2.0 technology

Web 2.0 is a new term introduced in 2003-04, which is commonly used to encompass various novel ideas on the World Wide Web. Although initially largely a marketing term, some of the key attributes associated with Web 2.0 include the growth of social networks, bi-directional communication, various new technologies, and significant diversity in content types (Du & Wagner, 2006). The most basic characteristics of Web 2.0 include:

- User interaction with web applications.
- The web as a platform.
- Rich user experiences.
- Data as the driving force.
- An architecture of participation.
- Harnessing collective intelligence.
- A rich, interactive, user-friendly interface.
- Leveraging of popular trends, including blogging, social tagging, wikis, and peer-to-peer sharing.
- Inclusion of emerging web technologies like RSS, AJAX, APIs (and accompanying mishaps), Ruby on Rails and others
- Open source or sharable/editable frameworks in the form of user-oriented “create your own” APIs.

The technology infrastructure of Web 2.0 is complex and evolving. It includes the use of content syndication, server software, standards-based browsers that have extensions and plug-in, client applications, and messaging protocols (Du & Wagner, 2006). All these advanced technology provides Web 2.0 with dissemination, creation, and information storage capability. There are a number of major techniques that Web 2.0 website uses. Some of them include:

- CSS, semantically valid XHTML markup, and micro formats.
- Significant and clean URLs.
- Aggregation of RSS/ATOM data.
- Syndication of data in RSS/ATOM.
- REST or XML Web service APIs.
- Some social networking aspects.
- Support posting to a weblog.

The most important benefit of applying the Web 2.0 technologies is the participation of students in the content creation. Students become more active and exercise their social communication and creative capabilities. Web 2.0, mobile technologies and concepts can be used for implementing tools, which will supplement existing formal learning programs and enable informal learning.

3 Related Work, Overview of our Prolog Intelligent Tutoring System and its Comparison

There are many benefits associated with ITSs because they have been constructed to help students to learn with pleasure and ease. ITSs have been constructed to help students learn geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, chemistry, etc. Intelligent Language Tutoring Systems teach natural language to first or second language learners.

Initially, in this section we will present ITS that have been developed for tutoring Prolog, called PITS for Prolog ITS. For each one we will make a comparison with our system. Then, we will present ITS for other domains.

3.1 Intelligent Tutoring Systems for Prolog

Prolog Intelligent Tutoring Systems (PITS) is an ITS which aims to teach Prolog programming. PITS is a system that detects errors and proposes corrections necessary to fix them (Hong, 2004). PITS is used for teaching Prolog programming exploited by both teacher and student. It is very interesting to underline that every PITS incorporates a program debugger, which diagnoses bugs in the student programs and suggests specific corrections of the identified bugs (Chalk, 1987). The aforementioned description emphasizes on the two PITS aspects. One aspect gives emphasis in the detection of errors and the provision of support for correcting them and the other aspect emphasizes on the teaching Prolog.

Advances in PITS have progressed to the point of being used in university systems with positives results. All developed PITS satisfy one basic rule. They detect errors. However, the ways in which errors are handled are quite different. Always, the main goal is to teach Prolog programming (Webb G. I., 1989).

For example, Augmented Prolog Programming Environment (APPE) is a system like PITS, which manages singularly student programs and can detect errors by dividing them into five categories (Moon-c, 1990). It also can provide advice in case the errors by the student. Therefore, as a teacher of Prolog, PITS notices the following categories of bugs from programs of students: 1) Incorrect Solutions, 2) Uncovered Solutions, 3) Non-termination, 4) Redundant Solutions, 5) Invalid Parameters. In this

way, it can detect the errors and help the students of Prolog programming (Moon-c, 1990).

Another PITS with automatic debugging of Prolog programs, detects errors from programs and proposes the use of multiple sources for their correction (Looi, 1991). This teaching procedure aims to enhance and to deepen learner's understanding. It also uses the best-first algorithm to report key definitions of the program in a hierarchical structure. This system consists of two parts. The first part performs multiple program analyses on the programs, and the second part uses task-specific knowledge to analyze the program, detect bugs and correct them (Looi, 1991).

A Schema is another PITS developed for teaching Prolog (Bielikova & Narvat, 1997). The system separates the diagnosis of errors in programs in two categories. The first category contains programs that cannot be completed because of errors. The second category includes programs that may work but they are not syntactically correct (Bielikova & Narvat, 1997). In fact, the development is based on a set of recursive program schemata, which cover three important aspects of recursion (Gegg-Harrison, 1993). That is 1) single multiple methods for recursive step execution, 2) tail fat recursion reduction to a simpler task, monotonic or non-monotonic recursion and 3) single multiple recursion change in one or several arguments (Bielikova & Narvat, 1997).

The system in (Vlahavas I. et al, 2009) is a PITS that can be used for distance learning and contains text, images and animation for each lesson. This system provides many examples, bibliography, exercises and many links for each lesson. This system consists of three levels and it is based on multimedia assets and HTML. The first level consists of the basic concepts, the second level consists of advanced topics and the third level consists of developing applications of Prolog (Vlahavas I. et al, 2009).

There are many researches developed PITS, which they have been tested a lot by users with interesting results. The developed systems provide teaching Prolog programming by supporting person mentoring and educational digital environment combined with electronic intelligent tutoring (Chalk, 1987).

3.2 I.T.S in domain other than Prolog

In nowadays there are many applications of Intelligent Tutoring Systems. In the following, we present a number of systems, which have been constructed for use in other domains than tutoring Prolog.

The **Mathematics Tutor** helps students solve problems using fractions, decimals and percentages. The tutor records the success rates while a student is working on problems. It continuously provides subsequent, lever-appropriate problems for the student to work on. The subsequent problems that are selected are based on student's ability and a desirable time in which is estimated that the student is expected to solve the problem (Beal C. et al, 1998).

E-Teacher is an intelligent agent that supports personalized e-learning assistance. It builds student profiles while observing student performance in online courses. Then it uses information from the student's performance to suggest personalized courses of action designed to assist their learning process (Schiaffino S. et al, 2008).

ZOSMAT was designed to address all the needs of a real classroom. It follows and guides a student in different stages of their learning process. This is a student-centered, ITS which records the progress of student learning, and the student program changes based on the student's effort. ZOSMAT can be used for either individual learning or in a real classroom environment alongside the guidance of a human tutor (Keles A. et al, 2009).

REALP was designed to help students enhance their reading comprehension by providing reader-specific lexical practice and offering personalized practice with useful, authentic reading materials gathered from the Web. The system automatically builds a user model according to student's performance. After reading, the student is given a series of exercises based on the target vocabulary found in reading (Heffernan N. et al, 2006).

SmartTutor University of Hong Kong (HKU) developed a SmartTutor to support the needs of continuing education students. Personalized learning was identified as a key need within adult education at HKU and SmartTutor aims to fill that need. SmartTutor provides support for students by combining Internet technology, educational research and artificial intelligence (Cheung B. et al, 2003).

AutoTutor assists college students in learning about computer hardware, operating systems and the Internet in an introductory computer literacy course by simulating the discourse patterns and the pedagogical strategies of a human tutor. AutoTutor attempts to understand learner's input from the keyboard and then to formulate dialog moves with feedback, prompts correction and hints (Graesser A. et al, 1999).

ActiveMath is a web-based, adaptive learning environment for mathematics. This system strives for improving long-distance learning, for complementing traditional classroom teaching, and for supporting individual and lifelong learning (Melis E. & Siekmann J., 2004).

SHERLOCK is used to train Air Force technicians to diagnose problems in the electrical systems of F-15 jets. The ITS create faulty schematic diagrams of systems for the trainee to locate and diagnose. It provides diagnostic readings allowing the trainee to decide whether the fault lies in the circuit being tested or if it lies elsewhere in the system. Feedback and guidance are provided by the system and help is available if requested (Lajoie S. & Lesgold A., 1989).

The **Cardiac Tutor's** aim is to support advanced cardiac support techniques to medical personnel. The tutor presents cardiac problems and, using a variety of steps, students must select various interventions. Cardiac Tutor provides clues, verbal advice, and feedback in order to personalize and optimize the learning. Each simulation, regardless of whether the students were successfully able to help their patients, results in a detailed report, which is then reviewed by the students (Eliot C. & WoolfB., 1994).

3.3 Overview of our Prolog Intelligent Tutoring System

In this section, we will make an overview of our Prolog Intelligent Tutoring System and a comparison with the related systems (Stathaki et al, 2017).

Our system is an Intelligent Tutoring System, which aims to teach Prolog programming. The tutoring of Prolog programming is organized into a set of lessons. It has been constructed in a way that share features of a ***Learning Management System (LMS)***. *The LMS is software for managing complex databases and it has been combined with digital frameworks for managing curriculum, training materials and evolution tools* (Ellis, 2009). A LMS allows anyone to create, track, manage and distribute learning materials of any kind. In addition, our PITS does not allow anyone to intrude in the material structure of courses. There is a way to change the material of

courses with the help of developer. In essence, the role of developer is reinforced and the communication between the developer and the teacher is necessary and it has to be done frequently.

The structure of the courses is based on the design of traditional teaching methods. This is a characteristic which is missing from others educational systems such as ITS and LMS. Each traditional lesson contains *lecture*, *practice* and *assessment*. In *lecture section*, teaching methods are used in order to introduce students to a particular field of study. Our system converts teaching to an active learning task because it contains teaching techniques like picture illustration and question-answer interaction. Therefore, our system supports different procedures of learning compared to other educational systems like LMS. In our system, a lecture is performed through the illustration of some pdf lecture material. The differences of our system from LMS are depicted in *Figure 3*.

The *practice section* is an important part of the training task. In LMS, the section of practice presents successively to students a sequence of exercises. Each student must give a solution to each exercise in the deadline which has been set by the teacher. When the deadline expires, the teacher can download the folder of student answers to exercises and he/she can correct them. After the correction, the teacher can make comments and evaluates the student's performance. In our PITS, the detection of errors is performed automatically by the system. Our PITS is different from all other similar systems in the several ways. First, it provides to students the ability to select the type of practice, which fits better to their personality and capabilities. As such, a student can construct and adapt the training knowledge to his own particular needs. The learning theory of constructivism is followed by our system. It is an effective approach, which is very useful to be supported from an educational system.

The students who want to practice in Prolog programming needs a compiler to detect syntactic errors in his/her program. It is very practical and convenient for a student to be able to practice in the same working environment and in the in the same window. In order to facilitate these aspects, our system has embedded an online interpreter of Prolog. In case that a student gives the wrong answer to an exercise, the system illustrates to the student the correct answer.

Finally, an additional section in our system with useful learning characteristics is the *assessment section*. Each lesson has a test to evaluate the student's performance. Each test has a timer, which determines the maximum time that the student has to complete the test. The majority of PITS, to the best of our knowledge, have not assessment sections for showing the results of the teaching performance and learning effort.

Our PITS allows teachers to monitor the educational process through its available services. To the best of our knowledge, no one of the ITS and PITS that we have studied has this feature. This is a feature available mainly in LMS.

Our PITS, has different interface for students and teachers since they have to perform different tasks. Teachers can assess the performance of each student and visualize other personal details, which are related to the training that he/she has followed. Therefore, teachers are allowed to select information about the use of the system and to inspect the problems set to students. In addition, our system allows teachers to have a special type of communication with each student. As such, our system gives the ability to each student to make a comment and his teacher can see this comment.

We underline that our system can be used in both traditional education and distance learning. In traditional education, it can be used as a tool to enrich the teaching process. In distance learning, it can replace completely the physical presence of teacher.

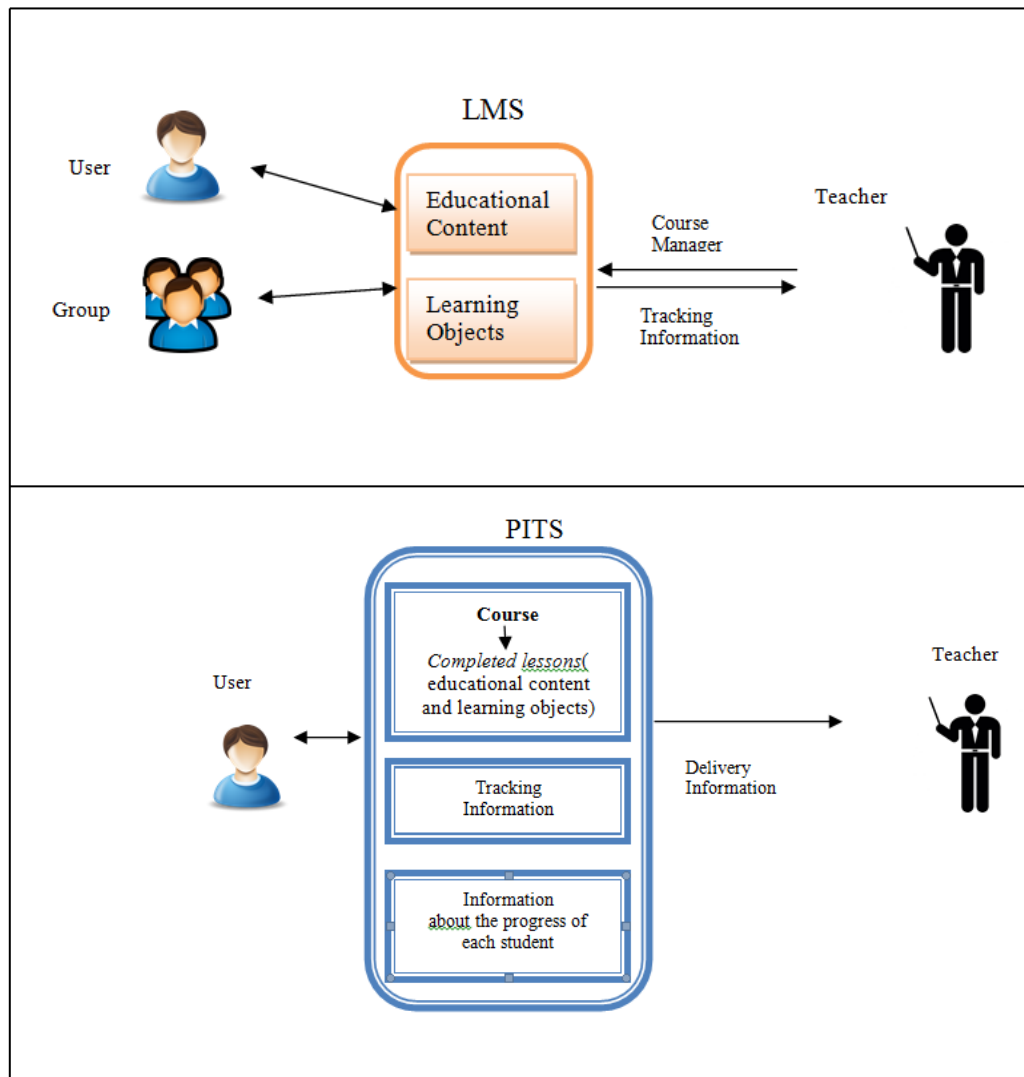


Figure 3: LMS vs PITS

3.4 Comparison of our PITS with Related Systems

The advantages of our PITS are its unique features and its nice functionality, which make it an effective learning system. The construction of our system has been based on three technologies, i.e. Prolog, Html and JavaScript, while the majority of PITS have been developed based on either JavaScript and Html or Prolog and Java or Prolog and Visual Basic. Apart from Prolog, we used Html and JavaScript with the aim to develop a web based system. As such, users can access to our system from any device with internet connection such as PC, laptop, smartphone, tablet, etc. The accessibility of our

system through many different technologies is one of its unique characteristics compared to other PITS, to the best of our knowledge.

There are some PITSs which have classified the possible problems of students into groups of different categories and they provide help to students based on these categories. The tutoring emphasis is based around these categories (Moon-c, 1990) (Looi, 1991). In our system, the exercises have been classified into different categories but the tutoring emphasis is based on complete lessons with *Lectures, Practice and Assessment*.

The majority of PITS detects errors from programs and proposes multiple solutions to these problems (Bielikova & Narvat, 1997).(Moon-c, 1990) (Looi, 1991). Our system provides different techniques and methods to students in order to construct a solution mainly based on providing guidelines and help. At the end, if the student does not succeed to construct a solution of the problem, our system illustrates a solution of the problem.

A PITS developed by Vlahavas (Vlahavas et al, 2009) which aims to be used in distance learning supports images and animation for each lesson. This system is divided into three levels in order to support three types of learners, i.e. beginners, middle level students and advanced students. Our system only in the practice section has a similar feature by supporting these three categories of users. That is, in the practice section, a learner can select an exercise and the type of help he will get it is based on whether he is beginner, middle-level or an advanced student.

Therefore, the unique features of our system are the following: 1) the structure of lessons which consist of three sections, i.e. lecture, practice and assessment sections, 2) tutoring and learning are based on the educational theory of constructivism and the advantages due to this theory and 3) the possibility of teacher to use the system for his own purpose are the characteristics which are different from all other PITS to the best of our knowledge.

4 Architecture of PITS

A PITS typically includes a set of components, which support the students in mastery of their knowledge, skills and ability to apply them. They follow an approach, which is based on the student's performance in scenarios. Our PITS consists of four basic components: 1) *The tutoring or pedagogical module*, 2) *the student or learner module*, 3) *the teacher module* and 4) *the interface*.

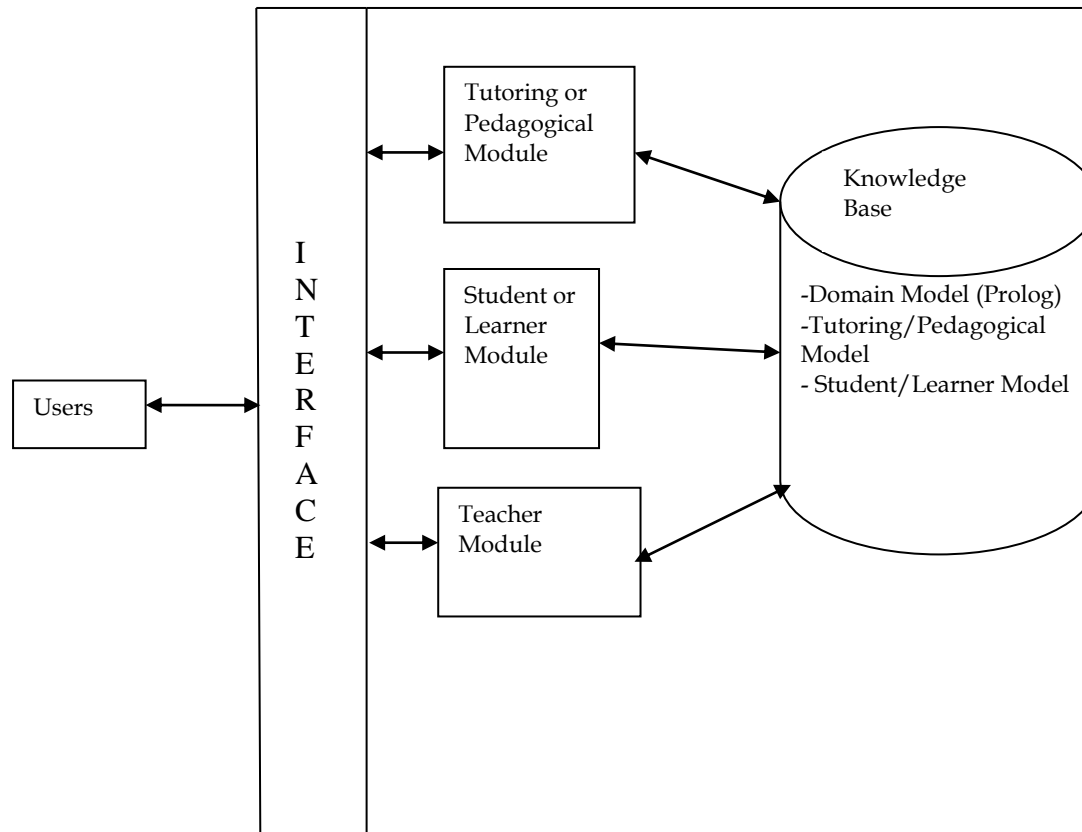


Figure 4:Architecture of our PITS.

1. The ***Tutoring or Pedagogical Module*** provides the knowledge needed to attain teaching goals and plan lessons. It has control over the sequence of learning material and selection of subject material to be presented to the student.
2. The ***Student or Learner Module*** represents cognitive processes (such as information retrieval, calculation and problem solving), meta-cognitive strategies (for example, learning from errors) and psychological attributes (developmental level, learning style, and interests) which are maintained for each learner. The student knowledge processes, known as diagnostics, analyze the behavior of the student.

3. The ***teacher module*** supports the operations of teacher in order to be able to follow the learning progress of each student etc. This is done by using the student model.
4. The ***Interface*** of the system is user-friendly and window-based and has been designed on usability criteria. The users of the system are students and their teacher.

The operations of *tutoring*, *student* and *teacher* modules are supported by the ***knowledge Base***, which contains the *domain model* (i.e. course material of Prolog), the *tutoring or pedagogical model* and the *learner's model*.

4.1 Tutoring or Pedagogical module

The ***tutoring module*** provides the knowledge needed to attain teaching goals and plan lessons. It should have control over the sequence of lessons and selection of subject material to be presented to the student. In addition, it should have response mechanisms to answer learners' questions with appropriate answers. Therefore, when learners need help, in the course of solving a problem or practicing a skill, the tutoring module has to offer several different methods of help. To achieve this, the tutoring model needs to embrace different teaching strategies. As such, it is responsible for selecting teaching goals, and for determining appropriate teaching strategies for learners based on student models, learner's needs and/or preferences, learning experiences, the domain of discourse and the instructional objectives of the intelligent tutoring system.

The PITS has lessons for Prolog programming. For each lesson, there is a schedule, which presents the teaching goals and the teaching methods. Sometimes, it needs to use different methods to achieve different goals. The tutoring module of PITS consists of a set of lessons. Each lesson is divided in 3 sections: 1) *the lecture section*, 2) *the practice section* and 3) *the assessment section*.

The *lecture section* of PITS is supported by several teaching methods, like *question-answer*, *class style lecture* and the method of *programmed and automated instruction*. The section of class lecture style can be enriched by the question-answer teaching method as well. In this way, PITS offers interaction and keeps alive the interest of learner. In any case, the teaching method is guided by an underlying instructional

theory. The theory of this teaching method is the theory of Skinner. The Skinner has been forerunner of teaching machines, he claims that *the question is a stimulus and the answer is a reaction*. If student react and there is positive amplification then student can learn (Orfanos, 2008).

The *practice section* includes different techniques for exercising. The system allows learners to discover which technique is more appropriate for them. The teaching method of *individualized instruction* is applied in this case. It also provides ability to learners to understand what exactly an exercise needs in order to complete this section. The teaching method of *activity* is used in this case. The underling theories of this section are the theories of “*discovery learning*” and “*fact-finding learning*”. These theories support that learners must have critical thinking and discover themselves for having positive results in process of teach (Robin, 2003).

The *assessment section* is the last section of tutoring module. The procedure of assessment starts with a test, which includes exercises in order to diagnose misconceptions. The final evaluation from the test reflects the outcome of the selected teaching methods on the learner and the amount of skills that he/she has gained from these teaching methods. The teaching method of *feedback* is used in this case.

Each of the main modules of our system applies a different set of teaching methods. The correspondence of teaching methods and the main modules of our system are shown in *Table 1*.

Teaching Methods in PITS		
<div> <div>Lecture Module</div> <ul style="list-style-type: none"> Questions Drill and practice Feedback Module Programmed and automated instruction Presentation and lecture </div>	<div> <div>Practice Module</div> <ul style="list-style-type: none"> Activity Direct instruction Drill and practice Feedback Individualized instruction Programmed and automated instruction </div>	<div> <div>Assesment Module</div> <ul style="list-style-type: none"> Activity Feedback </div>

Table 1:*Teaching Methods which are used in this PITS.*

4.2 Student or Learner module

The *student module* represent cognitive processes (such as information retrieval, calculation and problem solving), meta-cognitive strategies (for example, learning from errors) and psychological attributes (developmental level, learning style, and interests) are maintained for each learner.

The student module represents the learner's emerging knowledge and skills. Information such as learning preferences, past learning experiences and advancement may also be relevant in adapting the teaching process. It may also record the learner's errors and misconceptions. The student knowledge processes, known as diagnostics, analyze the behavior of the student.

The student module of PITS maintains information about the student's knowledge, and skills (current and advancing) in the student model. It also stores information about the student's cognitive processes, about the student's learning preferences and/or about past learning experiences. It monitors and assesses student performance and updates student model.

Generally, this module provides dynamic, student-centered tutoring. The learning workflow which is provided by the tutoring module in combination with one provided by the student module make the system a useful educational tool (Robins, Rountree, & Rountree, 2003). The goal of the student module is to make learners independent and autonomous individuals and to give them the opportunity to decide what is important to them.

4.3 The Teacher module

Our system supports in addition teacher's operations, which aim to follow the student's learning progress. More specifically, he can see details about the use of the system from students. The teacher module is using the KB and provides to the teacher the ability 1) to see the students who used the system, 2) to see details about the learning progress for each student-user, 3) to see some statistics which are derived automatically by the system about the use of the system and 4) to read the comments, criticism of students-users about the system and the lessons. In this case, the anonymity of the students is kept in order to have objective comments and criticism.

4.4 Knowledge Base

The *knowledge Base (KB)* contains the *domain model* (i.e. course material of Prolog), the *tutoring or pedagogical model* and the *learner's model*.

- The *domain model* contains knowledge related to the particular domain to be taught.
- The *tutoring model* contains information regarding the order of teaching material. In addition, it has knowledge about guidelines of problem solutions and about alternative solution directions of the same concept.
- The *learner's model* contains knowledge about what the student knows or knowledge that the student has learnt.

Apart from knowledge about Prolog, programming there is information related to the use of the PITS system. The KB contains data in Greek, because the system is used by Greek students in T.E.I of Crete.

4.4.1 The domain model

The *domain model* has knowledge about the subject matter. The domain model in our system has knowledge about Programming in Prolog, which consists of a set of lessons. Each lesson consists from a *lecture part*, a *practice part* and an *assessment part*. The domain model is represented in KB from the following predicates:

<i>schema/2</i>	<i>kb_lesson_practice/6</i>
<i>schemata/1</i>	<i>kb_exersise_practice/2</i>
<i>kb_helpText_practice/3</i>	<i>kb_help_practice/4</i>
<i>kb_lesson/3, kb_stoxoi/2</i>	<i>kb_helpText_practice/3</i>
<i>kb_lesson_lecture/5</i>	<i>schema/2, schemata/1</i>
<i>kb_theory_lecture/2</i>	<i>kb_answer_practice/2</i>
<i>kb_exersise_lecture/2</i>	<i>kb_lesson_assessment/3</i>
<i>kb_answer_lecture/2</i>	
<i>kb_test_assessment/5, kb_exersise_assessment/2 and kb_answer_assessment/2</i>	

These predicates represent the knowledge of the subject domain. The details of the predicates are as follows where each predicate and its arguments are given names, which are related with their role and content.

- **Schemata(SchemataIds)**, where ‘SchemataIds’ is a list of Identifies (Ids) of schemata. For example, `schemata([1,2,3])`.
- **Schema(SchemaId, SchemaRepresentation)**, where ‘SchemaId’ is an Id of a specific schema and ‘SchemaRepresentation’ is the representation of schema as a term of Prolog. For example,
`schema(1,DataStructureConstructionInTheHeadOfClauses)`. Where the variable `DataStructureConstructionInTheHeadOfClauses` has the representation of the programming technique “Construction of Data Structure in the Head of Clauses”. i.e.

‘PredicateName(ControlStructureBase, ConstructedStructureBase).

PredicateName (ControlStructure, FinalConstructedStructure):-

PredicateName(SmallerControlStructure, PartofConstructedStructure),

Construction (PartofConstructedStructure,FinalConstructedStructure).’

- **kb_lesson(LessonId, [LectureId, PracticeId, AssesmentId], GoalId)**. This predicate associates each lesson ‘LessonId’ with the corresponding lecture ‘LectureId’, practice ‘PracticeId’ and assessment AssesmentId’ parts as well as with the learning goal ‘GoalId’. For example, `kb_lesson(les1, [lct1,pra1, ass1], gol1)`.
- **kb_lesson_lecture(LessonId, LectureId ,TheoryId , ExId, [TypeEx, AnswerId])**. This predicate associates each lecture ‘LectureId’ of lesson ‘LessonId’ with a specific theory ‘TheoryId’, an exercise ‘ExId’, the type of exercise ‘TypeEx’ and the answer of exercise ‘AnswerId’. For example, `kb_lesson_lecture(les1, lct1, theoryId1, exId1, [full_example,answerId1])`.
- **kb_theory_lecture(TheoryId, TheoryOfLecture)**. This predicate represents the theory of a lecture. That is, it maps a theory Id TheoryId to a specific theory of a lecture TheoryOfLecture. The theory of a lecture is instance to the variable TheoryOfLecture. For example, `kb_theory_lecture(theoryId1, 'Εισαγωγή: Ένα λογικό πρόγραμμα ..')`.
- **kb_exersise_lecture(ExId, Example)**. This predicate represents an example. That is, it maps an exampleId ExId to a specific example Example. The example is instance to the variable Example. For
example, `kb_exersise_lecture(exId1, 'Να παραστήσετε σε Prolog τη σχέση <Ο Γιάννης είναι πατέρας του Κώστα>.'`

- `kb_answer_lecture(AnswerId, AnswerOfExercise)`. This predicate represents the answer of an exercise. That is, it maps an answer `IdAnswerId` to an answer of an exercise `AnswerOfExercise`. The answer is instance to the variable `AnswerOfExercise`. For example, `kb_answer_lecture(answerId1, 'pateras(yannis,kostas).')`.
- `kb_lesson_practice(LessonId, PracticeId, TypePractice, PracticeExId, HelpId, AnswerId, BibId)`. This predicate represents a full practice session, which has a unique `Id`. That is, for the practice session `PracticeId` of the lesson `LessonId` has the type of practice `TypePractice`, the exercise of practice `PracticeExId`, the help that may be given to the student `HelpId`, the answer of exercise `AnswerId` and bibliography `BibId` for further study. For example, `kb_lesson_practice(les1, pra1, s_b_s, practiceExId1, helpId1, answerId1, bib1)`.
- `kb_exercise_practice(PracticeExId, PracticeExample)`. This predicate represents an example in the practice section. That is, it maps the practice example `Id` `PracticeExId` to the example `PracticeExample`. The example is instance to the variable `PracticeExample`. For example,
`kb_exercise_practice(practiceExId1, 'Σας δίνεται η παρακάτω βάση γνώσης. ?-pateras(kostas,anna).?-mitera(maria,anna). ?-mitera(maria,ari). ?-pateras(kostas,ari). Δημιουργήστε τις ερωτήσεις <Ο Άρης είναι πατέρας του Κώστα?>.'`) . This predicate represents the answer of an exercise.
- `kb_help_practice(HelpId, Help_Level1_Id, Help_Level2_Id, Help_Level3_Id)`. This predicate represents the help that is given to students. There are three levels of help for each problem. Each problem is associated with a help `HelpId` and the three levels of help are represented by the `IdsHelp_Level1_Id, Help_Level2_Id` and `Help_Level3_Id`. If the student does succeed to solve his problem with these three level of help, then he has to repeat the lecture of the lesson. For example, `kb_help_practice(helpId1, Help_Level1_Id1, Help_Level2_Id1, Help_Level3_Id1)`.
- `kb_answer_practice(AnswerId, PracticeAnswer)`. This predicate represents the answer of an exercise in the practice section. That is, each answer has an `Id` `AnswerId` and the answer is instance of the variable `PracticeAnswer`. For example, `kb_answer_practice(answerId1, '?-pateras(aris,kostas).')`.

- `kb_stoxoi(LearningGoal_Id, LearningGoals)`. This predicate represents the learning goals of a lesson. The learning goals of a lesson have a unique Id `LearningGoal_Id` and the learning goals are represented as a string in variable `LearningGoals`. For example, `kb_stoxoi(gol1, "Να κατανοήσουν τα βασικά μέρη ενός προγράμματος Prolog και Να μπορούν να συντάξουν τους όρους της Prolog σε κανόνες παραγωγής.")`.
- `kb_lesson_assessment(LessonId, AssesmentId, [TestId1, TestId2])`. This predicate represents the assessment of a lesson. That is, For the lesson `LessonId` the corresponding assessment is `AssesmentId` and it involves a list of tests `[TestId1, TestId2,...]`. For example, `kb_lesson_assessment(les1, ass1, [testId1])`.
- `kb_test_assessment(TestId, ExId, ExersiceType, AnswerId, Grade)`. This predicate represents the details of an assessment test with Id `TestId`. That is, it has the Id of an exercise `ExId`, the type of exercise `ExersiceType`, the Id of the answer `AnswerId` and the grade of the student `Grade`. For example,
- `kb_test_assessment(testId1, exId1, full_example, answerId1, 4)`.
- `kb_exersice_assessment(ExId, 'TextOfExercise')`. This predicate represents the exercise with Id `ExId` and the text of the exercise follows in a string. For example, `kb_exersice_assessment(exId1, 'Δημιουργήσε τη σχέση στην Prolog ο Γιάννης είναι αδερφός του Κώστα')`.
- `kb_answer_assessment(AnswerId, 'TextOfAnswer')`. This predicate represents the answer of an exercise with Id `AnswerId` and the text of the answer follows in a string. For example, `kb_answer_assessment(answerId1, 'adelfos(yannis, kostas).')`.
- `kb_helpText_practice(HelpId, Help_LevelN_Id, 'TextOfHelpOfLevelN')`. This predicate represents the help of level N `Help_LevelN_Id` where N can be 1, 2 or 3. Details of help of level N follow as a string. For example, `kb_helpText_practice(helpId1, help_Level1_Id1, 'Για να κάνουμε μια ερώτηση στην Prolog χρησιμοποιούμε το σύμβολο <?->')`.
- `kb_helpdesc(Hd_id, 'StudentComments')`. This predicate represents the comments made by the students. Each comment has an Id and the comments are represented as strings. Note that, the Id of the student who made the comment is not registered

by our system, so a student can express his opinion free and not to think that he may have some positive or negative consequence. For

example, `kb_helpdesc(hd1, 'Χρειάζονται περισσότερες ασκήσεις')`.

- `kb_helpdesc_id(ListOfCommentsIds)`. This predicate represents the Ids of comments made by the students. For example, `kb_helpdesc_id([hd1, hd2, hd3])`.
- `kb_bibliog(BibId, 'webLink')`. This predicate represents on-line bibliography which the student may use during the study of this lesson. The representation has the Id of the bibliography `BibId` and the link on the web as string. For example, `kb_bibliog(bib1, 'http://www.swi-prolog.org')`.

4.4.2 The learner's or student's model

The *learner's or student's model* has the knowledge representation for the student. That is, what the student knows or what the student has already learned. The student's model is represented by the following predicates *max_student_id/1*, *students/1*, *student/2*, *kb_course_stud/2*, *kb_helpdesc/2* and *kb_helpdesc_id/1*. The details of these predicates are as follows:

1. `max_student_id(MaxId)`. This predicate has the maximum student identifier `MaxId` that has been assigned. It is used for the creation of student identifiers. For example, `max_student_id(4)`. The next identifier which will be assigned to a student is `std5`.
2. `students(ListOfStudIds)`. The predicate `students/1` is true if `ListOfStudIds` is the list of the identifiers of all students who use our system. For example, `students([std1, std2, std3, std4])`. That is, the students with identifiers `std1`, `std2`, `std3` and `std4` have used our system.
3. `student(Student_Id, [[FirstName, LastName], Dept, StudDeptId])`. The predicate `student/1` has the personal details of each student. It has identifier of the student `Student_Id`, his first and last name `FirstName`, `LastName` respectively, his department `Dept` and his student number `StudDeptId`. For example, `student(std1, [[afrodity, stathaki], ie, 3797])`.
4. `kb_course_stud(Student_Id, [[Lesson_Id, [Lecture_Id, Lect_values], [Practice_Id, Practice_Values], [Assessment_Id, Assessment_Values]])`. This predicate represents the learning details of a student. For each lesson that the student did, it

has the parts of the lesson which he did and the corresponding grades. For the student with Id Student_Id, [[Lesson_Id, [Lecture_Id,Lect_values],[Practice_Id, Practice_Values], [Assessment_Id, Assessment_Values

Where

- Student_Id: stdN where $N > 0$, e.g. std1, std2, etc.
- Lesson_Id: lesN where $N > 0$, e.g. lessid1, lessid2, etc.
- Lecture_Id: lctN, where $N > 0$, e.g. lectid1, lectid2, etc.
- Practice_Id: praN, where $N > 0$, e.g. practid1, practid2, etc
- Assessment_Id: assN, where $N > 0$, e.g. assid1, assid2, etc.
- Lect_values: corr(ect) , incompl(eted), nil_1. **corr** means that the student has answered correctly the exercises of lecture and he has completed that lecture. **incompl** means that he has tried the exercises but he didn't answer them correctly. **nil_1** means that he didn't study the lecture.
- Practice_Values: comph, incomph, compsch, incomp sch compl, incompl. **comph** stands for COMPLETED with Help and answered correctly. **compsch** stands for COMPLETED with the help of a SCHEMA and answered correctly. **compl** stands for COMPLETED without Help and answered correctly. In case for incorrect answer the aforementioned practice values become **incomph**, **incomp sch**, **incompl**.
- Assessment_Values: integers in the interval 0-10 which is the grade scale.

For example, kb_course_stud(std1,

```
[ [les1, [lct,incompl], [pra,comph], [ass,7]],
  [les2, [lct,corr], [pra,nil_p], [ass,6]],
  [les3, [lct,nil_1], [pra,comph], [ass,4]]
 ] ).
```

The student with identifier std1 did the lessons les1, les2 and les3. From lesson les1 he did the lecture section of lesson 1 incomplete. He did the practice section with help and he did the exercise correctly. He did the assessment section and he got grade 7. Similar comments apply for the rest lessons.

4.4.3 The tutoring model

The *tutoring model* must decide what the student has to learn next. It has to decide about the next part of lesson (lecture, practice, assessment) that has to be instructed. In order to take this decision, the system has to use the knowledge representation of the

student model. Due to the constructivism theory some learning decisions, e.g. which practice method to follow, are taken by the student. Our tutoring model sets some restrictions about lessons and parts of lessons, i.e. lecture, practice, assessment, that a student has to follow. This knowledge is encoded implicitly in the next predicates *kb_lesson/3*, *kb_lesson_lecture/5*, *kb_lesson_practice/6* and *kb_lesson_assessment/3*. These predicates have been presented in the domain model. In addition, it uses the predicate *kb_course_stud/2*, which is presented in the student model.

4.5 Interface

Interface design is the area in which classical multimedia authorship tools of PITS are used the most. User Interface requires three types of information to implement the dialogues. These are 1) information about the model, which forms the required explanations to understand the speaker during a dialogue and the required activities to create verbal expressions, 2) the information area that is necessary for the content of communication and 3) the information, which is necessary for the aim of communication. The *user interface module* provides the means for the student to interact with PITS, usually through a graphical user interface and sometimes through a rich simulation of the task domain, which the student is learning.

The tutoring task in our system requires often interaction with its users. This is achieved through special textbox areas where the user writes his answers to the questions of the system. The answers are submitted to the system for processing by pressing special submission buttons. In addition, our system has special buttons which allow its users to explore the system and then to proceed to the next learning step. Moreover, our system has images, which facilitate the tutoring task and make the learning environment friendlier to the users. A user can read details about the use of the system and about the learning procedure through special help texts. Finally, our system offers to its users the possibility to test their ideas and their solutions to an on-line Prolog interpreter before submission in the same window. Therefore, a student in the same environment writes his ideas and solutions, tests them and submits them for final evaluation.

5 Web Features of PITS

Technology enhances the learning process. This is achieved by performing learning activities through information and communication technologies. The existing definitions for technology-enhanced learning are very broad and they change continuously due to the dynamic nature of the fast evolving field of information and communication technologies.

Web 2.0 supports new ways for e- learning. The main reasons for making Web 2.0 so attractive are the following: a) it is free, b) interactive in nature and c) it could be accessible from anywhere. The applications, which are based on Web 2.0, have also the features of instructiveness and accessibility from everywhere. This web application allows learners to interact in a virtual community and be exposed to sounds, images, moving pictures, colors, and text that are characterized to be ideal for a classroom where different subjects can be taught. This is the main reason why we decided to develop a web-based Prolog Intelligent Tutoring System. Certainly, our system can be enhanced to support lessons for other topics as well as.

Prolog is extremely well suited for developing web applications. Web pages are naturally represented as Prolog terms and can be easily created, inspected and processed recursively. The purpose of this chapter is to describe a modular and customizable architecture for a WWW server, which employs Prolog programs to handle requests from HTML forms. It is our intention to provide as much details as necessary for the appropriate adaptation and customization of our architecture to other similar contexts.

Our web architecture of PITS consists of the following components:

- *A web application running on a Prolog server.* A Prolog CGI (CGI stands for Common Gateway Interface) library allows the Prolog web server to execute programs (Prolog applications like our application) as console applications running on a server, which can generate web pages dynamically. An HTML form is a web page which acts as an interface to remote users who want to pose requests to the web server.

- *A set of client-server modules.* In order to process the requests of the users of our system, a Prolog program is run initially in order to parse and tokenize the actual input and only then the requests are handled by the Prolog server.

An HTML form is an interface between users and web sites. HTML forms come in many different formats such as click-on buttons, scroll-down menus and areas for typing actual text, etc. HTML forms can also have combinations of these formats. In our system we have a Prolog server which is responsible for the reception and checking of the HTTP requests (provided by the HTML form) and for providing the actual services of the web site to the user's requests. The user's interaction is handled by the client-server modules.

6 Detailed Presentation of the PITS Components

In this section, we present the details of the components of PITS. Our PITS can be used in distance or real time learning. For this reason, the functionality of our PITS is based on a robust learning theory (constructivism) and a modern teaching methodology which they support a student to construct his/her own knowledge in the specific subject. The subject in this case is “Programming in Prolog”.

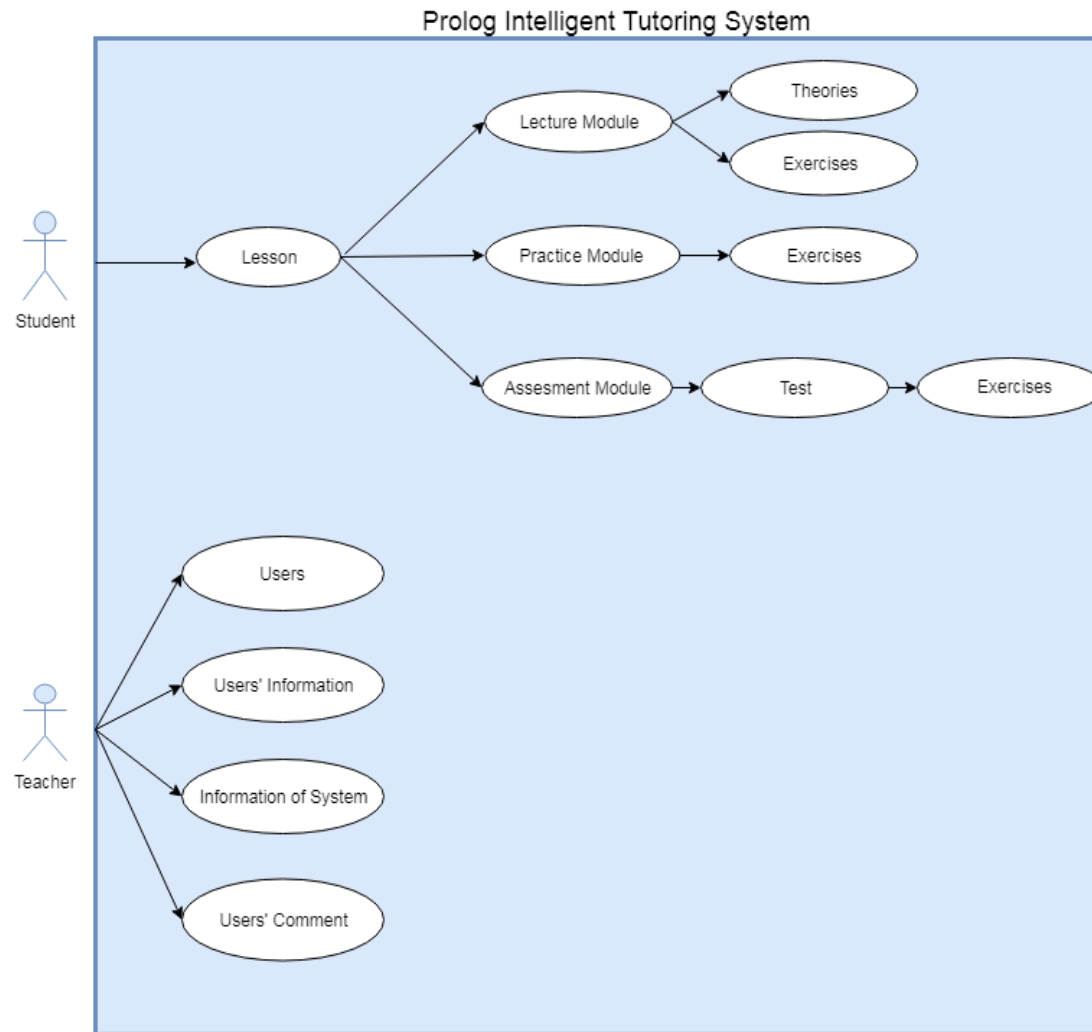


Figure 5: Use case diagram of PITS.

In real class, a teacher must plan the lessons before teaching. The planning of each lesson includes three parts: 1) the *lecture*, which is referred to as the theory, 2) the exercise of practice in which there are some exercises, which are based on theory. In this case, the teaching method of Drill and Practice is applied. In addition, 3) the *assessment* which includes some exercises. These exercises are based on theory and they are used by the teacher in order to see his/her student's progress. On this premise,

this system is divided into 3 modules: a) the *Lecture Module*, b) the *Practice Module* and c) the *Assessment Module*, which are intended to be used by the students. In addition, there is the *Teacher Module*, which is intended to be used by the teacher. A use case diagram of our PITS is shown in **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε..**

6.1 The Lecture Module

The lecture is a teaching method, which is a way of introduction of students to a particular field of study (Westwood, 2008). Some critics consider lecturing as a method of communication, which relies upon passive learning. Lecturing is often constructed based on active learning. This active learning happens when lecturing contains teaching methods like pictures, question-answer, video etc.

The main goal of this thesis is to create a system, which introduces the student in Prolog programming and offers interaction, keeping alive the interest of the learner. According to the theory of lecturing, we try to simulate the procedure of lecturing into PITS. The lecture planning of the lecture module follows the tree structure shown in *Figure 6*.

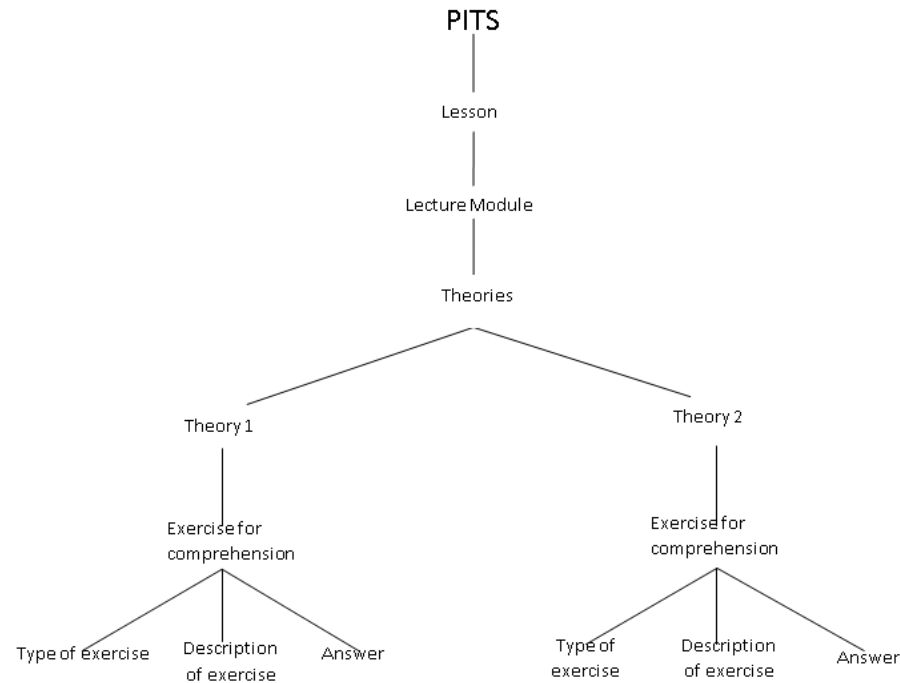


Figure 6: Structure of Lecture Module in PITS.

Upon the log in of the learner into the PITS, the system provides three choices. The learner can decide from which lesson he wants to start and he/she has to select one from three choices. Each lesson has a lecture, a practice and an assessment part. In case that the learner wants to have successfully completed the lesson, he/she has to do the test of assessment with success.

In addition, if he/she select a lecture and he/she wants to complete this section, it must complete the procedure, which is illustrated in the diagram in Figure 8. Each lecture of lesson includes a theory, which is separated into two parts. The first part is presented as a pdf file. The teaching method of “Presentation and lecture” is used in this case. If the learner thinks that he has understood the topic of the lecture, he can push the button “ok”. After that, the system displays a question, which helps the learner to understand the theory. If the learner answers correctly the question, he can go to the next part of theory. In case, the learner’s answers are wrong, the system provides the correct answer. Once he understands his mistake, he can push the button “ok” and he can go to the next part of theory.

In any case, the system observes and records the user’s actions in order to give information to the teacher about the progress of the student in each lesson.

6.2 The Practice Module

The Practice Module is the most important part of an educational system. In this section, the learner exercises in the taught theory. If a student does not select the lecture module, he can also practice in the practice section, which follows, considering to already having the knowledge of the corresponding lesson.

In addition, the learner can select among *hint-based practice*, *advanced practice* and *open-practice*. In the hint-based practice, many hints guide the learner to complete his task. In the schema-based approach, example schemata are suggested for completing the task. In addition, examples are shown and guidelines are given to the end-user on constructing accordingly his/her programs. Finally, in the open-practice approach no help is provided to the learner. The learner has to identify possible strategies by himself for solving the exercises. The structure of the practice module is shown in *Figure 7*.

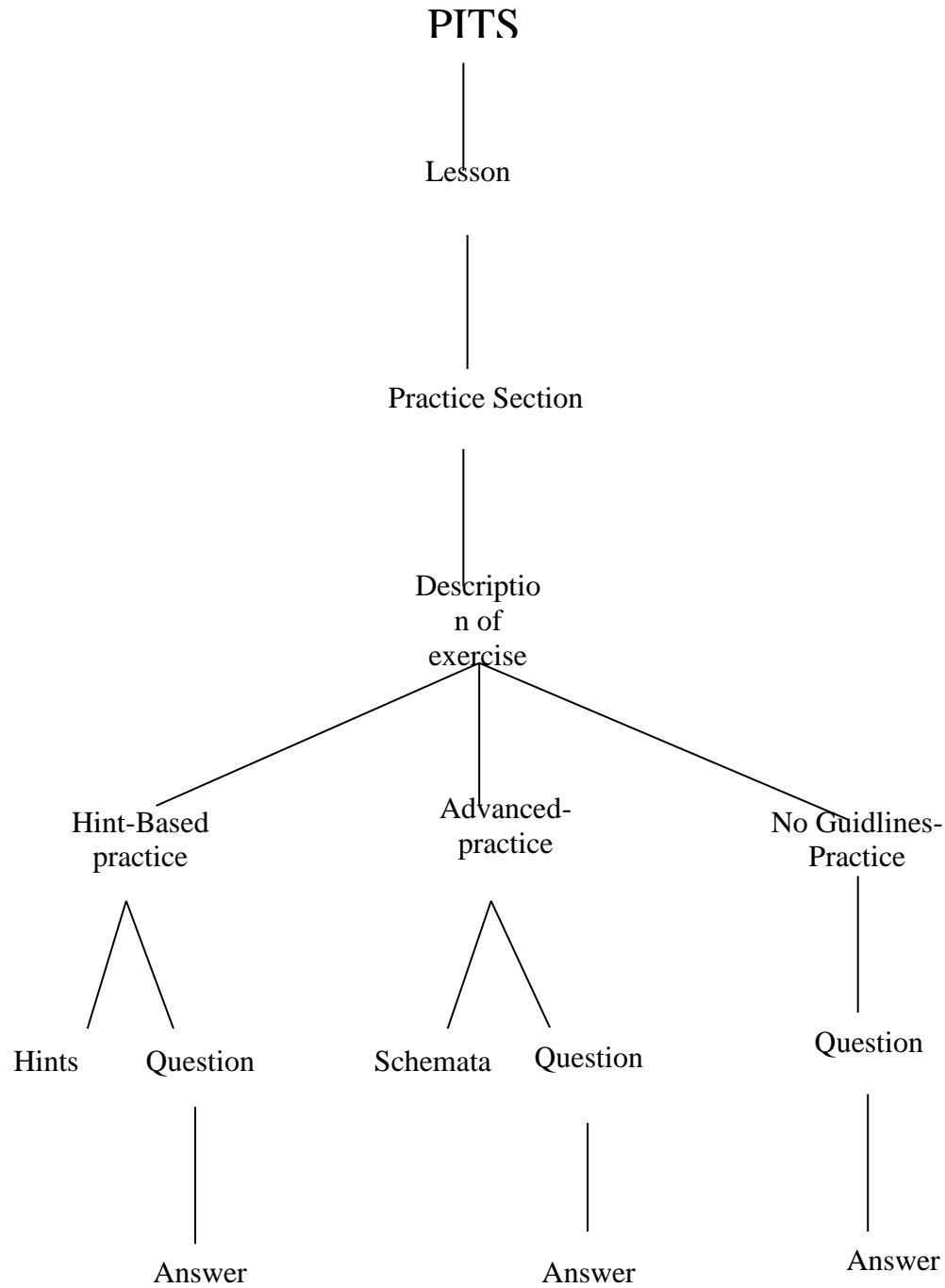


Figure 7: Structure of Practice Module in PITS.

Hint-based practice is a practice method, which is intended to be used by beginners. Firstly, a text is presented to the student, describing the exercise to be performed. After that, the working environment is separated into two parts. The left part of the screen displays a textbox in which the learner has to give an answer. In

addition, there is a button named “Help” which is displayed under the answer textbox. At any time, the user can press this button and a text is displayed containing information and some hints. Those have the aim to help the student to complete the exercise. The teaching methods of *programmed and automated instruction* and the one of *direct instruction* are applied in this practice method.

At the right part of screen, there is an online compiler of Prolog. Therefore, the student does not need to change working environment to compile and run his Prolog code. In fact, our PITS facilitate Prolog code writing, compiling, and debugging. Therefore, students can confirm themselves for the correctness of the solution that they have submitted to the system.

As soon as a student provides a solution for the exercise into the textbox and press the button “Submit” the system presents the feedback. If the solution which submitted by the student is correct, the system suggest to continue in the next section. If the solution is not correct, the system provides positive reinforcement and gives the correct answer to student and suggests him to study again the theory of lesson or he/she may continue in the next section. In any case, the choice for the next step belongs to the learner.

Schema-based practice is a method of system practice, which is intended to be used by beginners, intermediate and advanced students. Many Prolog programs share a common underlying structure. Such classes of programs exemplify general structural characteristics. In programming, a “program schema” or “schema” describes the basic features of a set of programs, which have the same underlying structure. In addition, some Prolog programming techniques can be represented by Program schemata. This method of practice aims to teach programming by program schemata construction. Program schemata construction is illustrated through some examples. The interested reader can read more in (Robertson, 2005),(Fragaki, 2006), (Marakakis., 2014). The teaching methods of *programmed and automated instruction* and the one of *direct instruction* are applied in this practice method.

A Prolog Programming Environment can be built around a set of basic Prolog schemata. A student selects a Prolog schema from the Prolog schema library and instantiates its schema variables. The Knowledge Module contains a library of Prolog schemata including the basic Prolog ones which represent programming techniques.

After a particular program schema is introduced, students are required to instantiate the schema as a programming exercise. There are two benefits of schema based learning approach:

1. Using program schemata in programming supports understanding of basic programming concepts (Bielikova, 2003).
2. Teaching with schemata provides the tutor with more accurate estimation of the student's problem solving capability (Bielikova, 2003).

In our PITS, the Prolog schema library of the Knowledge Module in its current prototype form has among the other schemata the two basic ones which are abstractions of two Prolog programming techniques. These two techniques help students to write recursive Prolog Programs. These programming techniques are taught in the form of program schemata. These two techniques are named “structure construction in the head of a clause” and “structure construction in the body of a clause” (Marakakis, 1997), (Robertson, 2005). These programming abstractions help a student to understand the essential components of the aforementioned techniques. These two schemata are shown in Program Schema 1 and Program Schema 2 .

```
PredicateName (ControlStructure, FinalConstructedStructure) :-
    NewPredicateName(ControlStructure, ConstructedStructureBase,
        FinalConstructedStructure).

NewPredicateName(ControlStructureBase, PartofConstructedStructure,
    FinalConstructedStructure) :-
    FinalConstructedStructure=PartofConstructedStructure.
NewPredicateName(ControlStructure, PartofConstructedStructure,
    FinalConstructedStructure) :-
    Construction (PartofConstructedStructure,NewPartofConstructedStructure),
    NewPredicateName(SmallerControlStructure,NewPartofConstructedStructure,
        FinalConstructedStructure).
```

Program Schema 1: Structure construction in the body of a clause.

```
PredicateName(ControlStructureBase, ConstructedStructureBase).
PredicateName (ControlStructure, FinalConstructedStructure):-
    PredicateName(SmallerControlStructure,
        PartofConstructedStructure),
    Construction (PartofConstructedStructure,FinalConstructedStructure).
```

Program Schema 2: Structure construction in the head of a clause.

deleteAll (_,[],[]). deleteAll (X, [X Tail1], Ended):- deleteAll (X,Tail1, Partial), Ended= Partial. deleteAll (X, [Y Tail1], Ended):- deleteAll (X,Tail1, Partial), Ended= [Y Partial].	PredicateName (ControlStructureBase, ConstructedStructureBase). PredicateName (ControlStructure, FinalConstructedStructure):- PredicateName (SmallerControlStructure, PartofConstructedStructure), Construction (PartofConstructedStructure,Fin alConstructedStructure).
---	---

Schema Instance Paradigm 1: Delete all of repetitions with using technique “structure construction in the head of a clause”.

deleteAll (X, L1, L2):- NewPredicate (X, L1, [],L2). NewPredicate (_, [],Partial, Ended):- Ended= Partial. NewPredicate (X, [X Tail1],Partial, Ended):- Partial= NewPartial, NewPredicate (X, Tail1, NewPartial, Ended). NewPredicate (X, [Y Tail1],Partial, Ended):- Append (Partial, [Y], NewPartial), NewPredicate (X, Tail1, NewPartial, Ended).	PredicateName (ControlStructure, FinalConstructedStructure) :- NewPredicateName (ControlStructure, ConstructedStructureBase, FinalConstructedStructure). NewPredicateName (ControlStructureBase, PartofConstructedStructure, FinalConstructedStructure) :- FinalConstructedStructure=PartofConstructedStr ucture. NewPredicateName (ControlStructure, PartofConstructedStructure, FinalConstructedStructure) :- Construction (PartofConstructedStructure, NewPartofConstructedStructure), NewPredicateName (SmallerControlStruct ure,NewPartofConstructedStructure, FinalConstructedStructure)
--	---

Schema Instance Paradigm 2: Delete all of repetitions with using technique “structure construction in the body of a clause”.

For a student using one of these two schemas, our system provides the ability to construct structured recursive Prolog programs. After the selection by the student one

of these schemata, our system displays the working environment. After reading the description of the exercise, the student is allowed to experiment with his program by using the online compiler of Prolog. He/She is expected to construct a program based on the selected schema. Subsequently, the student submits his program by pressing the appropriate button, i.e. submit, and the system displays the result of his effort. If the submitted solution is correct, the system suggests continuing in the next section. If the solution is not correct, the system provides positive reinforcement and gives him the correct answer and it suggests him to study again the theory of the lesson or to continue in the next section. The choice for the next action is always upon to the student. The system makes simply suggestions to the learners. The predicate `deleteAll(X,L1,L2)` is for example true if list L2 is list L1 without the occurrences of element X. This predicate has been implemented by using the schemata of these two programming techniques. This is shown in Schema Instance Paradigm 1 and Schema Instance Paradigm 2.

No Guidelines Practice is a type of practice, which is used by advanced students. In this section, there is not programming help. The student-user can read the description of the allocated exercise and he can try with his/her own skills without help to write the required Prolog code. If he thinks that he cannot do the allocated exercise in this type of practice, he can either repeat the previous lectures in order to assimilate the subject or he can select another type practice from the available ones. In case, that the student-user wants to continue with this type of practice he is offered by the system a working environment which is separated in two screen's parts. One screen part has the environment for constructing his/her solution to the allocated exercise and the other screen part has the compiler environment for running his Prolog code. The student can give the answer in a text box. Subsequently, he/she has to press the submit button for the submission of his/her solution of the problem and the system will display the result of his effort. If the solution submitted by the student is correct, the system suggests continuing in the following sections. If the solution is not correct, the system provides positive reinforcement and illustrates the correct answer of the problem and provides a suggestion to the student either for studying further the theory of this lesson or to continue in the following sections.

6.3 The Assessment Module

In real time learning, after the lecture and practice of a lesson, the teacher gives a test to his students. The goal of the test is mostly educational which allows the teacher to see the students' progress in the lesson. Our Tutoring System aims to simulate a teacher's task in real time. The Assessment Module is a key feature of our system.

This PITS has lessons for its learners. Each lesson includes lecture, practice and assessment. In addition, the assessment part includes a test, which contains some exercises. The student has to do these exercises with success, if he wants to go ahead to the next lesson. In fact, a student has to answer correctly at least half of test's exercises in 45 minutes. If the available time runs out, then the system considers that the lesson has not been completed and goes to the beginning. If the student achieves the goal and his score is 50% or more, he can go in the next lesson. If, his score is under 50%, the system suggests to the student to repeat the lesson and he can do the test after the repetition. In any case, the student knows his score. The structure of the assessment module is shown in *Figure 8*.

The system provides to its student-users evaluation of their learning performance. On the other hand, each student-user is given the possibility to evaluate the system. After completion of the student's evaluation learning performance, the student can make comments about the system's functionality and about the quality of lessons. These comments are saved in the knowledge base and they are shown to corresponding teachers in order to be used for upgrading the system. The system asks the student to complete a specific questionnaire, which is used for the evaluation of the system. In this way, the system could become more useful as an educational tool to both learners and teachers.

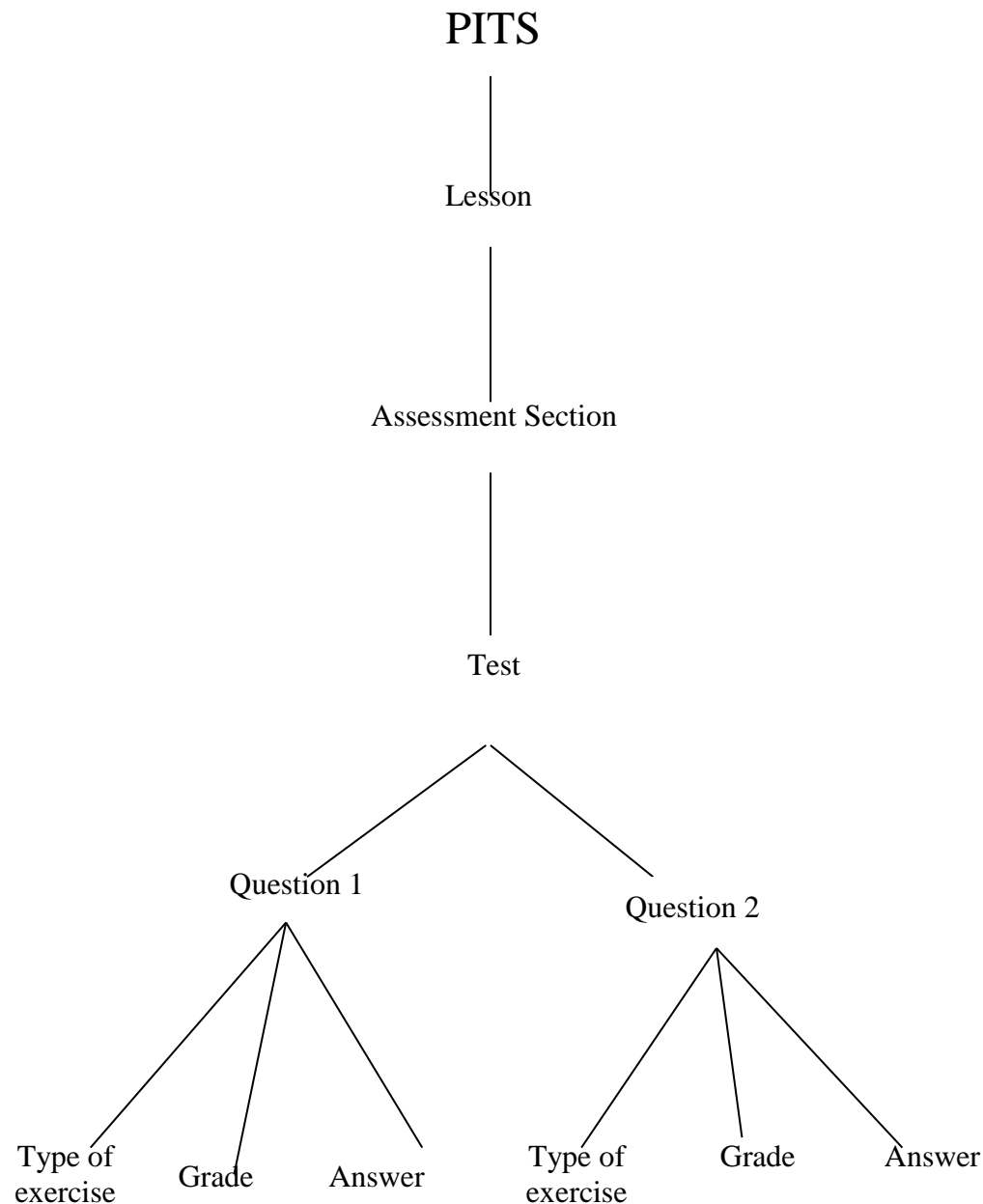


Figure 8: Structure of Assessment Module in PITS.

6.4 The Teacher Module

Each tutoring system can be used in real time learning or distance learning. In both cases, students and teachers can use it. This applies for our system as well. In real time learning, it could be used by the teacher to enrich the educational process. For example, when the teacher completes the lecture of Prolog programming, he could encourage his student to practice with our system. In this case, a teacher can supervise his/her students from his monitor. In distance learning, our system could be used by students

for comprehensive training on specific lessons. In this case, a teacher can have a complete view of students' progress through the learning process without having to intervene into educational procedure. The structure of the teacher module is illustrated in *Figure 9*.

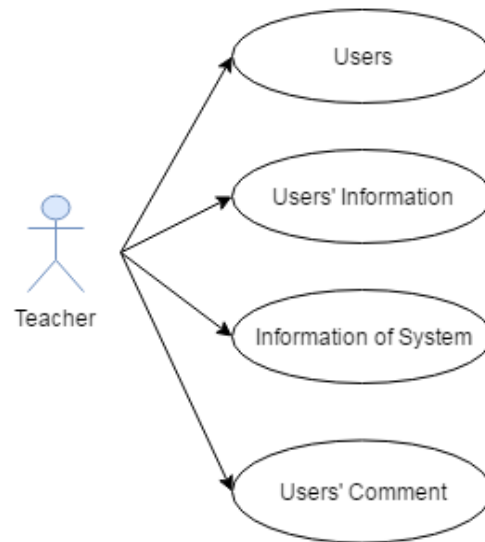


Figure 9: *Structure of Teacher Module in PITS.*

The main idea for our system is that a student is given the possibility to study the theory of a lesson; he/she can also practice if he/she wishes so but in any case, he/she must complete successfully the assessment part of a lesson in order to complete the given lecture successfully. The teacher has given the possibility to detect the shortcomings of his students, to observe students' evaluation and to check the students' evaluation. The teacher, who is in charge of students' progress, has the possibility to log in into the system as teacher. Teachers have a different working environment. This environment provides to a teacher four selections, as shown in *Figure 9*. A teacher can do the following tasks.

- 1) He can see the students who used the system.
- 2) He can see details about the learning progress for each student-user.
- 3) He can see some statistics, which are derived automatically by the system about the use of the system from its students-users and about the learning activities of each individual student-user. These details help for the improvement and for the upgrade of the system and of the lessons.

- 4) He can read the comments of students-users about the system and the lessons.

6.5 Our System and the Principles of Constructivism Theory

The constructivism is a theory describing how the learning process is achieved and it has the principles of 1) *interaction*, 2) *construction*, 3) *assimilation* and 4) *knowledge consolidation*. The functionality of our system has been designed with these principles in mind.

- 1) *Interaction*: This is a feature, which characterizes our system. The students, in each section of our system, participate actively in the learning process. In each section, there are questions, which must be answered by the students. The browsing into our PITS depends on students' interaction with the system. For example, if a student has completed the section of a lecture, he/she has to press the button "Return" to continue the learning process and to complete the lesson.
- 2) *Construction*: A student develops his knowledge by selecting the learning method, which fits to his particular needs. The student has to decide about 1) the lesson from which to start the learning process, 2) the type of practice method to follow for a particular theory of a lesson and 3) the completion of the learning process in order to test his performance in each lesson. In this way, a student develops his knowledge in an effective way.
- 3) *Assimilation*: The student assimilates the constructed knowledge by performing practice into Prolog Programming through the practice section of our system. The student is allowed to select the practice method, which is appropriate for his needs.
- 4) *Knowledge consolidation*: Interaction, construction and assimilation of knowledge are the preconditions in order to have knowledge consolidation. Our system archives knowledge consolidation by providing a test in the assessment section for each lesson. Each test consists of a set of exercises, which is based on the theory of the lesson. If a student achieves score 50% or more, the knowledge consolidation process is assumed to have completed successfully. If, his score is under 50%, he has to begin again the construction and assimilation of knowledge.

7 Sample Sessions of PITS

This chapter includes some samples sessions, which describe the functionality of system. The functionality of system separates into two categories, i.e. functions which support the tasks of students and functions, which support the tasks of teacher. The tasks of a student include the use of lecture, practice and assessment functions. Each sample describes how the system is presented in learner or student.

7.1 Sample Sessions of Lecture Section

The learning process starts by studying the lectures of a particular lesson. Initially, a student has to select the lecture section subsystem in order to begin his/her study. The lecture section provides the theory of each lesson and some questions, which aim to identify if the student comprehended the selected lesson.

Θεωρία

Λήψη

Κατασκευή δομής στην κεφαλή μιας πρότασης. Είναι μια προγραμματιστική τεχνική που βοηθά στην κατασκευή προγραμμάτων σε Prolog. Το αποτέλεσμα αυτής της τεχνικής συσσωρεύεται στην επαγωγική δομή δεδομένων χρησιμοποιώντας τη στοίβα της Prolog. Η επαγωγική δομή που δημιουργείται σε κάθε αναδρομική κλήση είναι ημιτελής και δεν υπάρχει πρόσβαση στο τμήμα που έχει ήδη κατασκευαστεί. Ένα τμήμα της αναμένεται να ολοκληρωθεί από την αναδρομική επεξεργασία των υπόλοιπων στοιχείων της επαγωγικής δομής η οποία ελέγχει την αναδρομή.

les3theory1.pdf 2 / 3

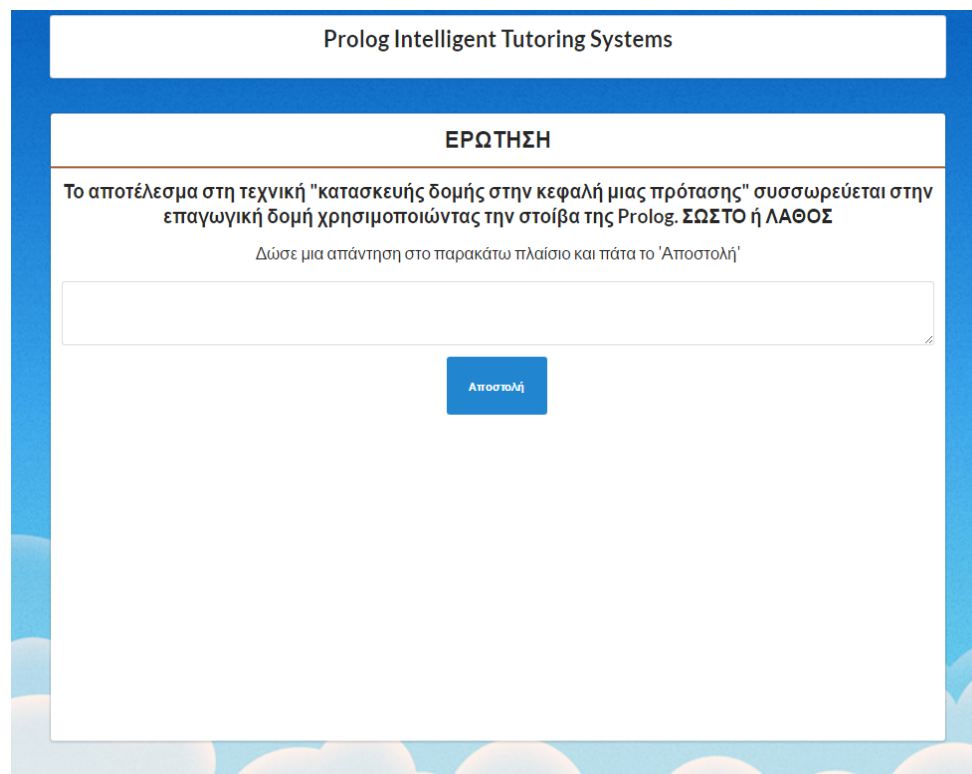
3.4. Προγραμματιστικές Τεχνικές

- ❑ **Τεχνική: Κατασκευή δομής στην κεφαλή μιας πρότασης.**
- ❑ Σε αυτή τη προγραμματιστική τεχνική το **αποτέλεσμα συσσωρεύεται στην επαγωγική δομή** (ή τύπο) δεδομένων χρησιμοποιώντας την στοίβα της Prolog.
- ❑ Η επαγωγική δομή που δημιουργείται σε κάθε αναδρομική κλήση είναι **ημιτελής** και
 - δεν υπάρχει πρόσβαση στο τμήμα που έχει ήδη κατασκευαστεί,
 - ένα τμήμα της αναμένεται να ολοκληρωθεί από την αναδρομική επεξεργασία των υπόλοιπων στοιχείων της επαγωγικής δομής η οποία ελέγχει την αναδρομή.
- ❑ Όταν ολοκληρωθεί η αναδρομική επεξεργασία των στοιχείων της επαγωγικής δομής εισόδου,
 - η **Prolog συνθέτει το τελικό αποτέλεσμα** το οποίο και επιστρέφει μέσω της επαγωγικής δομής η οποία κατασκευάζεται.

Εργαστήριο Λογικού Προγραμματισμού 11 Δρ Μανώλης Μαρακάκης

Figure 10: First page of lecture with theory.

The system allows to students to download as pdf file the presentation of the theory of the lecture. When a student has completed the reading of the reading part of lecture of theory, he can press the button “Continue” in order to proceed to the question part of lecture which verifies the student has comprehended the reading part of theory. At this stage, the system displays the question and the student has to submit an answer. If the answer is correct, the system displays the next theory of lecture, otherwise it presents to the student the correct answer. Then, he can proceed to the next part of theory.



The screenshot shows a web interface for 'Prolog Intelligent Tutoring Systems'. At the top, there is a blue header with the text 'Prolog Intelligent Tutoring Systems'. Below this is a white box with the title 'ΕΡΩΤΗΣΗ' (Question) in bold. The question text in Greek asks about the result of a technical construction in the head of a proposition and its use in Prolog. Below the question is a large text input field and a blue button labeled 'Αποστολή' (Submit). The interface has a blue border and a decorative cloud pattern at the bottom.

Figure 11: Page of lecture with question.

After the completion of a lecture, i.e. the student has completed the reading part of a lecture and has answered (correctly or incorrectly) all questions for assimilation of the theory, then the system updates the KB with all details about the performance of the student in the lecture section.

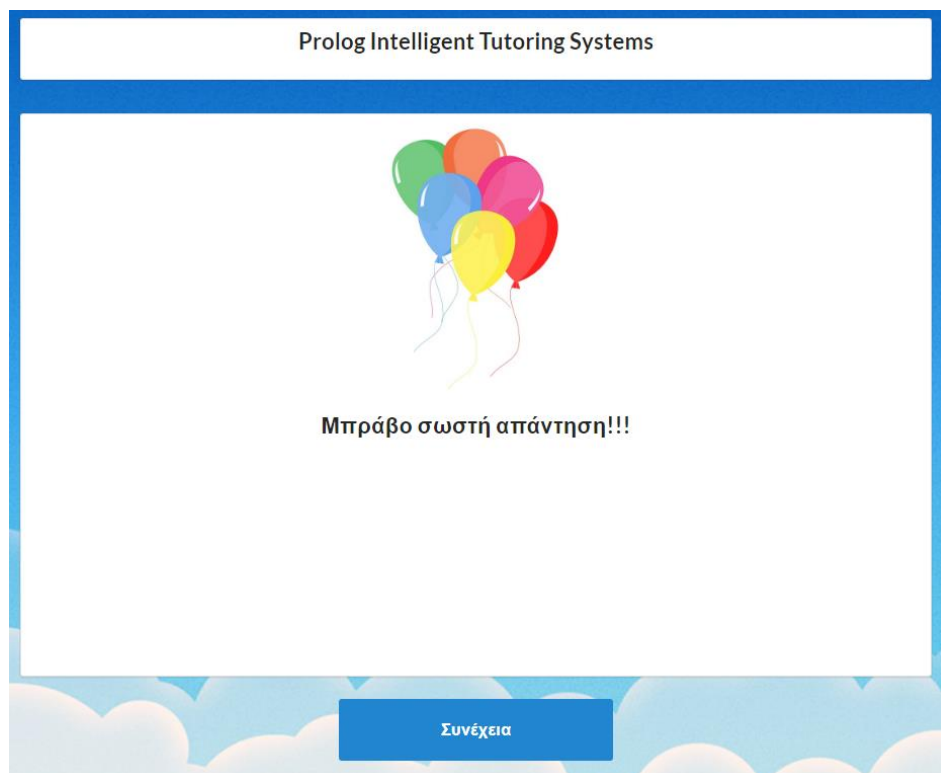


Figure 12: Page of feedback when the answer of student is correct.

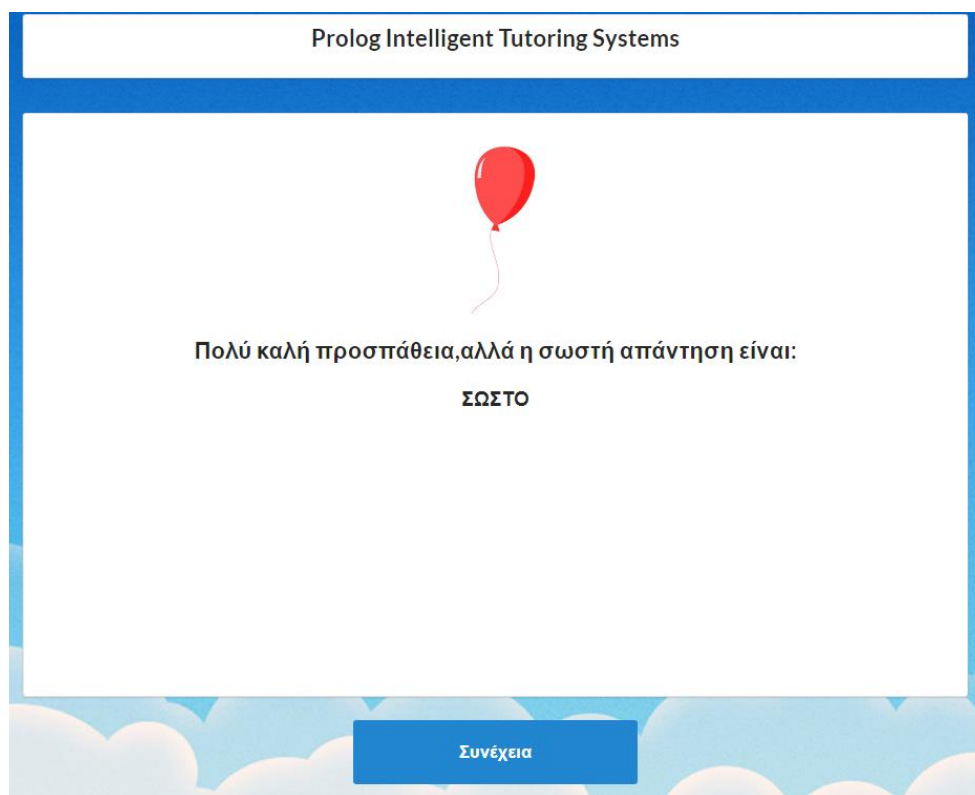


Figure 13: Page of feedback when the answer of student is not correct.

Then, the system presents the possibility for selection of any of the three sections. If the student wants to repeat the lecture, he presses the button “Lecture” and he studies again the lecture, otherwise he can select “Practice” or “Assesment”. In case, that a student wants to exit from the system the lesson is considered uncompleted. In this case, when the student logs in again into the system he has to continue on the same lesson where he stopped.

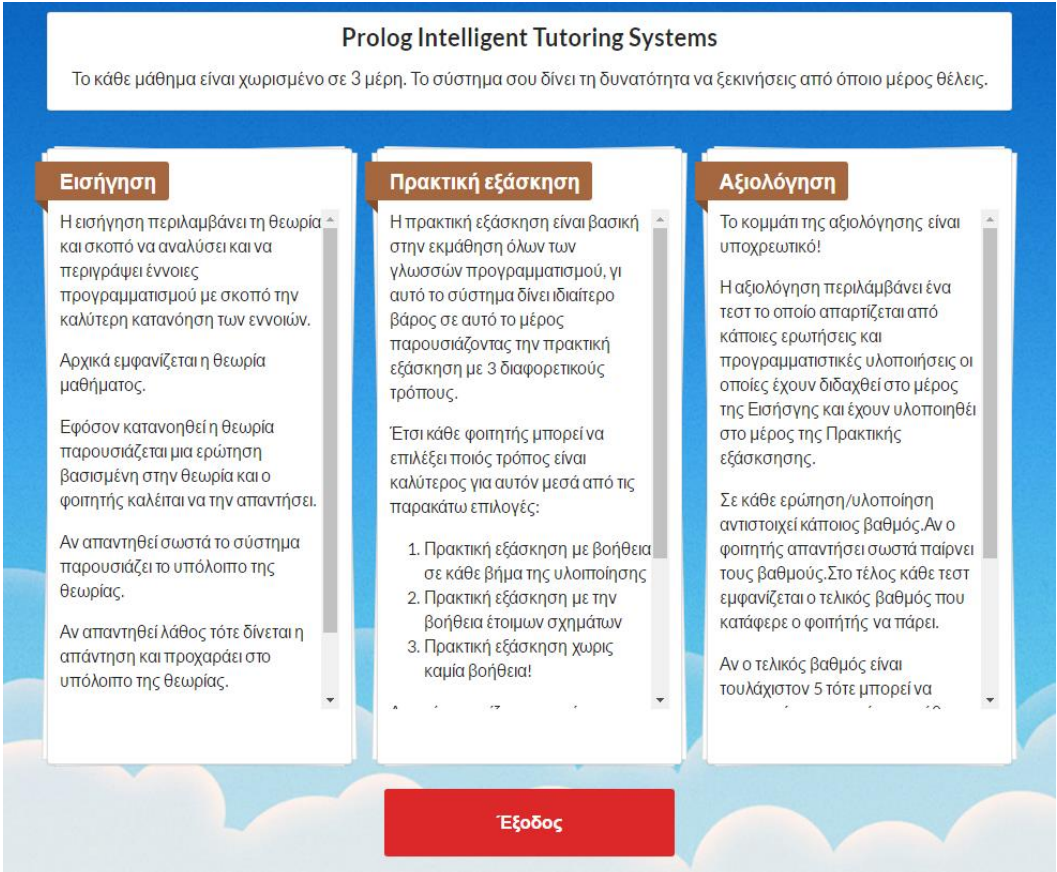


Figure 14: Page of selections: Lecture, Practice and Assessment.

7.2 Sample Sessions of Practice Section

The system provides the option to students to select the method of practice, which fits better into their capabilities. Therefore, a student is allowed to adapt the knowledge in his own particular needs and requirements. After the selection of the practice section by the student, the system presents the three options of practice. That is, 1) Step-by-step, 2) Schema-based and 3) No Guidelines.

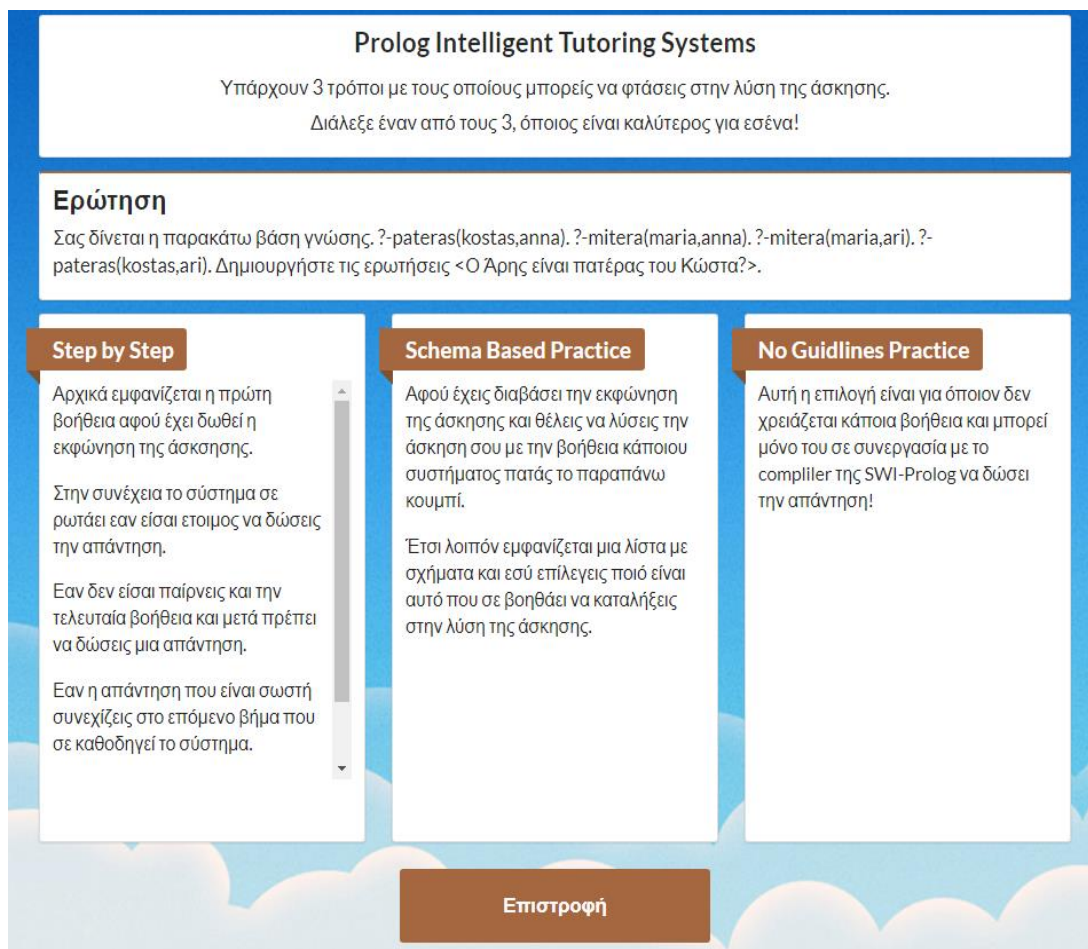


Figure 15: Page of practice selections: Step by step, Schema and Prolog.

Step by step is a method of practice, which is offered to beginners of Prolog. The working environment, which supports this method, separates the screen into left and right part. In the left part of screen a text is presented which describes the exercise, next a textbox is displayed in which the learner can give his answer. In addition, in the left part of screen there is a button “Help”. This button is used to provide hints and detailed help to the student. The aim of this method is to help the student as much as possible in order to be able to complete the exercise. This is the main characteristic of this method of practice.

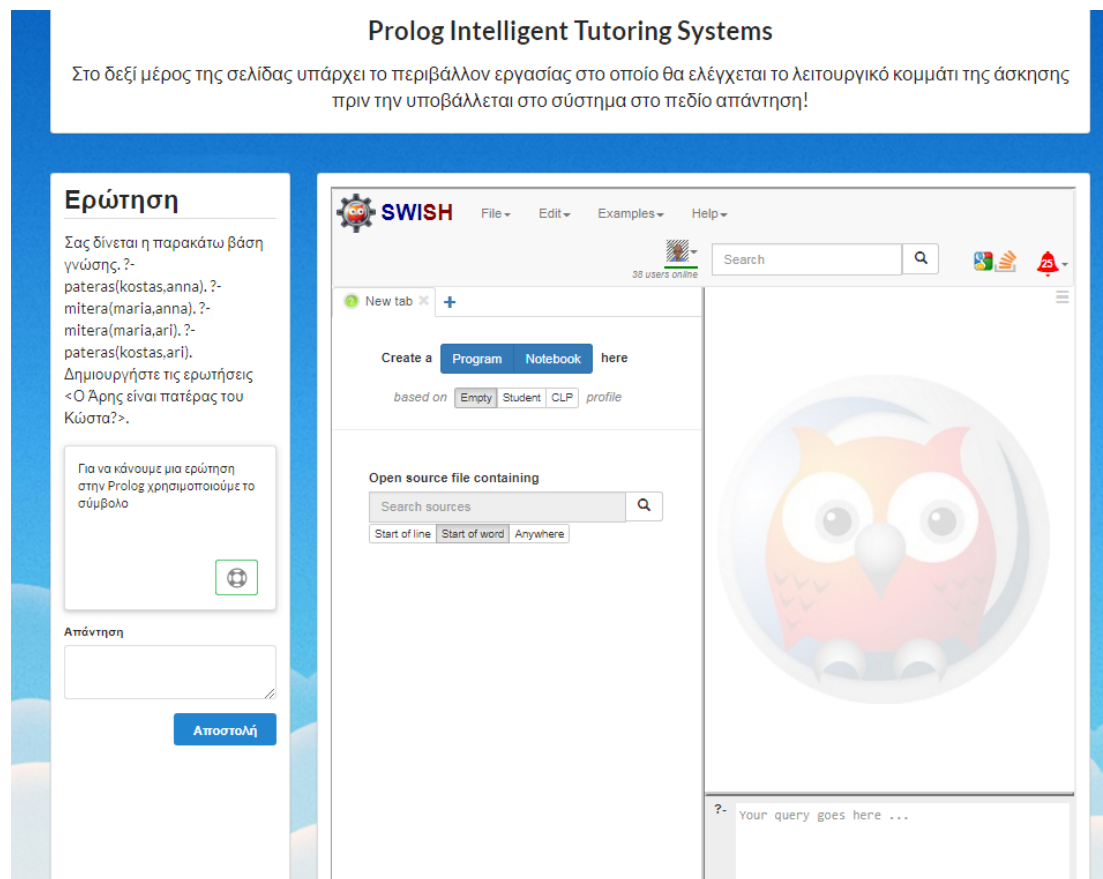


Figure 16: Page of selection practice with help: Step by step.

Schema-based practice is a method of practice, which is intended to be used from all types of users, i.e. beginners, intermediate and advanced, because it is essential for writing correct and well-structured Prolog programs. The student selects a Prolog schema from the library of schemata of our system and proceeds to the instantiation of the schema. The Knowledge Module contains a library of Prolog schemata including the basic Prolog schemata of the two programming techniques.

In this method, the student is given only the schema that he has to follow. During the practice session, he can test for the correctness of his program by using the online compiler, which is available in his working environment. At the end, the student submits his answer to the system, which automatically checks for its correctness. Subsequently, the system displays the result of his effort. If the submitted solution is correct, the system suggests continuing in the next section of lesson. If the solution is not correct, the system provides positive reinforcement and gives the correct answer to the student and suggests him to read again the theory of lesson or to continue in the following section of the same lesson.

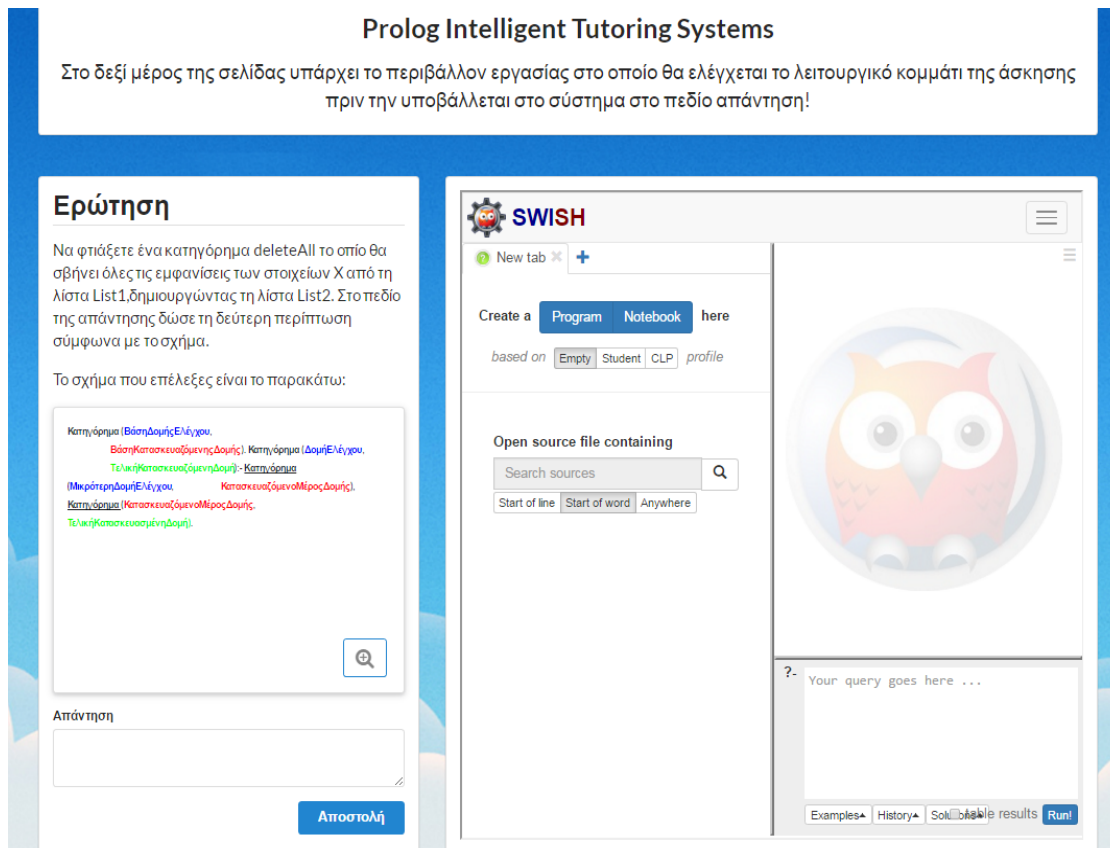


Figure 17: Page of selection practice with schema based help: Schema.

No Guidelines practice is a method of practice, which is intended to be used from advanced students. The student is given the problem specification and he is expected to complete it without any help. The only tool that he can use is the online compiler for running his program. At the end, the student submits his answer to the system, which automatically checks for its correctness. In this case, the system follows the same steps. That is, the system displays the correct answer in case of error and the student can continue in the following section of the same lesson.

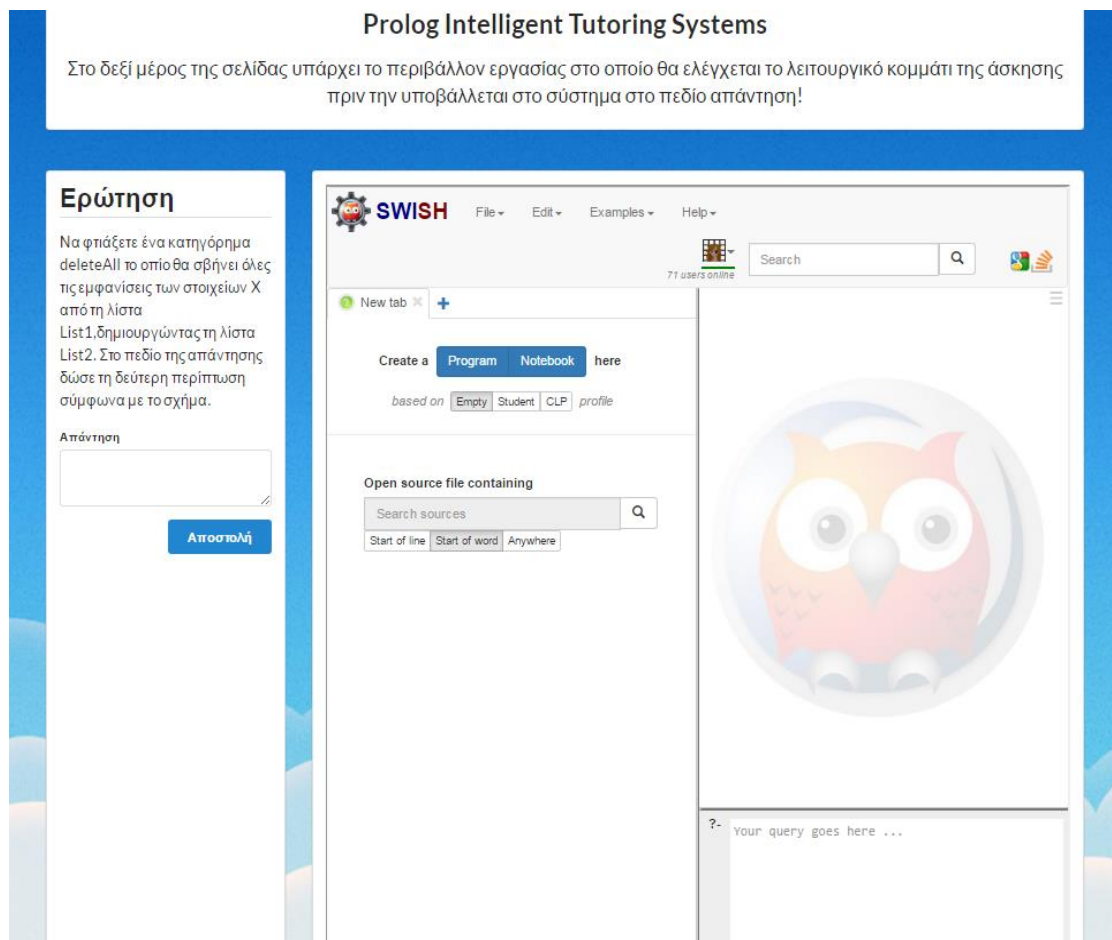


Figure 18: Page of selections practice without help: Prolog.

7.3 Sample Sessions of Assessment Section

The assessment section includes a test, which contains a set of exercises. If the student wants to proceed to the next lesson, he must do successfully these exercises. In fact, a student must give the correct answer at least in half of the test exercises in 45 minutes. If the available time runs out, then the system assumes that lesson as uncompleted and it automatically logs-out the user. If the student passes, the test that is his score is 50% or more than 50%, he can proceed to the next lesson. In any case, success or failure, the student knows his score. In case of failure, the system suggests the student to repeat that lesson and to do the test afterwards.

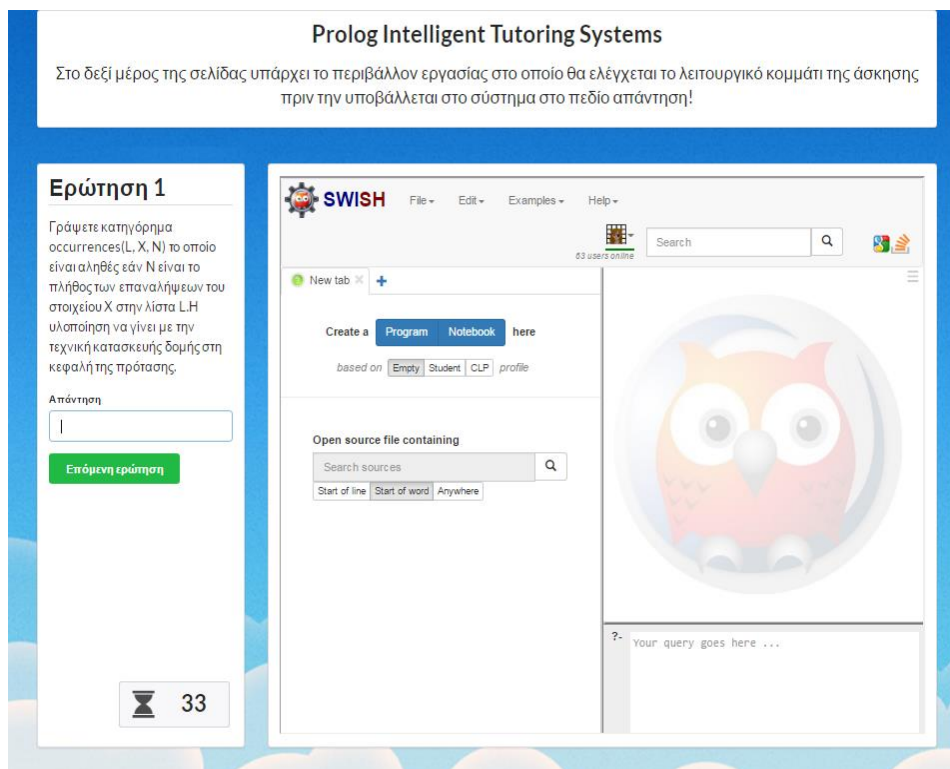


Figure 19: Page of selection: Assessment.

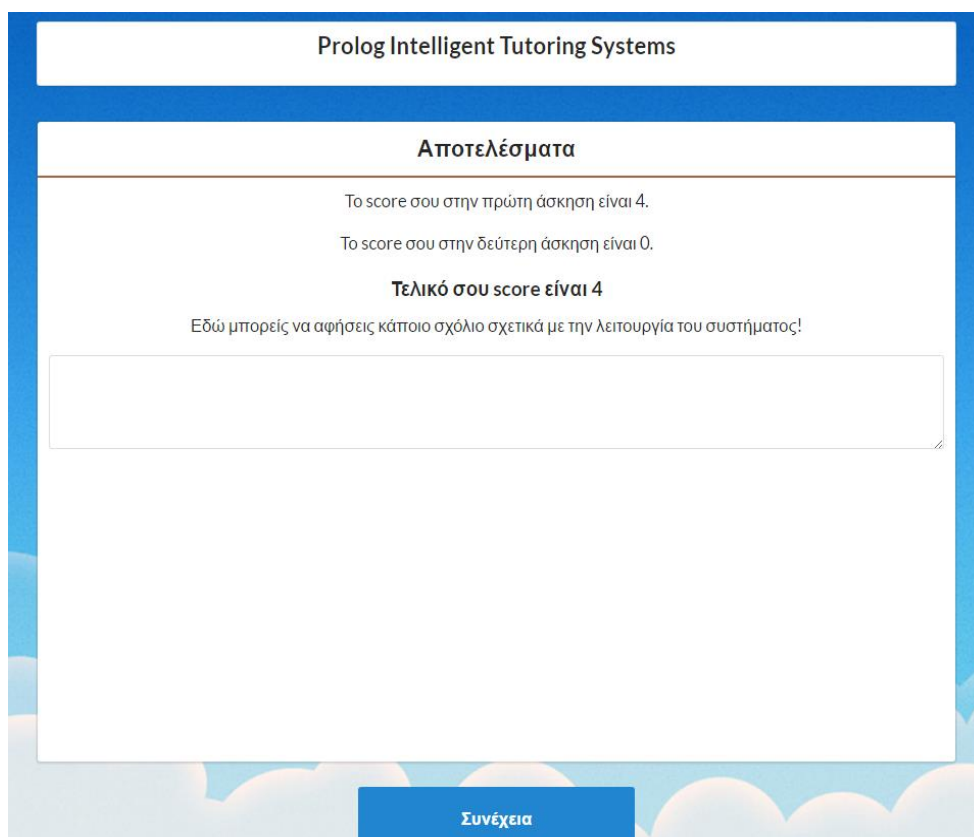


Figure 20: The demonstration of results.

At this point, the student can make some comments regarding this particular lesson. These comments are saved in to knowledge base. The corresponding teacher can read these comments and he/she may or may not take into consideration for this particular lesson.

7.4 Sample Sessions of Teacher Module

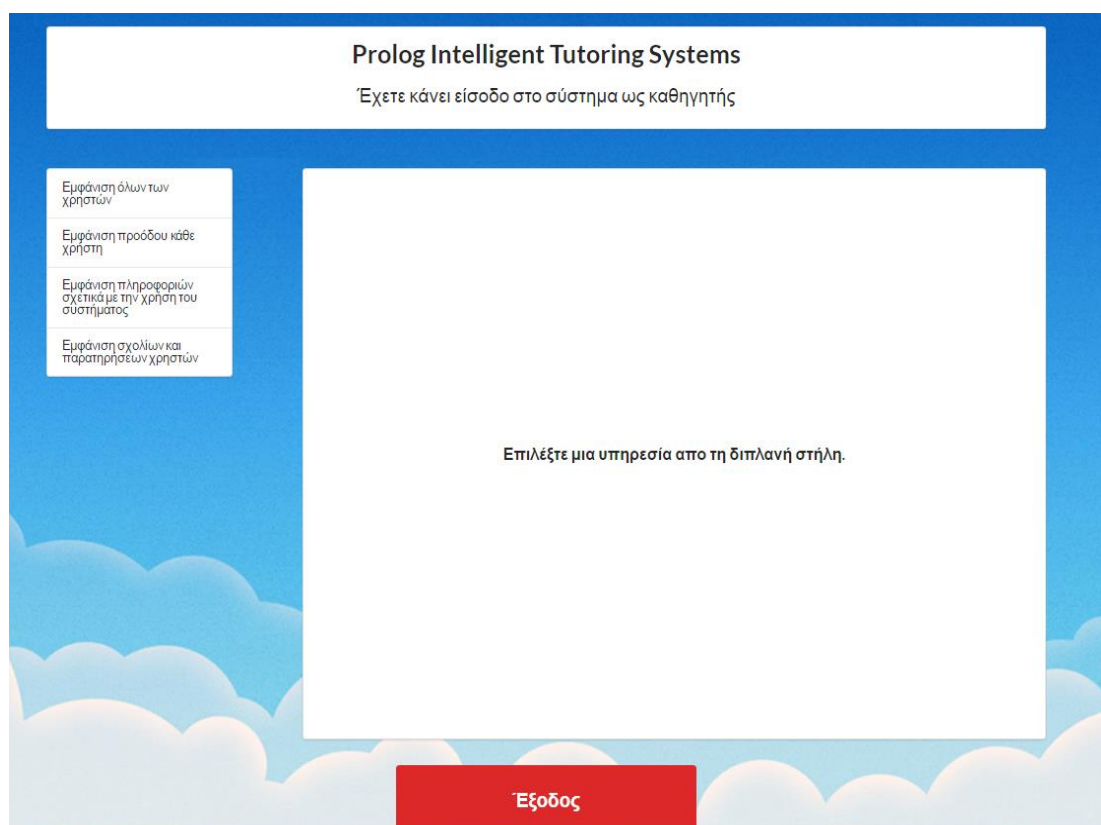


Figure 21: Page of teacher.

The potential users of our PITS are students and teachers. Each type of user has his own interface. Our PITS provides some services to teachers in the “Teacher Module”. These services are the following.

- 1) It can illustrate details of the users- students who used the system.
- 2) It can show for each particular student his/her performance and some details about all his learning activities. For a specific student our PITS illustrates, a) which lesson he studied and if he did the exercises for assimilation of the lesson, b) what

practice method(s) he followed and whether he did correctly the exercises and finally c) the grade of the student in the assessment test.

- 3) It illustrates some statistics, which are derived automatically by the system about the learner's students. For example, 70% of the learners used all functions of the system. That is, they did all sections (lecture, practice and assessment) and all available functions of each section.
- 4) It illustrates the comments, which are given by the learners.

8 Evaluation of PITS

This chapter evaluates and discusses the results of this study. A set of tables illustrate the percentages of answers, which were given by the learners during the evaluation procedure of PITS. In addition, it presents the evaluation procedure and the analysis of the questions of the survey questionnaire. Finally, it presents the analysis of the collected data.

8.1 Procedure of Survey

The first phase of the survey process was the acquisition of a database, from the secretary of Informatics Engineering Department, consisting of all students that fulfill the set requirement. The requirement was the attendance of the lab of the lesson of Logic Programming. Thirty students attended the course of Logic Programming. The subject of the lab is about learning programming in Prolog. The survey took place at the end of the academic semester so that the participants had already completed the formal course requirements. The second phase of the research process was the communication with these students in order to arrange an appointment with them. Many students were interested for participating in the survey. Approximately, half of the students achieved to make an appointment and to participate in the survey.

Our system was installed in a server in the Artificial Intelligent & Systems Engineering Lab and the students could use the system through web from the other PCs of the lab. Initially, we explained to each student the procedure of the survey. Each student had a unique password and username in order to log in to our system. The details concerning each participant in the survey are saved in the knowledge base of the system. All participants in the survey had to do as learners the same lesson. At the end, each one had to complete an on-line questionnaire.

8.1.1 Questionnaire Survey

In our questionnaire survey, we used Google Forms, which provide a fast way to create an online survey questionnaire. As soon as a questionnaire is created, a link is provided to this questionnaire that can be used directly to answer it from almost any

web browser, including mobile smart phone and tablet. It is very easy to view each response in a single row of spreadsheet, with each question show in a column.

The screenshot shows a web interface for a course completion screen. At the top, it says "Τέλος Μαθήματος" (End of Course). Below that, it says "Το score σου είναι 4." (Your score is 4). Then, it says "Δεν ολοκλήρωσες το μάθημα με επιτυχία. Προσπάθησε πάλι!" (You did not complete the course successfully. Try again!). In the center, there is a white box with a blue header that says "ΕΡΩΤΗΜΑΤΟΛΟΓΙΟ ΕΡΕΥΝΑΣ" (Questionnaire Survey). The text inside the box is in Greek and explains the purpose of the survey, which is to collect information about the course and the students' experience. It mentions that the survey is anonymous and that the results will be used to improve the course. At the bottom of the box, there is a red asterisk and the word "Απαιτείται" (Required). Below the box, there is a green button that says "Συνέχεια" (Continue).

Figure 22: *The end of course with questionnaire.*

We used the available questions as our primary tool in collecting the necessary information. By making the right choices on the type of survey questions, we were able to extract related data to the purpose of our survey.

The two most common types of survey questions are closed-ended questions and open-ended questions. The closed-ended ones limit the answers of the respondents to response options provided by questionnaire. In the open-ended questions, there are no predefined options. The participants should supply their own answers. Our questionnaire consists of both types of survey questions. It also includes ten questions, which will be analyzed in the following chapter.

8.1.2 Sample of Questionnaire Survey

The key terms in sampling are population and sampling frame. In our survey, the sampling refers to students of T.E.I who have learned Prolog Programming. Our

process and technique is known as simple random sampling, in which a simple random sample is an unbiased surveying technique. 15 students out of 30 registered students in the course of Logic Programming participated in the survey. The 15 students who participated in our sampling is a random sample of students of Informatics Engineering Department with knowledge about Prolog Programming. The distribution of our sample with respect to the sex of the students consists of 80% of male students and 20% female students. In addition, the question, “*Which is your current registration semester?*” aims to check the quality of the students with respect to the progress in their studies. Normally a student at the end of 8th semester should graduate. The majority of students (86,7%) have not yet graduated. That is, they study for more than 8th semesters. This is because the course of Logic Programming is an elective course and the students first try to finish the compulsory courses. 23,7% of the students are within the normal schedule of their studies.

8.2 Analysis of the Collected Data

8.2.1 Discussion and Analysis of Questionnaire

The questions of our questionnaire have been designed with the goal to evaluate the usability of our system. We have found this set of questions sufficient to satisfy our goal and they do not have redundancy and overlapping. The findings from the data analysis are presented below. In the following, we present 8 questions out of 10 questions of our questionnaires. That is, the ones which focus on the usability. Two of the questions are about the sex of students and their academic semester of studies. These two questions are about personal details. The goal of each question is discussed as well as. The analysis of the given answers is also presented in textual and in some cases in graphical form.

1) Do you understand at first glance of the first page (main interface) what this system is about?

Important aspect of each web-site is usability. A usability testing is the best way to understand how real users experience your website or application. One question of this questionnaire refers to the first page and if the first page is relevant with the

subject of the web site. The interface of web site or system must be relevant with the subject, which is presented (Nielsen & Mack, 1994b). 100% of the students participating in our survey considered that the main interface of the system is clear and understandable.

2) How easy is to find the theory of a lesson?

Our system is a web-based educational system, which aims to provide the theory through interactive learning techniques. Interactiveness sometimes poses problems to usability so the users cannot follow the reasoning of the system. As such, a question tries to identify, if the system gives enough support to students for easily finding a specific lesson theory. The question had three answers “not easy”, “easy” and “too easy”. The responses of students are either ‘easy” or “too easy” in order to find a specific lesson theory. The majority of the students (66.7%) have answered that is “too easy” and 33.3%, answered that is “easy”. These responses do not identify usability problems or problems in the educational procedure.

3) Do the examples that you find into the system help you understand the theory of the lesson?

Our system includes lessons and each lesson includes a set of theory lectures. In lecture module after displaying the theory, the system presents an example exemplifying the theory. The majority of students (93.3%) have answered that the example shown helps the students to understand the theory of the lesson. A small percentage (6,7%) supports that the example does not help to understand the theory. According to these responses, we do not identify usability problems or problems in the educational procedure.

4) Do the available practice methods satisfy your learning requirements?

Our system provides three methods of practice, one for each one of the following learner category: a) beginners b) intermediate and c) advanced Prolog programmers. To be able to answer this question the students were instructed to apply all practice methods. 86,7% of the students think that the available practice

methods meet their needs for learning Prolog programming and 13,3% think it does not. Therefore, we can conclude that the students have positive view about the available practice methods and about using new learning technologies in practice methods.

5) Evaluate the system in the scale 1 (I do not like) to 5 (like very much).

In each completed educational process, there is a grade to evaluate the students' effort and performance. At the same time, the teacher has to be evaluated for his teaching methods and the learning outcomes. As such, a question was devised for the overall evaluation of the system, i.e. the teacher, by the students. As such, the learners were asked to evaluate the system by giving a number from one to five. The scale 1-5 stands for "I don't like", "I like little", "I like", "I like a lot", and "I like very much". The results are shown in Figure 23. We observe that most of the learners are completely satisfied by the system. 20% of students have given three, i.e. a mid value evaluation. 33, 3%of the students have given five, the highest score. The majority of the students 46, 7% have given four, i.e. they like the system a lot. It is important to mention that none of the students has given two or one. In fact, our PITS is intended to be used by students whose learning requirements belong to a broad scale.

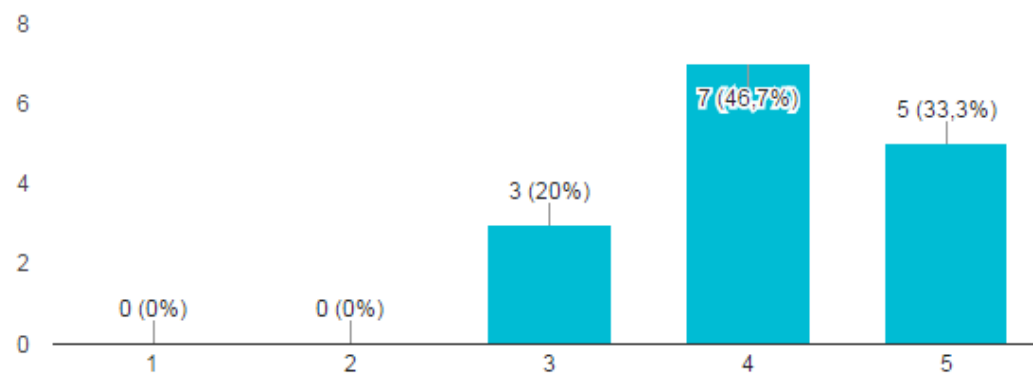


Figure 23 : Overall "like" scale of the system by the learners.

**6) Is the working environment pleasant?
(Give a number from 1 to 5)**

The students need a web environment, which it can satisfy their personal learning needs, and usability expectations in an efficient and effective way. It is very important to mention that the interface of each educational application should be based on user-friendly design criteria. Each educational application must provide usability through a pleasant working environment and as such, a question is devised to see the usability of our system. The scale 1-5 stands for “not pleasant”, “little pleasant”, “Pleasant”, “Pleasant a lot”, and “Very much pleasant”. 13,3% of students have found the usability “pleasant”, and 33,3% “pleasant a lot”. The majority of students, i.e. 53,3%, considered that the interface of PITS is “very much pleasant”.

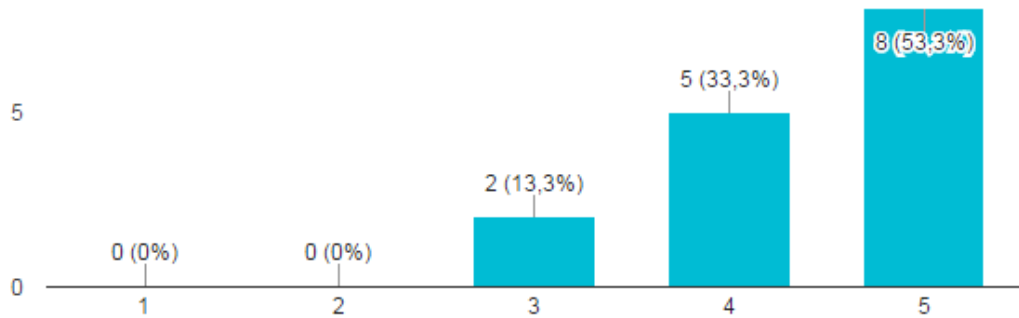


Figure 24: “pleasant” scale of the system by the learners.

7) Do you see any functionality problem in this system?

It is very important for a system to be complete in terms of functionality. As such, a question was created to discover if there are problems with the functionality of the system. If a student considers that there is a functional problem in our system, she/he is required to describe it in details. The majority of students, 80%, think that there is not any problem related with the functionality of our system. On the other hand, 20% of students think that there is some functionality problems.

7.1) Which is the problem? (Give an answer, if your response is YES in the above question.)

In case that student has given answer ‘yes’ in above question, he has to give details about the functionality problem. It is very useful for us to know the

suggestions of students for the functionality of the system. The positive responses of the students are the following: 1) Information for communication of the student with the lecturer is not available by the system. 2) Automatic detection of programming errors for any Prolog program. According to end-user suggestions, the proper updates were introduced to the system.

8) Have you successfully completed the lab of Logic Programming course?

The specific question is used in order to understand if the students of the survey have successfully completed the lab, i.e. programming in Prolog, of the Logic Programming course. The requirement set by this survey is the participants to have registered in the lab, i.e. “Prolog Programming”, of the Logic Programming course. Some students may have quitted in the middle of the course. 80% of students answered ‘yes’ and 20% study “Prolog Programming” incompletely. Only one student said that he did not successfully complete this course. This is mean that he selected this course with aim to study but he did not succeed.

8.2.2 Further Analysis of the Collected Data from Other Perspectives.

In this section we will further analyze selected data based on other parameters. There will be a cross effect analysis of the data. The following diagrams present the correlation among registration semester and the questions of usability and functionality of the system. We observe that the registration semester of the students affects the results. This is illustrated in Figures 29, 30 and 31. All students participating in the survey responded that they understood the type of the system they are using at first glance which can be correlated with the fact that the *80% of them have already completed the eighth of semester*. This is shown in *Figure 25*.

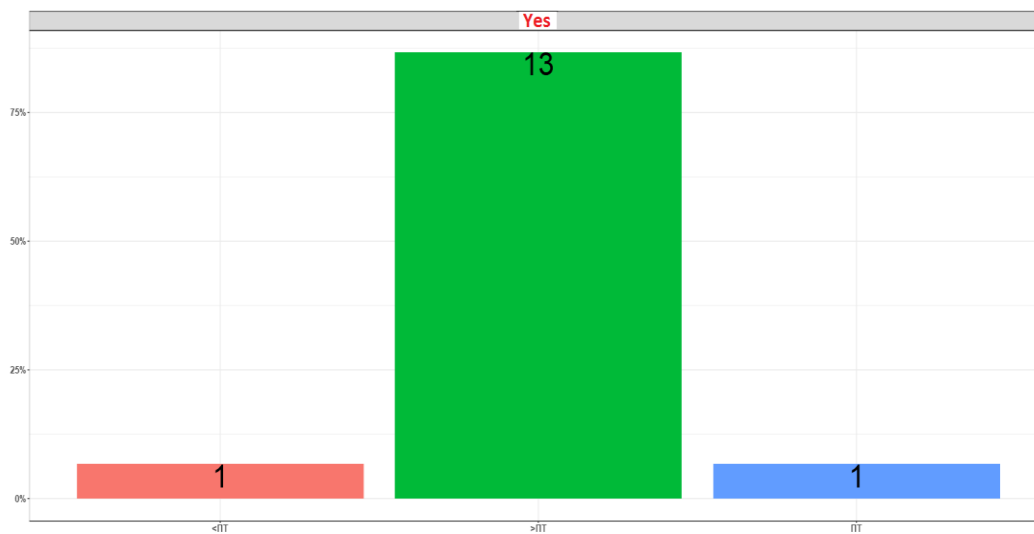


Figure 25: Diagram - “Do you understand at first glance of the first page (main interface) what this system is about?” VS “Which is your current registration semester?”.

In addition, 9 out of 15 of students who completed the eighth semester say that is ‘very easy’ to find the theory of lesson and 4 out of 15 say that is ‘easy’ respectively, again confirming that students in their late semesters find it easier to browse into the system.

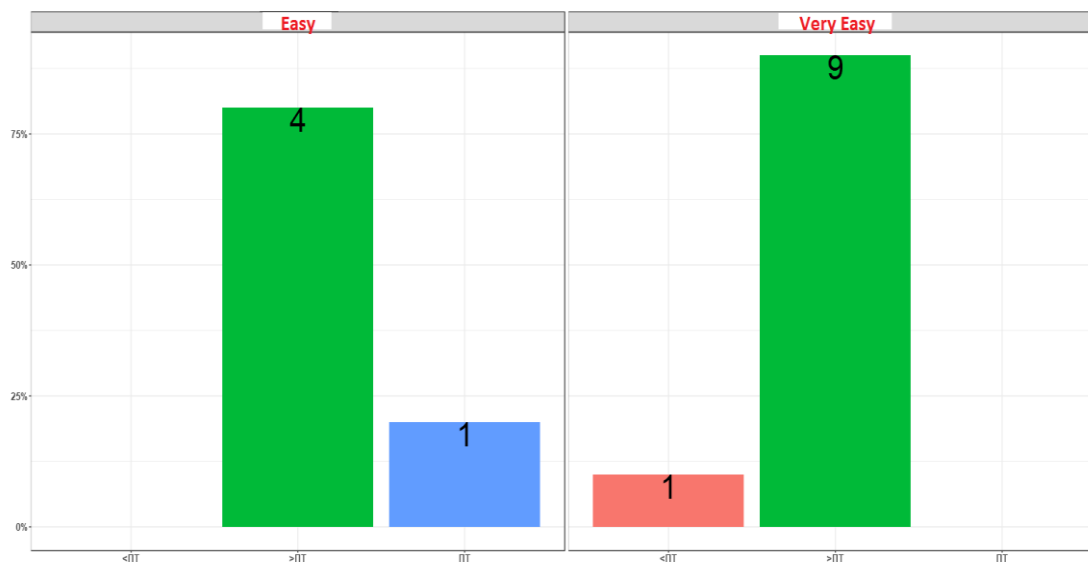


Figure 26: Diagram - “How easy is to find the theory of a lesson?” VS “Which is your current registration semester?”.

The system provides three methods of practice with aim to satisfy the need of different learners needs. 12 out of 15 students who have completed the eighth semester consider that the practice methods are provided by the system satisfy their needs. One responded

negatively. As such, we can conclude that students with experience in educational systems responded positively about the practice methods of our system.

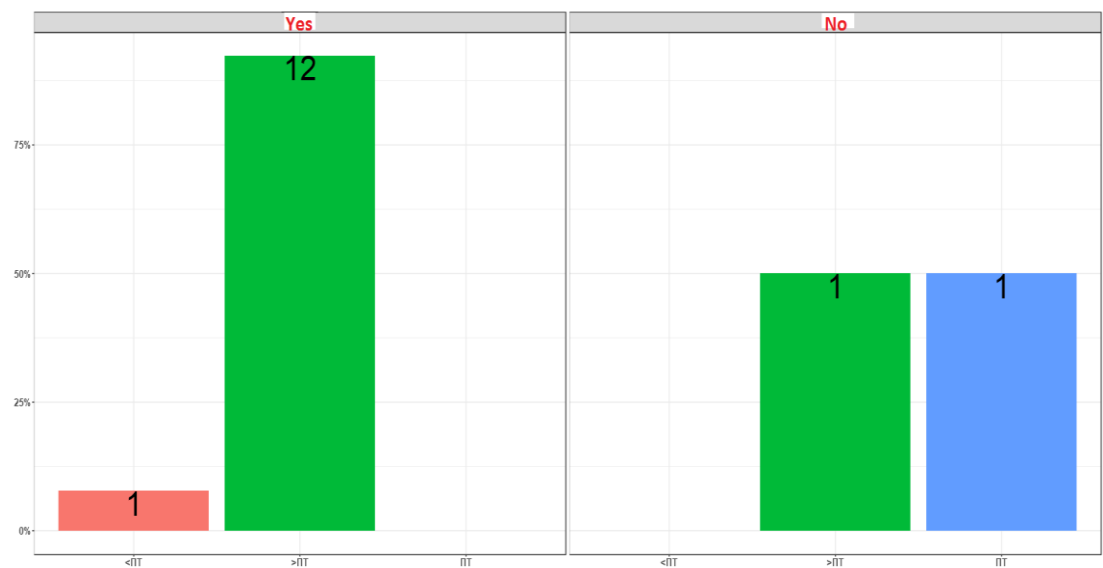


Figure 27: Diagram - “Do the available practice methods satisfy your learning requirements?” VS “Which is your current registration semester?”.

Usability is very important for a web based educational system as our PITS that it is why we check the usability. The usability engineers consider that a pleasant working environment is more attractive to a user (Nielsen & Mack, 1994b). 4 out of 15 of students evaluated our system as ‘very pleasant’. 4 out of 15 students evaluated our system with ‘very much pleasant’. 2 out of 15 students evaluated our system with ‘pleasant’.

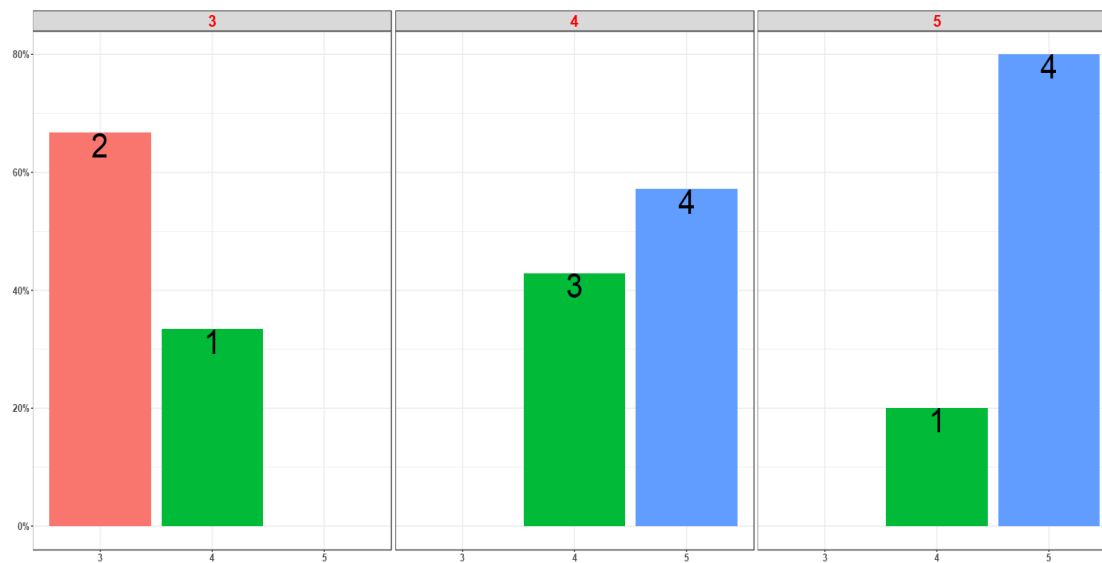


Figure 28: Diagram - “Is the working environment pleasant? (Give a number from 1 to 5)” VS “Evaluate the system in the scale 1 (I do not like) to 5 (like very much).”

Generally, the evaluation results are encouraging. The majority of students evaluated the system with ‘like a lot’ and ‘like very much’. According to the survey, there are not functionality problems. We see this positive evaluation from learners as very important for our system. Only two students said that there are functionality problems. These are the following: 1) Information for communication of the student with the lecturer, are not available by the system. This feature has been included in the current version of our system. 2) Automatic detection of programming errors for any Prolog program. This feature is not supported by our, because it is not within the scope of the current work.

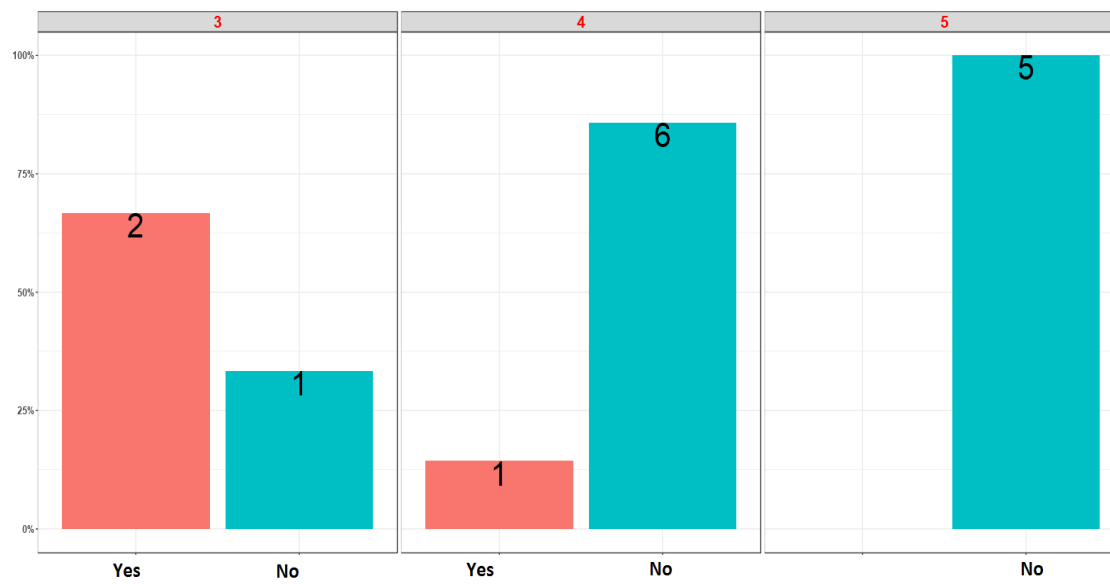


Figure 29: Diagram - “Do you see any functionality problem in this system?”
” VS “Evaluate the system in the scale 1 (I do not like)to5 (like very much).”

9 Conclusions and Future Work

9.1 Conclusions

In conclusion, our system is a Prolog Intelligent Tutoring system, which aims to teach Prolog programming. Our PITS follows the learning theory of constructivism. In addition, it shares some of the characteristics of Learning Management Systems. Moreover, the structure of its courses is based in the design of traditional learning. This is a benefit which is missing from the others ITS or educational systems. This system has the possibility to transform a traditional lecture into an active learning environment by using interactive teaching techniques. The practice section provides the ability to students to select the type of practice, which fit to their capabilities. This is an advantage enhancing the freedom of the users allowing them to further develop their knowledge (theory of constructivism).

We can observe that the students are eager to follow learning methods by using New Technologies. Our PITS is an e-learning system, which is used for student-centered teaching. It focuses on the learner and gives specific attention to provide knowledge by letting the students themselves to discover the learning style that best suits them. As such, the tool helps students to gain a positive attitude towards the learning process. The use of such a system can offer significant benefits for both the teacher and learners due to its ability to be used in many ways and to adapt according to individual student profiles. It can be applied to both traditional education and distance learning. In traditional education, it can be a tool enriching the teaching process, while it can be used remotely to replace completely the physical presence of the teacher. The main objective of this work is to enhance the interest of students and teachers and to become a precursor for new research that will evolve and improve its construction and its use.

In summary, the pros and cons of our system are the following.

Advantages

1. It can be used in real time learning in class and in distance learning.
2. It supports the teacher in his tasks by providing details about the learning activities and progress of each student.
3. The system provides full support to the student in the learning task. That is, it provides an integrated environment with all the tools that are needed in learning

Programming in Prolog. For example, a Prolog interpreter is integrated into our system.

4. A student can adopt the learning process to his particular needs. He can select the learning method which fits to his requirements. In this way, the student enforces his feelings of success about his learning progress. Since, the learning process is adapted to his requirements, it results in an easy and smooth learning task.
5. The development of our system takes into account the design of the lessons which is based on the theory of constructivism and in specific teaching techniques.
6. The theory of constructivism allows our system to be addressed to different types of learners, i.e. beginners, intermediate level and advanced learners.

Disadvantages

1. It sets learning limitations because the student directs his/her learning process.
2. It sets learning limitations because there is not communication with the other students in order to exchange ideas. That is, the social constructivism is not supported by the current form of our system.
3. It provides restricted support to the teacher.

Our system is built on the theory of constructivism. An evolution of it will be to have features of social constructivism. In the first development stage of the system we didn't like to include both constructivism and social constructivism. We selected to follow incremental development of our system. That is why we have left social constructivism to be supported in future version of our system.

9.2 Future Work

Tutoring by humans and computers is a powerful learning environment to the extent that implements sensible principles of learning. However, there are still a large number of unanswered fundamental questions that need attention in future research. Some aspects, which human tutors can do in the areas of dialogue and feedback, the intelligent tutoring systems, have limited performance. One reason for these limitations is that human tutors are able to interpret the affective state of the student.

One future challenge will be performance evaluation of human tutors versus machine tutors. Most people place their bets on human tutors under the assumption that they will be more sensitive to the student's profile and are more creatively adaptive in

guiding the student. However, the detailed analyses of human tutoring challenge such assumptions in light of the many illusions that humans have about communication and due to the modest pedagogical strategies in their repertoire. Computers may do a better job in cracking the illusions of communication, in inducing student knowledge states, and in implementing complex intelligent tutoring strategies. A plausible case could be made easily for betting on the computer over the human tutor.

The future use of PITS is on distance learning. For this reason each PITS should have some characteristics, which will ensure their use and will make PITS important teaching tools. So we suggest some features for ITS and specifically for PITS. A PITS should be characterized by flexibility. For example, it has to be easily adapted to different software environments and to various devices such as smartphone. Another future characteristic is the communication between groups of students and teachers. It would be better for increased learning results if students are divided into groups and they could communicate with each other and with their teachers. This feature is supported by the theory of social constructivism, which says that this type of communication enhances the learning process.

In addition, an appropriate environment is required which is aimed at teaching different ages and in different programming techniques. That environment has to be easy to use for both teachers and students. This means that a teacher can utilize all features, which are offered by PITS, and a student does not need specialized knowledge in order to use the available tools. The working environment should strengthen the interest for learning. This could be done by using learning objects with an interactive character, and with the development of virtual classes. Still something missing from the current PITS is the ability to communicate in natural language. It is very important for a student to communicate with a teacher in his own language and without having to change the working environment.

10 Bibliography

- Achi, I. I., & Agwu, C. O. (2015). Multi-Subject Area Based Intelligent Tutoring System. *International Journal of Scientific Engineering and Research (IJSER)*, pp. 2347-3878.
- Atkinson, K. (2002). Optimizing learning from examples using animated pedagogical agents. *Journal of Educational Psychology*, pp. 416-427.
- Beal, C., Beck, J., & Woolf, B. (1998). Impact of intelligent computer instruction on girls' math self concept and beliefs in the value of math. *American Educational Research Association*. American.
- Belikova, M., & Narvat, N. (1997). A Schema-Based Approach to Teaching Programming in Lisp and Prolog. *Proc. PEG' 97 int Conference P. Brna and D. Dicheva(Eds)*, pp. 22-29.
- Chalk, P. (1987). PROLOG-Based Computer-Aided Learning Environments. *Innovations in Education & Training International*, pp. 102-107.
- Cheung, B., Hui, L., Zhang, J., & Yiu, M. (2003). SmartTutor: An intelligent tutoring system in web-based adult education. *Journal of Systems and Software*, pp. 11-15.
- Du, H., & Wagner, C. (2006). Weblog success: exploring the role of technology. *International Journal of Human-Computer Studies*, pp. 789-798.
- Eliot, C., & Woolf, B. (1994). Reasoning about the user within a simulation-based real-time training system. *The fourth international conference on user modeling*, pp. 121-126.
- Ellis, R. (2009). Field Guide to Learning Management Systems, ASTD Learning Circuits. *Creative Education*.
- Fragaki, I. (2006, February). Iktinos: A System for Program Construction Based on Program Schemata. *Electrical Engineering Department*. TEI of Crete.
- Gegg-Harrison, T. (1993, April). Exploiting Program Schemata in a Prolog Tutoring System. *Department of Computer Science*. Duke University.
- Graesser, C., Conley, M., & Olney, A. (2010). Intelligent tutoring systems. *APA Handbook of Educational Psychology*, pp. 1-54.
- Graesser, C., Wiemer-Hastings, K., Wiemer-Hastings, P., & Kreuz, R. (1999). AutoTutor: A simulation of a human tutor. . *Journal of Cognitive Systems Research*, pp. 35-51.

- Heffernan, T., Turner, E., Lourenco, N., Macasek, A., Nuzzo-Jones, G., & Koedinger, R. (2006). The ASSISTment Builder: Towards an Analysis of Cost Effectiveness of ITS creation. *FLAIRS2006*. Florida.
- Hong, J. (2004). Guided programming and automated error analysis in an intelligent Prolog tutor. *International Journal of Human Computer Studies*, pp. 505-534.
- Keles, A., Ocak, R., & Gulcu, A. (2009). ZOSMAT: Web-based Intelligent Tutoring System for Teaching-Learning Process. *Expert Systems with Applications: An International Journal*, pp. 1229-1239.
- Lajoie, P., & Lesgold, A. (1989). Apprenticeship training in the workplace: Computer coached practice environment as a new form of apprenticeship. *Machine-Mediated Learning*, pp. 7-28.
- Looi, K. (1991). Automatic debugging of Prolog programs in a Prolog Intelligent Tutoring System. *Instructional Science*, pp. 215-263.
- Madden, G. (2014). APA Handbook of Behavior Analysis. Retrieved December 24, 2014, 51-58.
- Mandler, G. (2002). Origins of the cognitive (r)evolution. *Journal of the History of the Behavioral Sciences*, pp. 339-353.
- Marakakis, E. (1997). Logic Program Development Based on Typed, Moded Schemata and Data Types. *PhD thesis*. UK/Bristol: University of Bristol.
- Marakakis, E. (2014). Prolog: Programming in logic in Artificial Intelligence. *New Technologies Publications*, pp. 183-184.
- McKinley, J. (2015). Critical argument and writer identity: social constructivism as a theoretical framework for EFL academic writing. *Critical Inquiry in Language Studies*, pp. 184-207.
- Melis, E., & Siekmann, J. (2004). Activemath: An Intelligent Tutoring System for Mathematics. *7th International Conference "Artificial Intelligence and Soft Computing"*, pp. 91-101.
- Meyer, L. (2009). The Poverty of Constructivism. *Educational Philosophy and Theory*, pp. 332-341.
- Moon-c, L. (1990). An Augmented Prolog Programming for Tutoring Applications Environment. *Dept of Computer Science*, 898-906. Hong Kong /Shatin: Chinese University of Hong Kong .
- Nguyen, L., Menzel, T., & Wolfgang, G. (2009). Using Weighted Constraints to Diagnose Errors in Logic Programming - The Case of an Ill-defined Domain. *International Journal of Artificial Intelligence in Education*, pp. 381-400.

- Nielsen, J., & Mack, L. (1994). Heuristic evaluation. In Nielsen, J., and Mack, R.L. . *Usability Inspection Methods*. New York/: John Wiley & Sons.
- Orfanos, G. (2008). *Anthology of terms and theories in general didactic and phycho-paidagology of themes*. Athens : Gutenberg.
- Petrina, S. (2007). Advance Teaching Methods for the Technology Classroom . *Information Science Publishing.*, pp. 93-99.
- Robertson, D. (2005). Quick Prolog. *School of Informatics*. University of Edinburgh.
- Robins, A., Rountree, T., & Rountree, N. (2003). Learning and teaching programming:A review and discussion. *Computer Science Education* , pp. 63-75.
- Scerri, R. (2003). Philosophical Confusion in Chemical Education. *Journal of Chemical Education*, pp. 468-474.
- Schiaffino, S., Garcia, P., & Amandi, A. (2008). eTeacher: Providing personalized assistance to e-learning students. *Computers & Education 51*, pp. 1744-1754.
- Stathaki, A., Kondylakis, H., Marakakis, E., & Kalogerakis, M. (2017, October 30-31). i-Prolog: A Web-based Intelligent Tutoring System for Learning Prolog. *2nd EAI International Conference on Learning & Innovation*.
- Stottler, D., & Domeshek, E. (2005). Intelligent Tutoring Systems (ITSs): Advanced Learning Technology for Enhancing Warfighter Performance. *Education Conference (I/ITSEC)*, pp. 1-7.
- Vlahavas, I., Refanidis, I., & Sakellariou, H. (2005). Multimedia System for Teaching the Language of Logic Programming Prolog. *Univercity of Macedonia*.
- Wang, N., Johnson, L., Mayer, E., Rizzo, P., Shaw, E., & Collins, H. (2008). The politeness effect: Pedagogical agents and learning outcomes. *International Journal of Human-Computer Studies*, pp. 98-112.
- Webb, G. (1991). Inside the unification tutor : The architecture of an Intelligent Educational System. *International Journal of Artificial Intelligence in Education* , pp. 1-7.
- Wedd, G. (1989). The Unification Tutor - An Intelligent Educational System in the Classroom. *International Journal of Artificial Intelligence in Education*, pp. 1-12.
- Westwood, P. (2008). *What teachers need to know about teaching methods*. Camberwell: ACER Press.
- Yang, M. (2007). Approaches for Learning Prolog Programming. . *Innovation in Teaching and Learning in Information and Computer Sciences*, pp. 88-107.

11 Appendix A: Code of Modules

11.1 A.1. Algorithm of Lecture, Practice and Assessment Modules

```
if log_in=yes /\ stoiceia= ok then
{ Print 'You are in lesson', N;
  case
  selection= 'lecture'
  {i:=0;
  while i<2 do
  { Print Theory_of_Lecture[i];
  if Theory_of_Lecture[i]= studied then
  {Print Question_of_Lecture[i];
  Read Answer;
  if Answer=Answer_of_Lecture[i] then
  { Print 'This is the corect answer';

  else
  Print 'The answer is not correct';
  Print Correct_Answer_of_Lecture[i];
  }
  }
  i:=i+1;
  }
  Print 'End of Lecture';
}

selection= 'practice'
{ Print Question_of_Practice;

  case
  selection = 'practice_with_help'
  { Print Question_of_Practice;
  i:=0;
  while i>3 do
  { Prin Text_of_Help[i];
  i:=i+1;
  }
  Read Answer;
  if Answer = Answer_of_Practise
  { Print 'This is the corect answer';
  else
  Print 'The answer is not correct';
  Print Correct_Answer_of_Practise;
  }
  Print 'End of Practise';
}

selection = 'practice_with_schema'
{Print Question_of_Practice;
  Print 'You can select one of a schema';
  case
  Schema=Schema1
  { Print Schema1;
```

```

        Read Answer;
        if Answer=Answer_of_Practise
    { Print 'This is the corect answer';
else
    Print 'The answer is not correct';
    Print Correct_Answer_of_Practise;
    }

        Print 'End of Practise';
    }
    Schema=Schema2
    { Print Schema1;
    Read Answer;
        if Answer=Answer_of_Practise
    { Print 'This is the corect answer';
else
    Print 'The answer is not correct';
    Print Correct_Answer_of_Practise;
    }

        Print 'End of Practise';
    }

    }

    selection = 'practice_without_help'
    {Print Question_of_Practice;
    Read Answer;
        if Answer = Answer_of_Practise
    { Print 'This is the corect answer';
else
    Print 'The answer is not correct';
    Print Correct_Answer_of_Practise;
    }

        Print 'End of Practise';
    }

    }

    selection = 'assesment'
    {
        i:=0;
        score:=0;
    while i<2 do
    { Print Question_of_Assesment[i];
    Read Answer[i];
        if Answer[i]=Answer_of_Assesment[i] then
    { Score:=Score+Grage_of_Answer_of_Assesment[i];
        else
        Score:=Score +0;
    }

    i:=i+1;
    }

        Print 'The score of test is', Score;
        if Score >= 5 then
    {
    Print 'You can proceed into the next lesson';
    Lesson_N := Lesson_N+1;
    }
    }
else

```

```

        Print 'You can not proceed into the next lesson';
        Lesson_N := Lesson_N+0;
    }
}

```

Algorithm 1: Algorithm of Lecture, Practice and Assessment Modules.

11.2 A.2. Algorithm of Teacher Module

```

if log_in=yes /\ elements_of_Students= ok then
    { Print 'Our system supports some operations of the teacher
      which aim to follow the student's learning progress. You can
      select one of these.';

    case
    selection = 'show_users'
    {Print'You can see the students who used the system
      are',Number_of_students;

    i:=0;
    while i< Number_of_students
    {
    Print Elements_of_Students[i];
    i:= i+1;
    }

    Selection = 'capture_info'
    {Print 'You can see details about the learning progress for
      each student-user.';

    Print Helpdesc_Id_N;
    i:=0;
    while i< Number_of_students
    {
    Print Elements_of_Students[i];
    i:= i+1;
    }

    Print 'Give the id of student for seeing the
      learning progress.';

    Read Students_Id;
    Print Elements_of_Students[Students_Id];

    }
    Selection = 'info_of_PITS'
    {Print 'You can see some statistics which are derived
      automatically by the system about the use of the system
      from its students-users and about the learning activities
      of each individual student-user. ';

    i:=0;
    while i< Number_of_students
    { if details_of_progress[i]=pra then
      practice:=practice+1;
      if details_of_progress[i]=lct then
        lecture:=lecture+1;

      i:= i+1;
    }

    Print 'The system was used by', Number_of_students,

```

```

        'students.', 'From these',Number_of_students, 'students',
        practice,'have practiced in in Prolog programming,
        in addition', lecture, 'have visited the section of
        lexture';
    }
    Selection = 'helpdesc'
    {Print 'You can read the comments of students-users about
        the system and the lessons.';
        Print Helpdesc_Id_N;
        i:=0;
        while i< Helpdesc_Id_N
        { Print Helpdesc_Text_of_comments[i];
        i:= i+1;
        }
    }
}

```

Algorithm 2: Algorithm of TeacherModules.

12 Appendix B

12.1 B.1. On-line questionnaire of Survey

ΕΡΩΤΗΜΑΤΟΛΟΓΙΟ ΕΡΕΥΝΑΣ

Αγαπητέ Κύριε, αγαπητή Κυρία:

Το ερωτηματολόγιο αυτό αφορά μια έρευνα μεταπτυχιακών σπουδών και έχει σκοπό να συγκεντρώσει πληροφορίες σχετικά με το πόσο αποδοτικά και λειτουργικά είναι σχεδιασμένο ένα σύστημα εκμάθησης για τον χρήστη που την χρησιμοποιεί. Σε αυτό το ερωτηματολόγιο δεν υπάρχουν σωστές ή λανθασμένες απαντήσεις. Αυτό που χρειάζεται είναι να εκφράσεις τη γνώμη σου κάνοντας κάποιες ενέργειες και απαντώντας με ειλικρίνεια σε όλες τις ερωτήσεις. Να διαβάσεις με προσοχή ολοκληρή την ερώτηση πριν απαντήσεις. Τέλος θα ήθελα να σου υπενθυμίσω ότι το ερωτηματολόγιο είναι ανώνυμο, δηλαδή δεν χρειάζεται να γράφεις πουθενά το όνομά σου. Ελπίζω πως θα σε κούρασω και σε ευχαριστώ εκ των προτέρων για τη βοήθειά σου.

Τ.Ε.Ι ΚΡΗΤΗΣ
Μ.Π.Σ. ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΠΟΛΥΜΕΣΩΝ
Σταθάκη Αιροδίτη

* Απαιτείται

1.Με μια πρώτη ματιά καταλαβαίνεις τι παρουσιάζει αυτό το σύστημα? *

☐ ΝΑΙ

☐ ΟΧΙ

2.Πόσο εύκολο είναι να βρεις τη θεωρία του μαθήματος? *

☐ ΠΟΛΥ ΕΥΚΟΛΟ

☐ ΕΥΚΟΛΟ

☐ ΚΑΘΟΛΟΥ ΕΥΚΟΛΟ

3.Τα παραδείγματα που βρήκες στο σύστημα σε βοήθησαν στην κατανόηση της θεωρίας? *

☐ ΝΑΙ

☐ ΟΧΙ

4.Σε ικανοποιούν οι μέθοδοι εξάσκησης που είναι διαθέσιμοι στο σύστημα? *

☐ ΝΑΙ

☐ ΟΧΙ

5.Από το 1 έως το 5 δώσε ένα βαθμό για το σύστημα? *

Δεν μου αρέσει

1

2

3

4

5

Μου αρέσει

6.Το περιβάλλον είναι ευχάριστο? *

1

2

3

4

5

ΚΑΘΟΛΟΥ ΕΥΧΑΡΙΣΤΟ

☐

☐

☐

☐

☐

ΠΑΡΑ ΠΟΛΥ ΕΥΧΑΡΙΣΤΟ

7.Θεωρείς ότι υπάρχει κάποιο πρόβλημα στο σύστημα? *

☐ ΝΑΙ

☐ ΟΧΙ

7.1 Ποιο είναι κατά τη γνώμη σου το πρόβλημα?(Δώσε μια απάντηση αν απάντησες στην παραπάνω ερώτηση ΝΑΙ)

Η απάντησή σας

8.Έχετε διδαχθεί την ύλη του μαθήματος του Λογικού Προγραμματισμού?

☐ ΝΑΙ

☐ ΟΧΙ

☐ ΜΗ ΟΛΟΚΛΗΡΩΜΕΝΗ ΠΑΡΑΚΟΛΟΥΘΗΣΗ

9.Φύλο

☐ ΑΝΔΡΑΣ

☐ ΓΥΝΑΙΚΑ

10.Τρέχον εξάμηνο(ΠΤ= 8ο Εξάμηνο)

☐ >ΠΤ

☐ ΠΤ

☐ <ΠΤ

ΥΠΟΒΟΛΗ

Μην υποβάλετε ποτέ κωδικούς πρόσβασης μέσω των Φορμών Google.

Figure 30: On-line questionnaire of Survey.