Available online at www.sciencedirect.com

ScienceDirect

Computers & Education 51 (2008) 1017–1036

COMPUTERS & EDUCATION

www.elsevier.com/locate/compedu

TEx-Sys model for building intelligent tutoring systems

Slavomir Stankov *, Marko Rosić, Branko Žitko, Ani Grubišić

Faculty of Natural Science, Mathematics and Kinesiology, University of Split, Teslina 12, 21000 Split, Croatia

Received 26 February 2007; received in revised form 3 October 2007; accepted 7 October 2007

Abstract

Special classes of asynchronous e-learning systems are the intelligent tutoring systems which represent an advanced learning and teaching environment adaptable to individual student's characteristics. Authoring shells have an environment that enables development of the intelligent tutoring systems. In this paper we present, in entirety, for the first time, our approach to research, development and implementation related to intelligent tutoring systems and ITS authoring shells. Our research relies on the traditional intelligent tutoring system, the consideration that teaching is control of learning and principles of good human tutoring in order to develop the Tutor–Expert System model for building intelligent tutoring systems in freely chosen domain knowledge. In this way we can wrap up an ongoing process that has lasted for the previous fifteen years. Prototype tests with the implemented systems have been carried out with students from a primary education to an academic level. Results of those tests are advantageous, according to surveys, and the implemented and deployed software satisfies functionalities and actors' demands.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Architectures for educational technology system; Authoring tools and methods; Distributed learning environments; Evaluation methodologies; Intelligent tutoring systems

1. Introduction

E-learning is a revolutionary educational paradigm based on the information and communication technology that is being dynamically researched, developed and applied as a part of the traditional instruction and as a complementary mechanism for lifelong and distance learning. E-learning is enabled by the e-learning systems that can be observed as synchronous or asynchronous, regarding their implementation technology for delivering educational contents. A special class of asynchronous e-learning systems is the intelligent tutoring systems (ITSs) which represent an advanced learning and teaching environment adaptable to individual student's characteristics.

Intelligent tutoring systems are generation of computer systems aimed to support and improve learning and teaching process in certain domain knowledge, considering individuality of a student like in traditional

* Corresponding author. Tel.: +385 21 385 133.
E-mail address: slavomir.stankov@pmfst.hr (S. Stankov).

0360-1315/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2007.10.002

one-to-one instructional process. This process, also known as human tutoring, has been confirmed to be successful and presents the most efficient learning and teaching process (i.e. Bloom, 1984; Cohen, Kulik, & Kulik, 1982). The development of ITSs is, therefore, related to a number of serious problems, because proper implementation of "human" tutor can be done only in relation to cognitive psychology, artificial intelligence and education. Knowledge is a key to intelligent behavior and, therefore, ITSs are said to be knowledge-based because they have: (a) domain knowledge; (b) knowledge about teaching principles and about methods for applying those principles and (c) knowledge about methods and techniques for student modeling.

In this paper we present, in entirety, for the first time, our approach to research, development and implementation related to intelligent tutoring systems and ITS authoring shells. Our research relies on the traditional intelligent tutoring system (presented in Section 1.1), the consideration that "teaching is control of learning" (Pask, 1965) and principles of good human tutoring (Merrill, Reiser, Ranney, & Trafton, 1992) in order to develop the Tutor–Expert System (TEx-Sys) model (Stankov, 1997) for building intelligent tutoring systems in freely chosen domain knowledge (presented in Section 1.2). In this way we can wrap up an ongoing process that has lasted for the previous 15 years.

1.1. From intelligent tutoring systems to authoring shells

Intelligent tutoring systems present outgrowth of computer aided instruction (CAI) systems that were created in the late fifties of the last century based on the programmed instruction (Skinner, 1954). When talking about conjunction of computers and programmed teaching, Skinner said that micro computer is ideal hardware for programmed instruction (Skinner, 1986). The most productive and the most long-lasting example of programmed instruction is the PLATO system (www.plato.com) for teaching learners of every age and occupation. In the early 1970s, an artificial intelligence has just started to be applied into the CAI systems. Carbonell in Scholar (Carbonell, 1970) defines other type of CAI that is today known as knowledge-based or intelligent CAI (ICAI). The Scholar developers have done a pioneering effort while creating a computer tutor that is able to handle unexpected student question and is able to generate instructional material with changeable levels of detail, depending on the context of the dialog (Barr & Feigenbaum, 1986).

Sleeman and Brown edited in 1979 a special issue of the IEEE International Journal of Man–Machine Studies (Sleeman & Brown, 1979) in order to summarize and point up research in this area. Later on, they have published some of those papers in a book titled "Intelligent Tutoring Systems" where they reviewed the state of the art in computer aided instruction and first created the term intelligent tutoring systems to portray these systems and to distinguish them from the previous CAI and ICAI systems (Sleeman & Brown, 1982).

An interesting approach to this field of research is presented by Ohlsson (1986) in his analysis of some principles of intelligent tutoring where he uses terms "teaching", "tutoring" and "instruction" as synonyms. This, of course, is an issue for another very interesting discussion about consequences of very different terminology related to ICAI and ITS. For example, Hartley and Sleeman indicate the problem of mechanizing the teaching process and introduce and develop a term "teaching intelligence" (Hartley & Sleeman, 1973). Besides, they believe that a intelligent decision-making system for computer-based teaching needs four educational components: a representation of the teaching task, a representation of the student, a set of teaching operations instruction, and, finally, a set mean-ends guidance rules. Anderson and others (Anderson, Boyle, & Reiser, 1985) believes that computer systems for intelligent tutoring must provide the student with the same instructional advantages as a sophisticated human tutor. Moreover, they emphasis that this area is interdisciplinary as it has arisen on the intersection of cognitive psychology, artificial intelligence and computer technology. Wenger in his book continues to review the ITSs state of the art (Wenger, 1987). He highlights that many other scientific fields, such as artificial intelligence, education linguistics, psychology, and philosophy, contributed from research on ITS. Rickel (1989) presents excellent analysis of researches related to ICAI systems based on following system component: learning scenario, domain knowledge representation, student modeling, pedagogical knowledge, user interface.

Self writes about existence of a consensus on the standard ITS architecture: it consists of components which know about the subject matter, know about the student, and know about tutoring (Self, 1990), better known as "the what, the who and the how" components (Self, 1974). Nevertheless, a fourth component, called differently by different authors, has to be added to the previous list: an instructional environment and human–computer interface channel tutorial communication (Burns & Capps, 1988) or an interface as a form of communication (Wenger, 1987) or communication module (Woolf, 1992).

Shute and Psotka (1995) gave an extensive survey of the field (period from 1960s to mid 1990s), where they analyze the meaning of intelligent system. Intelligent system (we assume the authors mean intelligent tutoring system) must be able to: (a) accurately diagnose students' knowledge structure and (b) adapt instruction accordingly. Besides that, in this paper authors analyzed the opinions of twenty ITS experts who were asked what the „I" in the ITS meant. That analysis enabled them to determine some critical elements in ITS design and development: a real-time cognitive diagnosis (or student modeling) and adaptive remediation, what is closely related to „T" in the ITS.

Recently, the ITS research is oriented mainly on their Web environment, because it is currently a hot research and development area, opening new ways of learning and teaching for many people. However, the most of those systems have very limited capabilities because they are based on static representation of educational contents. Interactive and adaptable functions like curriculum sequencing, interactive problem solving support, and intelligent analysis of student solutions (Brusilovsky, Schwarz, & Weber, 1996), can increase the potential of those systems. These functions can be implemented using some active Web server technologies like dynamically generated Web contents that depend on students questions. Alpert and others (Alpert, Singley, & Fairweather, 1999) believe that a Web-enabled intelligent tutoring system with client/server architecture can have one of the possible approaches: (a) all tutorial behavior resides on the client; (b) all tutorial functionality resides on the server side and the user interacts with through a standard Web browser; and (c) distributed client–server architecture where some of the tutorial behavior resides in the client and some in the server.

While Web oriented intelligent tutoring systems are becoming more used in educational community and proved to be increasingly effective, they are difficult and expensive to build, have no interoperability and subject matter sharing. Authoring systems and authoring shells with their authoring tools, as well as standards in learning technology (like, IEEE Learning Technology Standards Committee - *ieeeltsc.org*; Advanced Distributed Learning – www.adlnet.gov), have been developed to resolve mentioned problems. Murray (Murray, 1999) analyses the research and development state of the art of authoring systems for building intelligent tutoring system considering their seven categories. Moreover, for an inclusive overview of the current authoring systems and authoring tools, the interested reader is directed to the monograph written by Murray, Blessing, and Ainsworth (2003).

We have analyzed some intelligent tutoring systems that are frequently cited and we have used them in comparison with our own approach to this propulsive area. ELM-ART is an intelligent textbook with an integrated problem solving environment to support learning programming in Lisp (Brusilovsky et al., 1996). CALAT provides an individual adaptation capability using explanation, exercise and simulation pages in its courseware (Nakabayashi, Maruyama, Kato, Touhei, & Fukuhara, 1997). DCG generates individual courses according to the students' goals and foreknowledge, and dynamically adapts the course according to the students' achievements in acquiring knowledge (Vassileva, 1997). MANIC is a system that uses synchronized audio and HTML slides, as well as interactive quizzes (Stern, 1997). AlgeBrain is an equation tutor with a rule-based expert system that simulates an expert equation solver and can decompose the operators and operands in every problem-solving step (Alpert et al., 1999). The following systems have recently preoccupied both researchers and teachers: REEDEM is an ITS authoring environment which allows teachers to turn existing computer-based teaching material into an ITS that provides distinguished and adaptive instruction (Ainsworth et al., 2003); SQLT is a Web based ITS for student learning and teaching the SQL database language. (Mitrovic, 2003); AutoTutor is a Web based environment that tutors students through a dialogue in natural language, inspired by explanation-based constructivist theories of learning, adaptive reply to student knowledge and empirical research on dialogue patterns in tutorial conversation (Graesser et al., 2004).

1.2. Overview of our previous work – our approach to ITS

Appreciating stipulated principles for building intelligent tutoring systems and authoring shells, and using the idea of the cybernetic model of the system (Božičević, 1980), we have developed the TEx-Sys model.

The first implementation of this model is the on-site TEx-Sys (in the period from 1992 to 2001). It has enabled realization of learning and teaching process for two main actors: a student and a teacher. These actors have two basic functionalities: (i) designing learning contents related to any domain knowledge and (ii) learning and teaching, as well as, testing and evaluating knowledge. The second implementation phase resulted with the system based on the dynamic Web documents, Distributed Tutor–Expert System (DTEx-Sys), in the period from 1999 to 2003 (Rosić, 2000). This Web oriented ITS has been developed by keeping in mind issues like (i) accessibility for as large a number of potential users and (ii) learning and teaching in arbitrary domains. DTEx-Sys builds upon the experiences of an earlier on-site TEx-Sys version with only one difference: it does not have environment for domain knowledge bases design, yet it uses the ones developed in on-site TEx-Sys. Finally, the last version is the system based on the Web services, eXtended Tutor–Expert System the xTEx-Sys (in the period from 2003 to 2005) (Stankov, 2005). In this version we have strictly differ the functionalities of experts for domain knowledge design and teachers for learning contents design or courseware design.

We have been recently using all three versions of implemented systems as a support to different courses. Prototype tests with the implemented systems have been carried out with students from a primary education to an academic level. Results of those tests are advantageous, according to surveys, and the implemented and deployed software satisfies functionalities and actors' demands. The student's environment has been tested to determine the system's usability and students' achievements in the learning and teaching process. In the period from 2001 to 2007, there were in total 5482 knowledge tests solved by 1302 students while evaluating their understanding of different domain knowledge, from University of Split, University of Zagreb and some primary schools in Split. Besides that, the TEx-Sys model has been used for preparation of 30 graduate theses written by students from the previously mentioned universities, six master's theses and three PhD theses.

Analyzing systems mentioned in previous section and comparing their functionalities with TEx-Sys model functionalities, we came across to some similarities, but as well, we discovered some particular features and we point out the most important ones:

- We rigorously divide domain knowledge design from courseware design. Expert has domain area knowledge and skills which presents grounds for teachers who design courseware. This approach enables us to introduce ontological environment as a specification of concepts (like Gruber, 1993) for experts, teachers and students.
- We observe the authoring shells as systems that have an ontological environment and enable experts to design domain knowledge, teachers to design educational contents and finally students to learn and test.
- Our view of authoring shells makes possible realization of requirements for interoperability, reusability, durability and accessibility of knowledge and learning objects embedded into educational contents (like SCORM – Sharable Content Object Referent Model, www.adlnet.gov).
- Dynamical quiz in TEx-Sys model enables adaptability by the evaluation of every step in student testing where the system generates new questions depending on a student's partial results.

We believe that second chapter will give the readers detail inside of originality of our approach in intelligent tutoring systems design as well as description of all three versions of the TEx-Sys model.

The paper is organized as follows: the second chapter presents the idea and the structure of the TEx-Sys model, actors' and functionalities' specification, as well as the architecture of the derived systems; the third chapter describes application of the derived systems and presents results of a certain number of experiments about the effectiveness of the TEx-Sys model.

2. Tutor–Expert System model

Our research has started with studying the cybernetic model of the system that identifies the process, the reference value and the control. Modifying that cybernetic model according to the didactic principles of an individual instruction, the idea of expert systems, as well as the methods and techniques of knowledge presentation, enabled us to develop a model of an authoring shell for implementing the ITSs called the TEx-Sys. We have modified the original cybernetic model in order to become more adaptable to the process of student's knowledge and skill acquisition (see Fig. 1) and we called it a model of instruction with a computerized tutor.

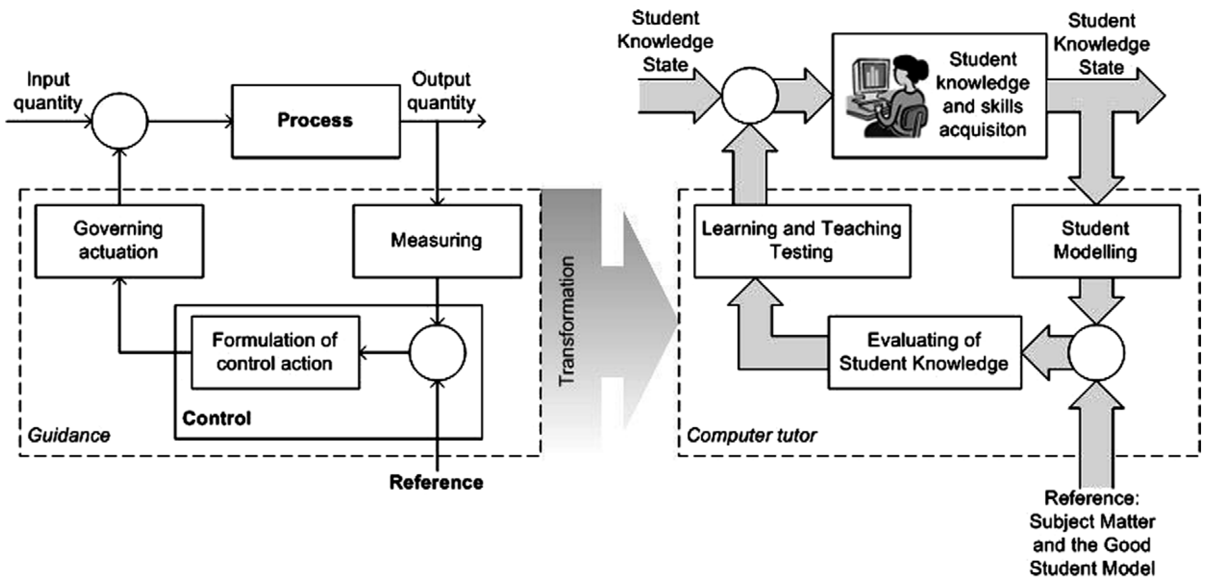

Input quantity
Process
Output quantity
Governing actuation
Measuring
Formulation of control action
Control
Guidance
Reference
Transformation
Student Knowledge State
Student knowledge and skills acquisiton
Student Knowledge State
Learning and Teaching Testing
Student Modelling
Evaluating of Student Knowledge
Computer tutor
Reference: Subject Matter and the Good Student Model

Fig. 1. Cybernetic model to the TEx-Sys model transformation.

The student's knowledge state is a manageable input and an output value of the process, regarding the actual subject matter unit of domain knowledge. The process in the TEx-Sys model is the student's knowledge and skill acquisition. The referent value (reference) is defined through: (i) goals and tasks of the subject matter, which need to be understood and (ii) the "good" student's model based on the evaluation criteria that implicates the cognition of the specified student's knowledge level.

The computer tutor, as a replacement for a "human" tutor, gives feedback in our model and has to: (i) build the student's model and diagnose the student's knowledge (student modeling, VanLehn, 1988), (ii) determine differences between the actual student's knowledge and the referent model, as well as manage the activities' states (evaluating student's knowledge) and (iii) shift the learning and teaching to the next element of the subject matter (learning and teaching) or perform remedial teaching (learning and teaching) and start testing (testing). The student interactively builds a learning and teaching environment according to the Piagetian paradigm "guided free play" (Sugerman, 1978) and the "guided learning by doing" (Merrill, Reiser, Merrill, & Landes, 1995) scenario in accordance with his/her individual capabilities. The TEx-Sys model allows teachers to decompose domain knowledge in order to build hypermedia-rich cross-platform tutorials and learning environments for learning and teaching. Finally, the TEx-Sys model is adaptable to individual needs of both students and teachers.

The pedagogical framework of the TEx-Sys model is expressed with a continuing four-phase activity cycle: didactic, perception, diagnostic and evaluation, and finally help and remediation. The didactic phase involves a number of factors: the subject matter of teaching (What is being taught?), the object of teaching (Who is being taught?) and the method and strategy of adaptation of teaching to students' individual needs (How is being taught?). The perception phase deals with the student's actual knowledge level (the student's knowledge states). The diagnostic and evaluation phase determines the level of student's knowledge. Misconceptions in the student's knowledge activate the help and remediation mechanism to minimize differences between the student's and the domain knowledge.

The TEx-Sys model is knowledge-based because it includes: (1) the domain knowledge with examples and explanations; (2) the teacher's knowledge – the principles used for teaching and the methods used for applying those principles; (3) the student's knowledge that is developed as a result of overlay with the teacher's knowledge, including misconceptions and missing conceptions. This model exhibits intelligent behavior and therefore it has the characteristics of an intelligent system mainly because of the following statements: (i) it deduces or solves a problem in the application environment of a chosen domain knowledge; (ii) it deduces the student's knowledge and skills; (iii) it has the strategy which enables minimizing differences between the student's and the tutor's knowledge. In Section 2.1, we explain the formalism for knowledge presentation in the TEx-Sys model.

2.1. Formalism for knowledge presentation

Formalism for knowledge presentation needs to enable direct modeling techniques that the domain knowledge expert can use during problem solving, as well as explicitly introduce problem solving strategies and techniques. Although the majority of ITSs represent their knowledge using the production rules, the semantic networks have had a great impact on domain knowledge presentation from the early days of the ITS development, due to their ability to express the cognitive model of human memory and reasoning. The formalism for knowledge presentation in the TEx-Sys model is based on the semantic networks with frames and production rules.

Generally speaking, the semantic network design is based on human associative memory, where knowledge is presented by using concepts and relations, and where concepts are nodes and a relation is a link between two nodes. Nodes are used for presentation of the domain knowledge objects, while links show relations between the objects. The lack of formal semantics and standard terminology is perhaps the basic drawback of the semantic networks. The fundamental distinction is the difference between a generic and an individual interpretation of nodes.

Generally, some nodes of the semantic network are taken as descriptors applicable on many individual objects or descriptions (generic nodes), while others are used for presentation of those individual objects or descriptions applicable on the individual objects or descriptions (an individual node).

The TEx-Sys model uses the following semantic primitives for describing relations between generic nodes: is_a, subclass and a_kind_of, and for the relation between a generic and an individual node it uses the semantic primitive instance. The relation part_of shows that some object is a part of some other object as a whole. Besides that, all other types of relations that the domain knowledge expert can define are acceptable. The TEx-Sys model uses the semantic primitive labeled property for showing properties, as well as the Minsky's diagram (Touretzky, 1992) that encodes knowledge in packages, so called frames, which are incorporated in the network with a searching capability. A frame has an optional number of slots, or to be exact, an attribute set (slot) and its values (filler). Besides names and links towards other objects, objects in the knowledge base can also have one of the structural attributes: a textual description, a picture, a slide (or a sequence of slides), a sound and animation as well as, URL addresses. Multimedia and hypertext improve the semantic structure of the knowledge base and engage more students' senses in the learning and teaching process.

2.2. Specification of actors and functionalities

The TEx-Sys model has the following actors: (i) students who are involved in the knowledge and skills acquisition process in an arbitrary domain knowledge, (ii) domain knowledge experts who create knowledge bases, (iii) teachers who use the created knowledge bases to didactically design subject matter or courseware, and finally (iv) the system administrator who monitors the system, the users and the ways of using the system. The TEx-Sys model's actors have the following main functionalities: (i) domain knowledge design, (ii) subject matter or courseware design, (iii) learning, teaching and knowledge testing, (iv) system administrator functionalities.

We have to emphasize that these functionalities have been progressively developed and implemented into the three systems: the on-site TEx-Sys, the DTEx-Sys and the xTEx-Sys, and are discussed in more detail in the following sections. Table 1 illustrates which functionalities have been implemented in which systems, with names of the modules and Web services that achieve these functionalities.

2.2.1. Domain knowledge design

The environment for domain knowledge design involves an ontology which forms the basis for the domain knowledge formalization in the TEx-Sys model. Fig. 2 gives a specification of the ontology for knowledge presentation using the semantic network with frames. The top level of domain knowledge hierarchy is called the

Table 1
Actors and functionalities of developed systems based on the TEx-Sys model

Functionalities	TEx-Sys	DTEx-Sys	xTEx-Sys
Domain knowledge design	Yes Developing module	No	Yes Expert web service
Subject matter or courseware design	No	No	Yes Teacher web service
Student learning and teaching	Yes Learning and teaching module	Yes Learning and teaching module	Yes Student web service
Testing and evaluating	Yes Testing and evaluating module Quiz module	Yes Quiz module	Yes Student web service
Administration	No	Yes Administration module	Yes Administrator web service
Actors	Student, teacher	Student, teacher, administrator	Student, expert, teacher, administrator

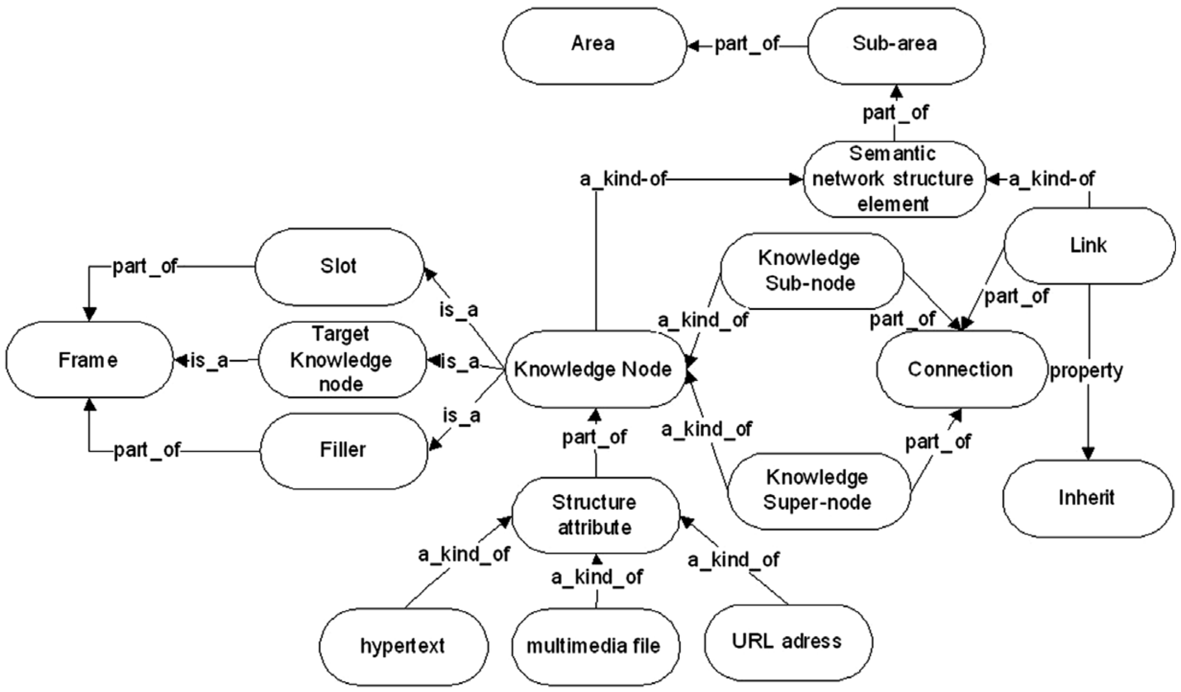


Fig. 2. Ontological specification for the domain knowledge design.

area, every area has sub-areas, and finally all sub-areas contain elementary knowledge objects with knowledge nodes and links from the semantic network that represents particular domain knowledge. The domain knowledge bases design, at the elementary knowledge objects level, is done in compliance with the formalism for knowledge presentation described in Section 2.1.

2.2.2. Subject matter or courseware design

The teacher designs a subject matter or a courseware in a specialized environment where the top level concept is a course. A courseware has a multilayered structure and it is divided into four levels: (i) the first level – a unit; (ii) the second level – a lesson; (iii) the third level – a topic; (iv) the fourth level – an instructional item.

These elements of the courseware structure have been identified according to the pedagogical tradition in many European states, but extended by one new expression, an instructional item which is considered to be an undividable element of the subject matter. A unit generally includes one or more lessons, a lesson includes one or more topics and finally a topic includes one or more instructional items. A quiz-type test can be assigned to a course, a unit, a lesson or a topic, but not to an instructional item. Teachers freely design the courseware structure and it includes both a vertical and a horizontal decomposition of the subject matter elements structure. That means that the courseware developed by the teacher has a tree structure and its elements can be sequenced. Nodes in the courseware tree are the subject matter elements (see Fig. 3).

Knowledge presentation as well as the way of design and the sequence of educational contents is done according to the SCORM (www.adlnet.gov). Knowledge presentation in the TEx-Sys model is based on the semantic network, and the SCORM puts knowledge into SCO's (sharable content object) assets (see Fig. 4). The semantic network nodes are undividable domain knowledge elements, just like assets in the SCORM. We believe that knowledge in the TEx-Sys model, namely because of its semantic network structure,

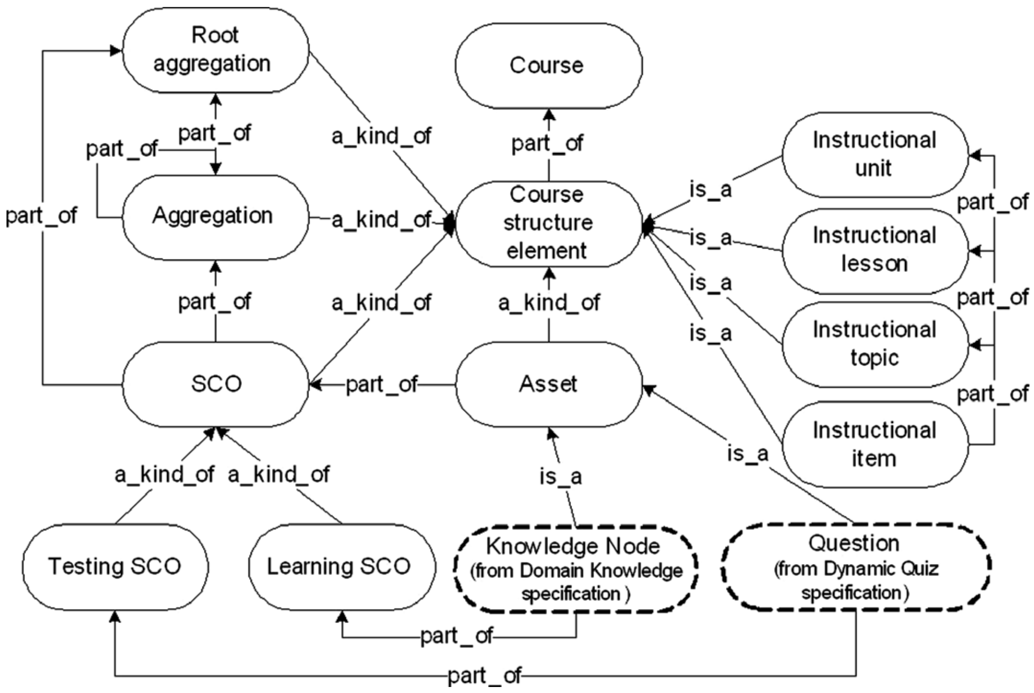


Fig. 3. Ontological specification for subject matter or courseware design.

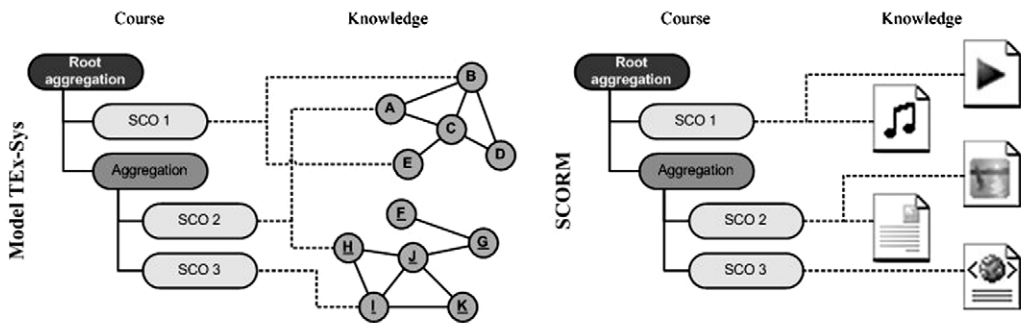


Fig. 4. Differences between the TEx-Sys model and the SCORM content aggregation model.

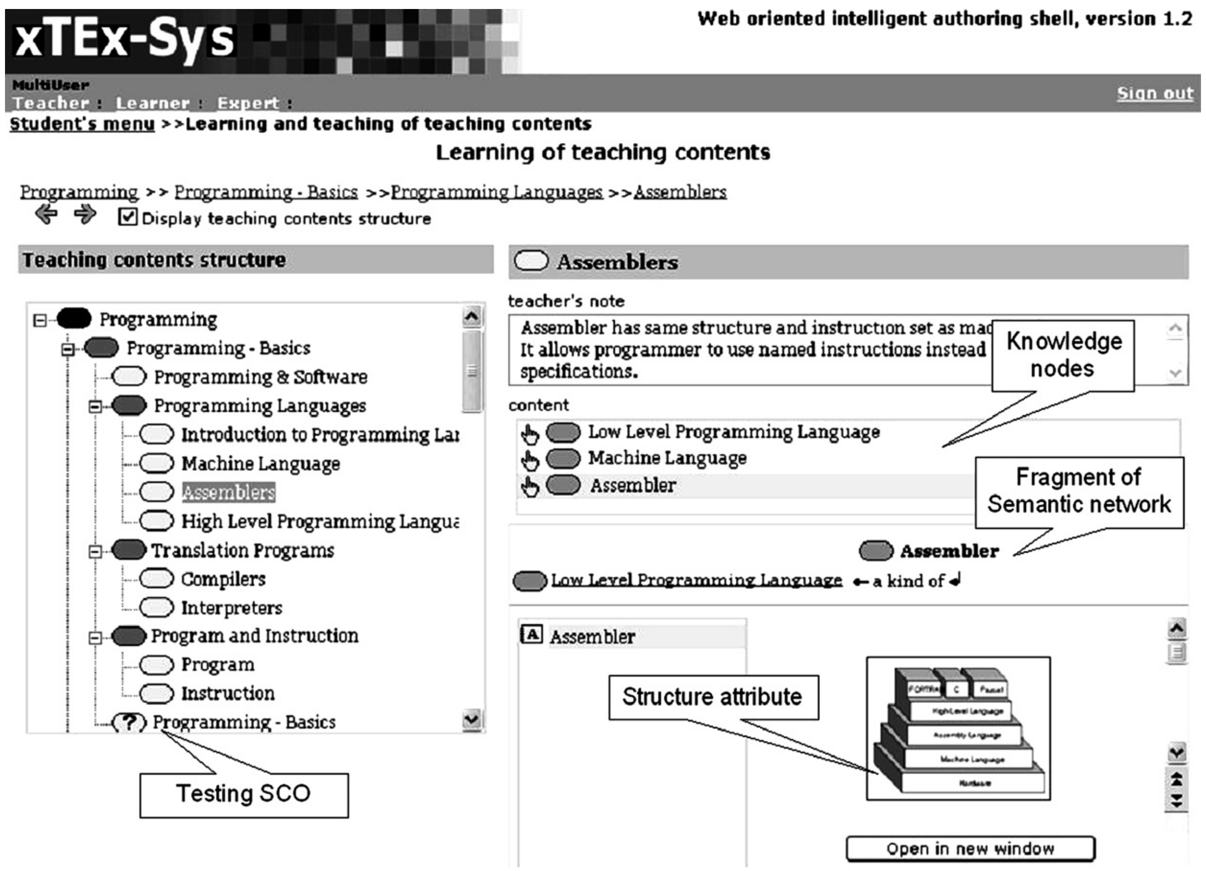
forms the basis for intelligent behavior in the e-learning systems, which is a great advantage when compared to the SCORM. We have to mention that the TEx-Sys model was designed before the SCORM, so its original idea could not have been based on it. However, the sequence of educational contents is done according to the SCORM. That is implemented into our newest TEx-Sys model version, the xTEx-Sys.

2.2.3. Student learning and teaching

Learning, teaching and testing the student's knowledge is done by using an ontology whose specification is defined by the following concepts: a course, elements of the SCORM content aggregation model, a dynamic quiz and a knowledge presentation. The course presented to the student has its educational contents defined by the teacher. The SCORM elements used in the student's ontology are the root aggregation at the course level, the aggregation as an educational content element and the SCO as an undividable educational content element structured using nodes from the domain knowledge bases assigned to the course. The dynamic quiz that is used for the student's knowledge testing is assigned to those educational content elements that the teacher selects. The student's learning and teaching environment is shown in Fig. 5. The most important thing is that students have to be appropriately prepared in order to work with the knowledge presentation formalism that uses the semantic networks with frames.

2.2.4. Knowledge testing and evaluating

The scenario for the student's knowledge testing and evaluation was in the center of our interest during implementation and deployment of the systems based on the TEx-Sys model (the on-site TEx-Sys, the DTEx-Sys and the xTEx-Sys). Consequently we have developed two methods for knowledge evaluation: (i)


xTEx-Sys
Web oriented intelligent authoring shell, version 1.2
MultiUser
Teacher : Learner : Expert :
Sign out
Student's menu >>Learning and teaching of teaching contents
Learning of teaching contents
Programming >> Programming - Basics >>Programming Languages >>Assemblers
Display teaching contents structure
Teaching contents structure
Programming
Programming - Basics
Programming & Software
Programming Languages
Introduction to Programming Lar
Machine Language
Assemblers
High Level Programming Langua
Translation Programs
Compilers
Interpreters
Program and Instruction
Program
Instruction
Programming - Basics
Testing SCO
Assemblers
teacher's note
Knowledge nodes
content
Low Level Programming Language
Machine Language
Assembler
Fragment of Semantic network
Assembler
Low Level Programming Language ← a kind of
Assembler
Structure attribute
Open in new window

Assembler has same structure and instruction set as mac
It allows programmer to use named instructions instead
specifications.

Fig. 5. Student's learning and teaching environment in the xTEx-Sys.

overlaying the student's and the teacher's knowledge (the overlay method) (Goldstein, 1977) and (ii) testing the student's knowledge using the quiz-type tests (the quiz method) (Rosić, 2000). In the next paragraphs we present functionalities of the mentioned knowledge evaluation methods.

2.2.4.1. The overlay method. Overlaying the student's knowledge with the teacher's knowledge, including misconceptions and missing conceptions (VanLehn, 1988) is done by using the following three knowledge bases: (i) the ⟨Expert⟩ knowledge base that contains domain knowledge; (ii) the ⟨Problem⟩ knowledge base that contains a generated subset of nodes and links (e.g., all links are deleted and the student is asked to add them; a fragment of knowledge is generated but some nodes are wrongly connected.) and (iii) the ⟨Solution⟩ knowledge base that contains the student's solution as well as possible misconceptions or missing conceptions. Formalization of the student's knowledge in the TEx-Sys model is based on the same syntax and semantics as the formalization of knowledge presentation. During the problem solving the student in a constructive learning environment can perform the following operations with nodes: delete a node, add a missing node and add a new node (not existing in the ⟨Expert⟩ base). Operations that can be performed with links are: add a new link (with newly entered nodes), delete a correct link, delete an incorrect link, add a correct link, add an incorrect link and add a missing link. Overlaying nodes and connections in the ⟨Expert⟩, ⟨Problem⟩ and ⟨Solution⟩ knowledge bases completes the reconstruction of the learner's actions during creation of the solution. However this process also determines the status of nodes and links in the base ⟨Solution⟩ (see Table 2).

Afterwards, the diagnostic interpreter evaluates the student's knowledge and forms the basis for the overall student's evaluation. Particularly devised point criteria and 170 production rules contained in a specially designed expert system MARK, enable evaluation of the student's knowledge and generation of the student's knowledge grade. The point criterion provides a quantitative and a qualitative description of the student's activity in the problem solving process. The MARK eventually offers the students a description of their success, explanations and recommendations for future work. The overlay method for the student's knowledge evaluation is implemented only in the on-site TEx-Sys.

2.2.4.2. The quiz method. A quiz is an implementation of the knowledge evaluation in which the student gets a set of questions with attached answers which can be correct or incorrect. The student solves the test by marking the answers he assumes to be correct. Quizzes can be classified according to many criteria, but in this model we observe a static or a dynamic background of the offered answers. Static quizzes are the ones with predefined questions. On the other hand, during execution of a dynamic quiz the questions are generated dynamically by the problem generator.

Table 2
Overlay nodes and links status

	⟨Expert⟩	⟨Problem⟩	⟨Solution⟩	Overlay
				E – Expert, P – Problem, S – Solution
Status of nodes				
Added node	1	0	1	$(E \cap S) \backslash P$
Missing node	1	0	0	$E \backslash (P \cup S)$
Deleted node	1	1	0	$(E \cap P) \backslash S$
Node without change	1	1	1	$E \cap P \cap S$
New node	0	0	1	$S \backslash (E \cup P)$
Status of Links				
Deleted incorrect link	0	1	0	$P \backslash (E \cup S)$
Add correct link	1	0	1	$(E \cap S) \backslash P$
Correct given link	1	1	1	$E \cap P \cap S$
Missing link	1	0	0	$E \backslash (P \cup S)$
Incorrect link given	0	1	1	$(P \cap S) \backslash E$
Correct link deleted	1	1	0	$(E \cap P) \backslash S$
Incorrect link added	0	0	1	$S \backslash (E \cup P)$
New link	0	0	1	$S \backslash (E \cup P)$

Implementation of dynamic quizzes in the TEx-Sys model enables adaptability of the testing process to students and has three basic tasks: (i) the quiz generation, (ii) the student's testing using the given quiz and (iii) the evaluation of the student's knowledge according to her/his answers and recommendation for future work. Adaptability is supported by evaluation of every step in the student's testing, along with measurement and diagnostics of the student's knowledge. After each step the system generates new questions depending on the student's partial results. There are three heaviness categories of questions (see Table 3).

Each heaviness category contains four predefined question formats which are defined according to the didactic categories of knowledge. When the student starts solving a dynamic quiz, the first pair of questions is chosen dynamically from the second heaviness category. A pair of incorrectly answered questions results with a new pair of easier questions, while a combination of correctly/incorrectly answered questions results with a new pair of questions from the same heaviness category. However, a pair of correctly answered questions results with a new pair of harder questions. The exception is if the pair of correctly answered questions is from the third heaviness category, then a new pair of questions is generated from the same category. A combination of two incorrectly answered questions from the easiest category results with the quiz ending and the student getting a bad mark.

Table 3
Quiz specification in the TEx-Sys model

Question heaviness category	Question description	Question format	Answer
First heaviness category 1. Point for correct answer	Select the type of connection (direct or indirect) between two nodes	Are ⟨Node1⟩ and ⟨Node2⟩ connected?	– Yes – direct – Yes – indirect – No
	Confirm if two nodes are connected	Are ⟨Node1⟩ and ⟨Node2⟩ connected with ⟨Link⟩?	– Yes – No
	Select the node that has certain structural attribute	What is on the ⟨Structure Attribute⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select if the node has a certain frame	Does ⟨Slot⟩ for ⟨Node⟩ has ⟨Filler⟩?	– Yes – No
Second heaviness category 2. Points for correct answer	Select appropriate connections of a certain node with its parent nodes and child nodes	What is ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select parent nodes of a certain child node that are connected in a certain way	What super-node is connected by ⟨Link⟩ with ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select child nodes of a certain parent node that are connected in a certain way	What sub-node is connected by ⟨Link⟩ with ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select if and what type of connection (direct or indirect) exists between two nodes	How are ⟨Node1⟩ and ⟨Node2⟩ connected?	– Link – Yes – direct – Yes – indirect – No
Third heaviness category 3. Points for correct answer	Select the filler for selected slot of selected frame	What ⟨Slot⟩ has ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select the node that has a certain frame	Whose ⟨Slot⟩ is ⟨Filler⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select parent nodes and appropriate links to a certain child node	Which are the super-nodes of ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)
	Select child nodes and appropriate links to a certain parent node	Which are the sub-nodes of ⟨Node⟩?	Student chooses one or more nodes from the list (one correct and three incorrect)

The heaviness categories enable adaptation of the system to the student, because the student's knowledge determinates every further step. After the last series of questions, the system calculates the final mark based on relation between the accomplished points and the maximal possible points. That result is called rang. The mark function determines the final mark as follows: (i) rang [0%, 50%⟩ for the insufficient mark; rang [50%, 70%⟩ for the sufficient mark; (iii) rang [70%, 80%⟩ for the good mark; rang [80%, 90%⟩ for the very good mark and finally (v) rang [90%, 100%] for the excellent mark. This phase ends with the system's recommendations for the student's further work, thus "closing the loop" by starting another learning cycle. The dynamic quiz method is implemented in all the systems based on the TEx-Sys model.

2.3. Architectures of developed systems based on the TEx-Sys model

Analyses of architecture of the systems derived from the TEx-Sys model gives a development overview of the information and communication technology that has been used for the e-learning systems' implementation in the last decade. Furthermore, we got a general idea about the trends and the e-learning paradigms in the period while we were researching and developing our systems. Fig. 6 presents an architectural comparison of all the systems developed and implemented according to the TEx-Sys model.

Design and implementation of the on-site TEx-Sys is realized by using the 2-tiered architecture as a stand-alone application, without connections and on-line data interchange with the environment. This layered software architecture isolates the user interface and the application logic from the databases. Implementation of data storage using a database has no support for the network-based application usage. The executable files require installation of a runtime engine and all the ActiveX components used in the user interface. Development of the on-site TEx-Sys passed through several versions, and the final one was finished in 2001.

Once the on-site TEx-Sys proved its usability, it was time to start thinking about how to access the domain knowledge bases using the Web. That was the reason why we started developing the DTEx-Sys in 1999 with the following architectural determinants: (i) using technologies for dynamic generation of the Web documents' contents and (ii) using the 3-tier client-server architecture. The technologies for dynamic generation of the Web documents' contents have improved the capabilities of Web-oriented systems and in our case they were used for implementation of the ITS's functions. An important advantage of the 3-tier client-server architecture

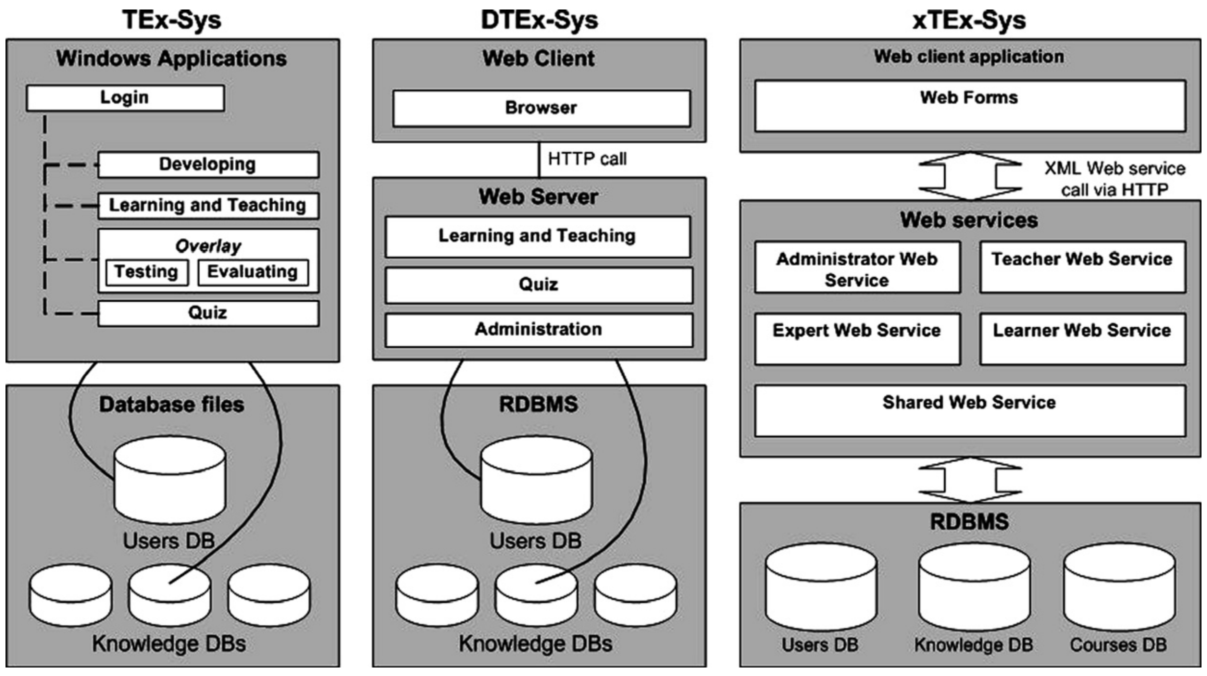


Fig. 6. Architecture of implemented systems based on the TEx-Sys model.

is the possibility of deploying application tiers on several computers thus additionally reducing the system's load and speeding up the application itself. The DTEx-Sys is developed and implemented in the 3-tier client–server architecture where the intelligent tutoring functionalities are separated from the user interface and the knowledge bases. The Web browser is the basic component of the DTEx-Sys user interface. The middle tier (learning and teaching, quiz and administration modules) generates a document to be distributed to the client according to a previously issued client query. The application logic tier is being run on a Microsoft Windows NT platform under the internet information server (IIS) comprising the active server pages (ASP). The application logic tier accesses the databases via the ODBC (open database connectivity) thus enforcing independence from the database management system.

The year 2003 was the beginning of design and implementation of the xTEx-Sys. The xTEx-Sys is a Web oriented system based on a new concept (Rosić, Glavinić, & Žitko, 2004). In this concept, the subject matter or the courseware design functionality is of great importance. The xTEx-Sys has been built in the services-oriented architecture. Therefore, the application logic of the system is implemented through the Web services (administrator service, teacher service, expert services, student service and shared service). Communication with these services is done using the SOAP protocol (simple object access protocol) (Snell, Tidwell, & Kulchenko, 2001).

The services' location, their implementation and the platform where the activated processes are running, are not visible to the object requesting a response from the service. In this way the further system's extensions are made easier, because new services only need to know the definition of the existing ones. This is enabled using the WSDL (web services description language) (Zimmermann, Tomlinson, & Peuser, 2005) while describing the services.

3. The TEx-Sys model applications and effectiveness evaluations

We, like many other e-learning systems developers, have become so involved in making our system work that we have forgotten our original goal: to build an e-learning system that is as good as highly successful human tutors. Moreover, we have paid little attention to the process of evaluation. Since the major goal of an e-learning system is to teach, its evaluation's main test is to determine whether students learn effectively from it (Mark & Greer, 1993).

In the past decade, there were numerous applications of the TEx-Sys model in learning and teaching process that involved students from primary education all the way to academic level. In the period from 2001 to 2007, there were in total 5482 knowledge tests solved by 1302 students while evaluating their understanding of different domain knowledge using one of the systems based on the TEx-Sys model (Table 4).

Questionnaires about students' impressions were given to the students after finishing the courses that were supported by the TEx-Sys model. Qualitative analysis of the questionnaires results revealed that most students were pleased while working with the evaluated system and that they were open minded for embracing that kind of learning and teaching support. On the other hand, qualitative analysis could not determine the effect of educational influence of the TEx-Sys model. Since all instructional software should be evaluated before being used in educational process, we have recently conducted a certain number of experiments in order to evaluate the educational influence of the TEx-Sys model (Table 5). The experimental factor in all experiments was the xTEx-Sys.

Table 4
The number of knowledge tests realized on TEx-Sys model and the number of students involved in every academic year from 2001 to 2007

	TEx-Sys	DTEx-Sys	xTEx-Sys	Total
2001/2002	72 Tests, 18 students	–	–	72 Tests, 18 students
2002/2003	–	648 Tests, 72 students	–	648 Tests, 72 students
2003/2004	–	591 Tests, 153 students	–	591 Tests, 153 students
2004/2005	169 Tests, 119 students	1077 Tests, 165 students	527 Tests, 73 students	1773 Tests, 357 students
2005/2006	–	–	1368 Tests, 552 students	1368 tests, 552 students
2006/2007	–	–	1030 Tests, 150 students	1030 Tests, 150 students
Total	241 Tests, 137 students	2316 Tests, 390 students	2925 Tests, 775 students	5482 tests, 1302 students

Table 5
Results from effectiveness evaluation experiments

Course	Sample size after drop-off	Duration	Statistically significant difference	Original score means and standard deviations	Gain score means and standard deviations	Effect size
Academic year 2005/2006						
Introduction to computer science	Experimental group: 40 1st year students Control group: 40 1st year students	14 weeks	$\alpha = 0.05$, df = 78			0.16
			chk test 1: $t = -0.73$, $p = 0.4676$ No	ctrl: $\overline{X} = 40.72$, sd = 15.78 exp: $\overline{X} = 46.13$, sd = 16.80	ctrl: $\overline{X} = -9.28$, sd = 17.74 exp: $\overline{X} = -6.19$, sd = 18.97	(0.17)
			chk test 2: $t = 2{,}31$ $p = 0.0235$ Yes	ctrl: $\overline{X} = 54.95$, sd = 17.36 exp: $\overline{X} = 46.95$, sd = 12,80	ctrl: $\overline{X} = 4.95$, sd = 21.68 exp: $\overline{X} = -5.36$, sd = 17.86	(−0.47)
			final test: $t = -3.62$, $p = 0.0005$ Yes	ctrl: $\overline{X} = 37.48$, sd = 13.44 exp: $\overline{X} = 51.23$, sd = 12.30	ctrl: $\overline{X} = -12.53$, sd = 14.32 exp: $\overline{X} = -1.09$, sd = 13.66	(0.81)
Chemistry	Experimental group: 20 8th grade primary school pupils Control group: 21 8th grade primary school pupils	10 weeks	$\alpha = 0.05$, df = 39 $t = -1.81$, $p =$ 0.0780 Yes	ctrl: $\overline{X} = 24.42$, sd = 8.16 exp: $\overline{X} = 25.70$, sd = 5.28	ctrl: $\overline{X} = -9.83$, sd = 7.95 exp: $\overline{X} = -8.63$, sd = 6.71	0.60
Physics – optics	Experimental group: 40 8th grade primary school pupils Control group: 40 8th grade primary school pupils	7 weeks	$\alpha = 0.05$, df = 78 exp: $\overline{X} = 62.28$, sd = 20.91	ctrl: $\overline{X} = 44.03$, sd = 22.89 exp: $\overline{X} = 0.34$, sd = 0.20	ctrl: $\overline{X} = 0.16$, sd = 0.24	0.75
Nature and society	Experimental group: 24 2nd grade primary school pupils Control group: 24 2nd grade primary school pupils	6 weeks	$\alpha = 0.05$, df = 46 $t = -2.88$, $p =$ 0.0060 Yes	ctrl: $\overline{X} = 79.83$, sd = 10.58 exp: $\overline{X} = 91.00$, sd = 13.10	ctrl: $\overline{X} = 8.17$, sd = 8.30 exp: $\overline{X} = 14.83$, sd = 7.69	0.80
	Experimental group: 24 3rd grade primary school pupils Control group: 24 3rd grade primary school pupils	6 weeks	$\alpha = 0.05$, df = 46 $t = -3.08$, $p =$ 0.0035 Yes	ctrl: $\overline{X} = 86.17$, sd = 7.64 exp: $\overline{X} = 95.50$, sd = 5.32	ctrl: $\overline{X} = 8.67$, sd = 7.24 exp: $\overline{X} = 15.17$, sd = 7.36	0.83
	Experimental group: 20 4th grade primary school pupils Control group: 20 4th grade primary school pupils	6 weeks	$\alpha = 0.05$, df = 38 $t = -3.48$, $p = 0.0013$ Yes	ctrl: $\overline{X} = 84.00$, sd = 10.24 exp: $\overline{X} = 92.83$, sd = 7.82	ctrl: $\overline{X} = 10.83$, sd = 6.57 exp: $\overline{X} = 18.17$, sd = 7.98	1.11
Academic year 2006/2007						
Introduction to computer science	Experimental group: 20 1st year students Control group: 19 1st year students	14 weeks	$\alpha = 0.05$, df = 37			0.42
			chk test 1: $t = 1.04$, $p = 0.3051$ No	ctrl: $\overline{X} = 54.74$, sd = 19.62 exp: $\overline{X} = 50.30$, sd = 18.62	ctrl: $\overline{X} = 13.74$, sd = 19.62 exp: $\overline{X} = 7.35$, sd = 18.62	(0.33)
			chk test 2: $t = -1.11$, $p = 0.2742$ No	ctrl: $\overline{X} = 31.89$ sd = 23.30 exp: $\overline{X} = 42.05$, sd = 22.78	ctrl: $\overline{X} = -9.11$, sd = 23.30 exp: $\overline{X} = -0.90$, sd = 22.78	(0.35)
			final test: $t = -3.77$, $p = 0.0006$ Yes	ctrl: $\overline{X} = 40.79$, sd = 11.79 exp: $\overline{X} = 57.20$, sd = 12.14	ctrl: $\overline{X} = -0.21$, sd = 11.79 exp: $\overline{X} = 14.25$, sd = 12.14	(1.23)

QBASIC programming	Experimental group: 20 1st year students Control group: 19 1st year students	14 weeks	$\alpha = 0.05$, df = 37			0.45
			chk test 1: $t = -1.27$, $p = 0.2110$ No	ctrl: $\overline{X} = 59.94$, sd = 27.24 exp: $\overline{X} = 61.72$, sd = 28.19	ctrl: $\overline{X} = 36.53$, sd = 25.82 exp: $\overline{X} = 46.20$, sd = 21.28	(0.37)
			chk test 2: $t = -0.94$, $p = 0.3533$ No	ctrl: $\overline{X} = 52.72$, sd = 23.78 exp: $\overline{X} = 51.89$, sd = 25.35	ctrl: $\overline{X} = 29.37$, sd = 22.11 exp: $\overline{X} = 35.60$, sd = 19.08	(0.28)
			final test: $t = -1.83$. $p = 0.0753$ Yes	ctrl: $\overline{X} = 43.61$, sd = 23.90 exp: $\overline{X} = 49.22$, sd = 23.66	ctrl: $\overline{X} = 21.74$, sd = 15.23 exp: $\overline{X} = 32.60$, sd = 21,47	(0.71)
Mathematics	Experimental group: 9 6th grade primary school pupils Control group: 9 6th grade primary school pupils	7 weeks	$\alpha = 0.05$, df = 16			1.32
			chk test 1: $t = -2.07$, $p = 0.0550$ Yes	ctrl: $\overline{X} = 38.67$, sd = 14.19 exp: $\overline{X} = 54.33$ sd = 21.11	ctrl: $\overline{X} = -34.33$, sd = 11.85 exp: $\overline{X} = -19.11$, sd = 18.62	(1.28)
			final test: $t = -2.33$. $p = 0.0332$ Yes	ctrl: $\overline{X} = 30.00$, sd = 15.81 exp: $\overline{X} = 49.44$, sd = 23.78	ctrl: $\overline{X} = -43.00$, sd = 13.93 exp: $\overline{X} = -24.00$, sd = 20.11	(1.36)
	Experimental group: 9 8th grade primary school pupils Control group: 9 8th grade primary school pupils	7 weeks	$\alpha = 0.05$, df = 16			0.15
			Chk test 1: $t = -0.30$. $p = 0.7680$ No	ctrl: $\overline{X} = 57.22$, sd = 24.12 exp: $\overline{X} = 58.89$, sd = 22.93	ctrl: $\overline{X} = 6.67$, sd = 13.78 exp: $\overline{X} = 8.44$, sd = 10.98	(0.13)
			final test: $t = -0.24$, $p = 0.8134$ No	ctrl: $\overline{X} = 67.78$, sd = 22.93 exp: $\overline{X} = 70.00$, sd = 30.92	ctrl: $\overline{X} = 17.22$, sd = 14.12 exp: $\overline{X} = 19.55$, sd = 25.56	(0.17)
	Experimental group: 24 5th grade primary school pupils Control group: 24 5th grade primary school pupils	5 weeks	$\alpha = 0.05$, df = 39			0.38
			Chk test 1 $t = -0.19$, $p = 0.8503$ No	ctrl: $\overline{X} = 61.00$, sd = 17.07 exp: $\overline{X} = 77.40$, sd = 16.61	ctrl: $\overline{X} = -9.10$, sd = 15.20 exp: $\overline{X} = -8.05$, sd = 18.67	(0.07)
			Chk test 2 $t = -2.16$, $p = 0.0370$ Yes	ctrl: $\overline{X} = 61.00$, sd = 20.78 exp: $\overline{X} = 85.00$, sd = 13.27	ctrl: $\overline{X} = -12.70$, sd = 17.46 exp: $\overline{X} = 0.14$, sd = 20.53	(0.74)
			final test: $t = -1.03$, $p = 0.3094$ No	ctrl: $\overline{X} = 69.50$, sd = 15.26 exp: $\overline{X} = 85.00$, sd = 15.36	ctrl: $\overline{X} = -5.65$, sd = 12.87 exp: $\overline{X} = -1.29$, sd = 14.36	(0.34)

Some of those experiments (the ones with reported only one effect size) have been conducted according to classical two-group experimental design with pre-and-post test. Those experiments, with several reported effect sizes – an average effect size and partial effect sizes (reported in brackets) – were conducted according to modified experimental design, where we have combined classical two-group experimental design with pre-and-post test and a factorial design. We have added arbitrary number of checkpoint-tests to determine the effectiveness in intermediate states, that is, to determine the extent to which knowledge and understanding has been improved by the educational intervention in several phases of the experiment. This intermediary effectiveness evaluation distinguishes our approach from the others, and we have named this experimental design a pre-and-post test control group experimental design with checkpoint-tests (Grubišić, Stankov, & Žitko, 2006).

Our experimental design is structured as follows: at the beginning of an experiment (Fig. 7), initial states Si_{1A} and Si_{2B}, and respectively their means Xi_{1A} and Xi_{2B}, should be captured using 45 min pre-test before introducing experimental factors to determine some individual starting level of knowledge or understanding. The group A should use an e-learning system (experimental factor F_1) and the group B should be involved in traditional learning and teaching process (experimental factor F_2) in every learning and teaching cycle. At the end of each cycle, approximately every 4 weeks, final states Sf_{11A}, $Sf_{12A}, \ldots, Sf_{1(n-1)A}$, Sf_{21B}, $Sf_{22B}, \ldots,$ $Sf_{1(\ n-1)B}$ and respectively their means Xf_{11A}, $Xf_{12A}, \ldots, Xf_{1(n-1)A}$, Xf_{21B}, $Xf_{22B}, \ldots, Xf_{1(n-1)B}$, should be captured using the exact comparable 45 min $n-1$ checkpoint tests, in order to calculate a partial effect size of the e-learning system as an experimental factor. And, finally, at the end of the experiment, final states Sf_{1nA} and Sf_{2nB}, and respectively their means Xf_{1nA} and Xf_{2nB}, should be captured using 45 min post-test, in order to calculate the last partial effect size of an evaluated e-learning. All tests used in the experiment are scored on 0–100 scale. In this kind of the experiment, the dependent variable is students' knowledge and the independent variable is different way of treatment. The partial effect sizes are, in fact, standardized mean differences, calculated by dividing the difference between the experimental and the control group means of gains by the standard deviation of the control group. The average effect size in our approach is calculated as the arithmetic average of partial effect sizes.

There were in total 11 experiments that were conducted in order to calculate the xTEx-Sys effectiveness. The majority of them (eight experiments) involved primary school pupils. We only left out the first grade pupils, and that is something that has to be done in the future. It is very interesting that primary school pupils have embraced learning and teaching process with the xTEx-Sys better than university students, what can be seen while observing effect sizes. This observation needs to be elaborated and verified through additional experiments.

One experiment with 1st year students that enrolled 'Introduction to Computer Science' course in academic year 2005/2006, was replicated the next academic year. The replication is the repetition of an experiment as closely following the original experiment as possible and is considered to be critical aspect of the scientific method (Litoiu, Rolia, & Serazzi, 2000). Although the results of the two studies are promising, we expected to get larger average effect sizes. A reasonable explanation for the small, or even negative partial effect sizes, could be that the xTEx-Sys's domain knowledge presentation is rather novel for students and therefore difficult to grasp and apply in earlier phases of experiment. When students get familiarized with the system's knowledge presentation, the system itself is very efficient (large post-test partial effect sizes for both experiments). As a consequence, in future experiments, the presentation of the xTEx-Sys should be improved.

Results gained through the conducted experiments have shown a need for adding some extended functions for courseware development and learning management in the xTEx-Sys in order to get it as close as possible to the Bloom's 2-sigma target (Bloom, 1984) which means that the average tutored student was about two standard deviations (2-sigma) above the average control group one. This research had subsequently started an avalanche of research seeking ways of accomplishing this result under more practical and realistic conditions than one-to-one tutoring with human teachers. One possible solution for, as stated by Bloom, the 2-sigma problem, is the usage of ITSs, that provide each student with a learning experience similar to the ideal one-to-one tutoring.

Another paper that provides a referent point for everyone who wants to evaluate the effectiveness of a certain e-learning systems, is Fletcher's research (Fletcher, 2003) where he summarized some research findings for technology-based instruction: computer-based instruction 0.39 sigma (233 studies), interactive multimedia

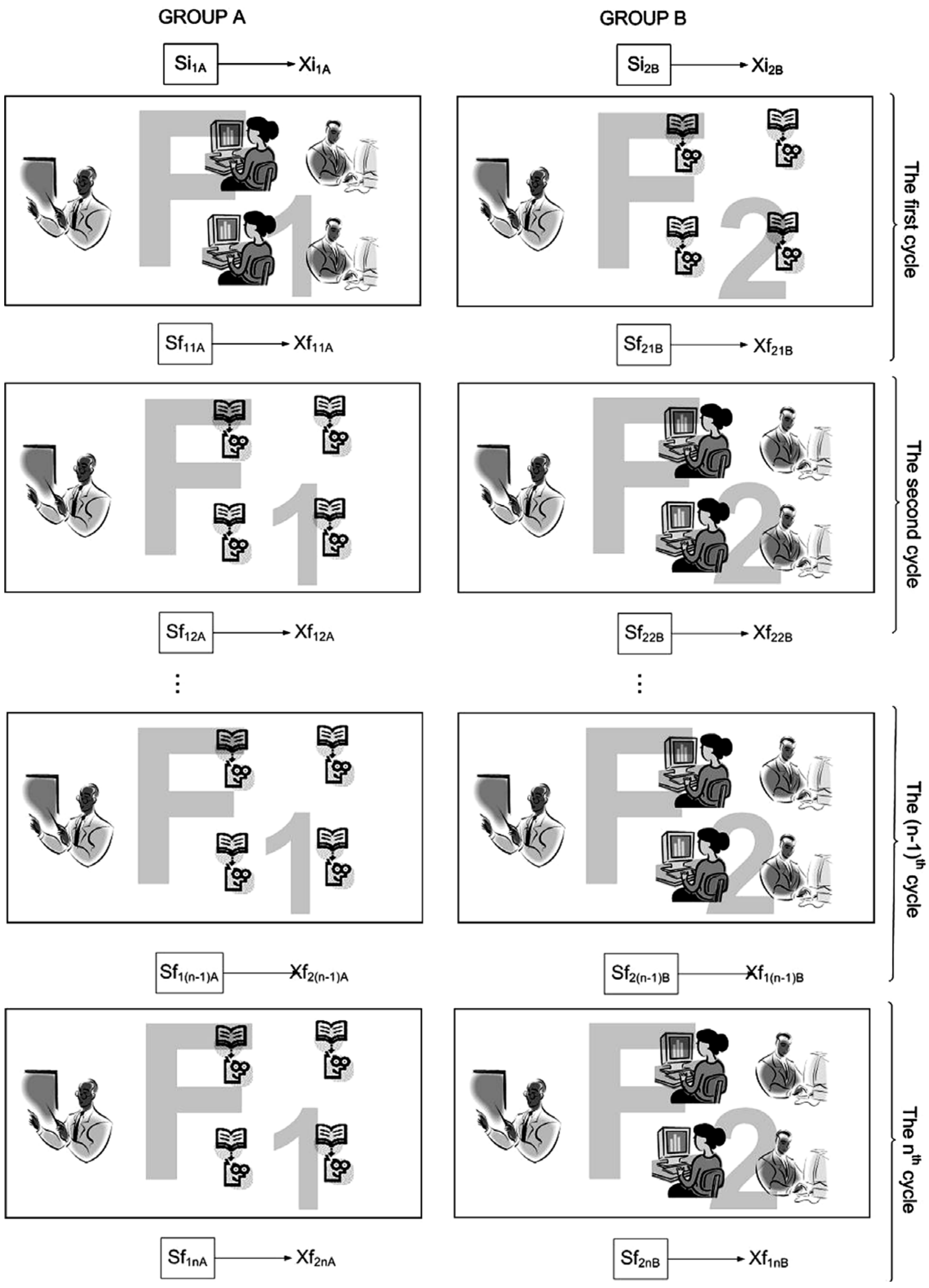


Fig. 7. Pre-and-post test control group experimental design with checkpoint tests.

instruction 0.50 sigma (47 studies), ''intelligent'' tutoring systems 0.84 sigma (11 studies), recent intelligent tutoring systems 1.05 sigma (5 studies). These results suggest steady progress in learning outcomes.

4. Conclusion

The research on the TEx-Sys model that is presented in this paper can be summarized as follows: (i) research and development of the on-site TEx-Sys, the DTEx-Sys and the xTEx-Sys, influenced by the constant evolution of technologies, tools and methods, (ii) usage of these systems by students from a primary education to an academic level and finally (iii) development of a special methodology for evaluating the systems' educational influence using the students' results in the learning and teaching process.

Currently, we are at the end of one research phase in the area of intelligent tutoring systems, and we have clear directions for future TEx-Sys model research and development: (i) knowledge representation supported by natural language processing, specially natural language generator, (ii) mobile computing environment and (iii) student's environment for learning programming skills. All these areas require also new approach to their effectiveness evaluation. To conclude, we briefly elaborate on each of mentioned directions.

First, the students involved in TEx-Sys model's learning and teaching process usually have two major objections. If the student is first time user of TEx-Sys then he has to know the way that knowledge is represented, meaning he has to be familiar with the structure of semantic network with frames. For such students we usually made short pre-lecture about such formalism. We have observed that knowledge represented by semantic network is more easily accepted by the student from elementary school than the older ones. Another objection refers to the tests which are realized by dynamic question and answer generation. In fact, generated questions have incorrect syntax from a natural language point of view and it is sometimes very hard for student to understand its meaning. In order to deal with these problems, we have chosen Web Ontology Language (OWL) because of its nature to provide graphical observation of described knowledge, as well as expressiveness realized by expressions equivalent to description logic axioms and facts. To provide fluent and intelligible reading of domain knowledge we have to involve natural language processing techniques between knowledge formalism and the student. Natural language generation system is the key component for such task. The main task of this system is converting knowledge formalized by OWL to natural language sentences. Another important task is to generate questions and answers from the test templates. One step forward to make TEx-Sys even better is to incorporate agent with conversation dialogue ability for tutoring the student. Whatever discovers emerges on a field of intelligent tutoring, we believe that activating natural language processing in TEx-Sys will make it much more acceptable and easy to use.

Second, the systems derived from the TEx-Sys model are not customized for eminently mobile devices such as mobile phones or personal digital assistants (PDAs). The mobile computing environment is namely very different with respect to the present traditional distributed systems milieu; bandwidth, delay, error rate, interference and the like, may change dramatically as a user moves from one location to another, thus changing the computing environment. The TEx-Sys mobile instance will be targeted to fit into a mobile learner's daily routine without disrupting her/his other activities, but conversely enhancing the effectiveness of learning in the context of handheld terminals of restricted capabilities. Our intention is to achieve transfer from TEx-Sys model to mobile computing environment by assigning the tasks of classical intelligent tutoring system to the personal agents. We consider that such a transfer into the mobile computing environment is natural because each module of an intelligent tutoring system can be regarded as a personal agent.

Third, problems in programming teaching motivated us to approach developing a prototype model of computerized tutor for learning and teaching programming. The goal of intelligent tutoring system for teaching procedural languages programming skills (BASIC, C, Pascal) is to help student to build a cognitive model for solving programming problems. In order to accomplish this goal, a tutoring system has to: (i) define programming tasks; (ii) determine from student's behaviour what he knows and misunderstanding he has; (iii) interrupt student's work when his choice leads to errors; (iii) give help on student's demand; (iv) verify correctness of student's program; (v) estimate student's knowledge and skill and advise what he should do and when he should proceed to new material. The knowledge contained in knowledge base, as well as, programming tasks are universal, and the system has to adapt to selected programming language.

Finally, in order to develop and improve the TEx-Sys model, and verify the effectiveness of the systems that will be developed according to mentioned directions for the TEx-Sys model research and development, further experiments must be conducted. The following questions should be addressed by prospect experiments: (i) are the experiments' results influenced by subjects more than the system itself; (ii) is the system evenly effective regardless of domain knowledge; (iii) could the TEx-Sys model be further improved in order to produce a more positive impact in every stage of the experiment. The results from conducted experiments, as well as, results from new experiments that will be done in the future, will enable conduction of meta-analyses for determining overall effectiveness of TEx-Sys model regardless of the domain knowledge and age of the subjects used in studies.

Acknowledgement

This work has been carried out within projects 0177110 ‘Computational and didactical aspects of intelligent authoring tools in education’, 177-0361994-1996 ‘Design and evaluation of intelligent e-learning systems’ and TP-02/0177-01 ‘Web oriented intelligent hypermedia authoring shell’, funded by the Ministry of Science, Education and Sports of the Republic of Croatia.

References

Ainsworth, S., Major, N., Grimshaw, S., Hayes, M., Underwood, J., & Wood, B. W. D. (2003). REDEEM, simple intelligent tutoring systems from usable tools. In T. Murray, S. Blessing, & S. E. Ainsworth (Eds.), *Authoring tools for advanced technology learning environment* (pp. 205–232). Kluwer Academic Publishers.

Alpert, S. R., Singley, M. K., & Fairweather, P. G. (1999). Deploying intelligent tutors on the web: An architecture and an example. *International Journal of Artificial Intelligence in Education, 10*(2), 183–197.

Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). Intelligent tutoring systems. *Science, 228*, 456–462.

Barr, A., & Feigenbaum, E. A. (1986). *The handbook of artificial intelligence – Volume I. Chapter III: Knowledge Representation*. Addison–Wesley Publishing Company, Inc.

Bloom, B. S. (1984). The two-sigma problem: the search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher, 13*, 4–16.

Božičević, J. (1980). *Fundamentals of automatic control, part I – system approach and control* (10th ed.). Zagreb: Školska knjiga.

Brusilovsky, P., Schwarz, E., & Weber, G. (1996). ELM-ART: An intelligent tutoring system on world wide web. In C. Frasson, G. Gauthier, & A. Lesgold (Eds.). *Intelligent tutoring systems lecture notes in computer science* (Vol. 1086, pp. 61–269). Berlin: Springer.

Burns, H. L., & Capps, C. G. (1988). Foundations of intelligent tutoring systems: An introduction. In M. C. Poison & J. J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 1–19). London: Lawrence Eribaum.

Carbonell, J. R. (1970). AI in CAI: an artificial-intelligence approach to computer-assisted instruction. *IEEE Transaction On Man–Machine Systems, MMS-11*(4), 190–202.

Cohen, P. A., Kulik, J. A., & Kulik, C. L. C. (1982). Educational outcomes of tutoring: a meta-analysis of findings. *American Educational Research Journal, 19*(2), 237–248.

Fletcher, J. D. (2003). Evidence for learning from technology-assisted instruction. In H. F. O’Neal & H. F. Perez (Eds.), *Technology applications in education: A learning view* (pp. 79–99). Mahwah, NJ: Lawrence Erlbaum Associates.

Goldstein, I. P. (1977). *Overlays: A theory of modeling for computer aided instruction.* AI Memo 406. Cambridge, MA: MIT.

Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H., Ventura, M., Olney, A., et al. (2004). AutoTutor: a tutor with dialogue in natural language. *Behavioral Research Methods, Instruments, and Computers, 36*, 180–193.

Gruber, T. R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition, 5*(2), 199–220.

Grubišić, A., Stankov, S., & Žitko, B. (2006). An approach to automatic evaluation of educational influence. In Proceedings of the 6th WSEAS International Conference on Distance Learning and Web Engineering (DIWEB 06) (pp. 20–25). Lisabon, Portugal, September 22–24.

Hartley, J. R., & Sleeman, D. H. (1973). Towards more intelligent teaching systems. *International Journal of Man–Machine Studies, 5*, 215–236.

Litoiu, M., Rolia, J., & Serazzi, G. (2000). Designing process replication and activation: a quantitative approach. *IEEE Transactions on Software Engineering, 26*(12), 1168–1178.

Mark, M. A., & Greer, J. E. (1993). Evaluation methodologies for intelligent tutoring systems. *Journal of Artificial Intelligence and Education, 4*(2/3), 129–153.

Merrill, D. C., Reiser, B. J., Merrill, S. K., & Landes, S. (1995). Tutoring: Guided learning by doing. *Cognition and Instruction, 13*(3), 315–372.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: a comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences, 2*(3), 277–306.

Mitrovic, A. (2003). An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education, 13*, 171–195.

Murray, T. (1999). Authoring intelligent tutoring systems: an analysis of the state of the art. *International Journal of Artificial Intelligence in Education, Vol. 10*, 98–129.

Murray, T., Blessing, S., & Ainsworth, S. E. (Eds.). (2003). *Authoring tools for advanced technology learning environments*. Amsterdam: Kluwer Academic Publishers.

Nakabayashi, K., Maruyama, M., Kato, Y., Touhei, H., & Fukuhara, Y. (1997). Architecture of an intelligent tutoring system on the WWW. In B. D. Boulay & R. Mizoguchi (Eds.), *Proceedings of AI-ED'97, World Conference on Artificial Intelligence in Education* (pp. 39–46). Kobe, Japan. IOS, Amsterdam.

Ohlsson, S. (1986). Some principles of intelligent tutoring. *Instructional Science, 14*, 293–326.

Pask, G. (1965). A cybernetic model of concept learning. In *Proceedings of 3rd, congress international association cybernetics*. Gauthier-Villars.

Rickel, J. W. (1989). Intelligent computer-aided instruction: a survey organized around system components. *IEEE Transaction on System, Man, and Cybernetics, 19*(1), 40–57.

Rosić, M. (2000). Establishing of distance education systems within the information infrastructure. Master's thesis, Faculty of Electrical Engineering and Computing. Zagreb, Croatia: University of Zagreb.

Rosić, M., Glavinić, V., & Žitko, B. (2004). Intelligent authoring shell based on web services. In S. Nedevschi & I. J. Rudas (Eds.), *Proceedings of the 8th international conference on intelligent engineering systems (INES 2004)* (pp. 50–55). Cluj-Napoca, Romania: Technical University of Cluj-Napoca.

Self, J. (1974). Student models in computer aided instruction. *International Journal of Man–Machine Studies, 6*, 261–276.

Self, J. (1990). Theoretical foundations for intelligent tutoring systems. *Journal of Artificial Intelligence in Education, 1*(4), 3–14.

Shute, V. J., & Psotka, J. (1995). Intelligent tutoring systems: past, present, and future. In Jonassen, D. H. (Ed.), *Handbook of research on educational communications and technology* (pp. 570–600).

Skinner, B. E. (1954). The science of learning and the art of teaching. *Harvard Educational Review, 24*, 86–97.

Skinner, B. F. (1986). Programmed instruction revisited. *Phi Delta Kappan, 68*(2), 103–110.

Sleeman, D. H., & Brown, J. S. (1982). Introduction – intelligent systems. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems* (pp. 1–10). London (Ltd): Academic Press, Inc.

Sleeman, D. H., & Brown, J. S. (1979). Editorial: intelligent tutoring systems. *International Journal of Man–Machine Studies, 11*, 1–3 (January 1979).

Snell, J., Tidwell, D., & Kulchenko, P. (2001). *Programming web services with SOAP* (1st ed.) O'Reilly Media.

Stankov, S. (1997). *Isomorphic model of the system as the basis of teaching control principles in an intelligent tutoring system*. Doctoral dissertation, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Croatia: University of Split, Split.

Stankov, S. (2005). *Technology project TP-02/0177-01 Web oriented intelligent hypermedia authoring shell*. Ministry of Science and Technology of the Republic of Croatia, 2003–2005.

Stern, M. K. (1997). *Web based intelligent tutors derived from lecture based courses*. A Dissertation Proposal Presented December, Computer Science Department, University of Massachusetts, Amherst.

Sugerman, R. (1978). 'What's new, teacher?' Ask the computer. *IEEE Spectrum*(September), 44–49.

Touretzky, D. S. (1992). Inheritance hierarchy. In I. Shapiro & C. Stuart (Eds.), *Artificial intelligence – encyclopedias* (pp. 690–701). New York: John Wiley & Sons, Inc.

VanLehn, K. (1988). Student modeling. In M. C. Polson & J. J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 55–78). New Jersey: Lawrence Erlbaum Associates Publishers.

Vassileva, J. (1997). Dynamic Course Generation on the WWW. In B. D. Boulay & R. Mizoguchi (Eds.), *Proceedings of AI-ED'97, 8th world conference on artificial intelligence in education* (pp. 498–505). Kobe, Japan. IOS, Amsterdam.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge*. Los Altos, CA: Morgan Kaufman.

Woolf, B. (1992). AI in education. In S. Shapiro (Ed.), *Encyclopedia of artificial intelligence* (pp. 434–444). New York: John Wiley & Sons, Inc.

Zimmermann, O., Tomlinson, M. R., & Peuser, S. (2005). *Perspectives on web services: Applying SOAP WSDL and UDDI, to real-world projects*. Springer.

Slavomir Stankov is an associate professor of computer science. Since 1996 he has been heading three scientific and one technological project. He has participated on some projects related to the information and communication technology applications in education, in particular the intelligent tutoring systems and authoring shells for their design, implementation and deployment. He has authored or co-authored more than seventy scientific and expert papers. He is a member of the Croatian Systems Society.

Marko Rosić is an assistant professor of computer science. His scientific work is related to the information and communication technology applications in the distance learning systems. Researching possibilities of using agents and the semantic Web technologies for developing the distance learning systems based on the intelligent tutoring systems is in the centre of his interest. He participates in many scientific and technological projects related to the field of his scientific interest. He has authored or co-authored about twenty scientific papers.

Branko Žitko is a scientific novice with a master's theses who works on the scientific project related to computational and didactical aspects of intelligent authoring tools in education and Web oriented intelligent hypermedia authoring shell. He researches the intelligent e-learning systems' architecture, especially methods and techniques for the natural language processing. He participated in several scientific conferences and specializations and has authored or coauthored about ten papers.

Ani Grubišić is a scientific novice with a master's theses who works on the scientific project related to computational and didactical aspects of intelligent authoring tools in education and Web oriented intelligent hypermedia authoring shell. Her scientific interest is related to the e-learning systems' educational influence evaluation. She has authored or co-authored about ten scientific papers about the student modeling and effectiveness evaluation.