2003

An Intelligent Problem Solving Environment in the Domain of Electrical Engineering

Vera YAKIMCHUK, Hilke GARBE, Heinz-Jürgen THOLE
Kuratorium OFFIS e.V.
R&D Division Safety Critical Systems
Escherweg 2
D-26121 Oldenburg, GERMANY
{Yakimchuk, Garbe, Thole}@offis.de

Claus MÖBUS
Carl von Ossietzky University Oldenburg
Department of Computing Systems
Innovative Learning and Teaching Systems
D-26111 Oldenburg, GERMANY
Claus.Moebus@informatik.uni-oldenburg.de

Edwin WAGNER
Technical University of Ilmenau
Faculty of Electrical Engineering and Information Technology
Institute of General Electrical Engineering
Helmholtzplatz 2, PF 10 05 65
D-98684 Ilmenau, GERMANY
Edwin.Wagner@tu-ilmenau.de

Abstract. In our paper we present a special kind of ITS for general electrical engineering which is provided in the context of the BMBF-project[1] mile [1]. The objective of this project is the cooperative development of web-based e-learning content on the domains of general electrical engineering, technical mechanics, and multimedia and communication science. As a part of this project we provide an intelligent problem solving environment (IPSE) for a topic area from the basics of general electrical engineering. The objective of our work is to develop a knowledge-based learning environment, which supports the learner throughout the whole process of problem solving of selected topics of the basics of general electrical engineering. Our work is based on cognitive science research (ISP-DL[2]-Theory) of our group and the development of IPSEs in various domains as well as on the competence of the partner universities in general electrical engineering and their elaborated task collections.

In contrast to previous projects no formal specification of the tasks exists. Therefore, we classified the tasks according to their informal specified goals and elaborated a new model of domain specific tasks. This model is used to define goal-mean-relations (gmr) which are used for the *elaboration of the planning objectives* and for the symbolic computation of the solutions. Compared to gmrs previously used in IPSEs the gmrs in our IPSE are more complex because of the different types of task goals. The gmrs form the core of our generative expert system, which is used to offer the students adaptive support while solving tasks according to the ISP-DL-Theory.

After an introduction a brief overview of ISP-DL and IPSE is given. Following the working environment implemented in JAVA and the diagnostic component implemented in PROLOG are presented in detail. Finally, the further work and the different application possibilities are described.

Keywords Intelligent problem solving environment, adaptive hypotheses testing, electrical engineering education, models of tasks, knowledge representation, cognitive diagnosis, adaptive environment

---

[1] The German Federal Ministry of Education and Research finances mile by means of the NMB funding program "Neue Medien in der Bildung" (New Media in Education).

[2] Impasse-Success-Problem-Solving-Driven-Learning

Introduction

In the last years a lot of e-learning content has been developed for the basics of general electrical engineering. Many of them are instructional or they visualize experiments or the fundamentals of electrical engineering, physics and so on. Some of them contain tasks the learner has to solve. But most of them just check symbolic or numeric values as results. They don't analyse the steps of the solution path or incomplete solution proposals. So they can't give situation-adapted help, if the user is in an impasse while solving the task. Other programs used by students are computer algebra systems (CAS) like Mathematica or MathCad [2, 3] and circuit editors like PSPICE or CircuitMaker. These programs are powerful instruments but they can't analyse if an equation or a circuit is the solution for a given task. CAS check if the given equations are algebraic correct, calculate numeric or symbolic solutions and draw graphs. Circuit editors simulate circuits if they are electrical correct. But they don't give answers to the questions: "Is this circuit resp. equation a solution for the task?" or "Is my proposal a part of the solution?". Furthermore, for our tasks typically circuits as well as equations belonging to the circuits are demanded.
In the context of the BMBF-project mile (multimedia learning environment) we provide an IPSE for a topic area from the basics of general electrical engineering. The IPSE should help the student to acquire procedural knowledge from this domain, the declarative knowledge can be derived from the web-based learning modules of the partner universities. Our work is based on cognitive science research and the development of IPSEs in various domains as well as on the competence of the partner universities in electrical engineering [4] and their extensive task collections. The objective of our project is to develop a knowledge-based learning environment, which supports the learner throughout the whole process of problem solving of selected topics of the basics of general electrical engineering. To reach this aim a working environment is needed, that enables the user for example to edit circuits and equations. In doing so he should not be constrained to typical solutions, he should be as free as possible in solving the tasks. The system will be able to analyse even incomplete solution proposals and give adapted help.
To avoid design errors the design of IPSEs should be guided by a theory of knowledge acquisition. Our work is based on ISP-DL-Theory an acronym for "Impasse-Success-Problem-Solving-Driven-Learning".

1. ISP-DL and IPSE

1.1 ISP-DL

ISP-DL [5,6,7] (Impasse - Success - Problem - Solving - Driven - Learning) is a cognitive meta-learning theory, influenced by cognitive theories of Newell [8] and Van Lehn [9] (impasse driven learning), Anderson [10,11] (success driven learning), as well as the motivational "Rubikon" theory of Heckhausen [12] and Gollwitzer [13] (problem solving phases).
From our own empirical investigations [14] we concluded that it is fruitful to describe learning as an interplay of impasse- and success-driven learning.
Learning has two aspects: the process of knowledge optimization occurs after a solution has been found. The process is deductive in the sense that the new, optimized knowledge is a logical consequence of old knowledge [15]:

$$\text{Background knowledge} \cup \text{evidence} \models \text{optimized knowledge}$$

The more interesting knowledge acquisition process occurs after solutions have been found with the help of heuristics. This process is inductive:

$$\text{Background knowledge} \cup \text{new knowledge} \models \text{evidence}$$

Therefore, heuristics can be seen as inductive inference rules.
ISP-DL theory motivates among others the following principles to implement computer systems to enable problem solving: As information is only helpful at impasse time, the system should offer help information only on demand. The help provided by the system should be adapted to the solution proposal of the learner and his pre-knowledge to avoid follow-up impasses. The learner should not be restricted to the typical solution examples.

1.2 IPSEs

Intelligent Problem Solving Environments (IPSEs) [16,17,18] - a special kind of Intelligent Tutoring Systems (ITS) - belong to the class of constructivistic learning environments.
They enable the student to solve problems in a determined domain. During this process he can get adaptive help. Unlike an ITS they do not contain teaching or learning models. The system offers a sequence/set of domain relevant tasks. The adaptive help is created by a generative expert system (GXPS), which is able to solve the tasks and analyse the students' proposals and is therefore able to respond adaptively to student hypotheses.
We describe briefly a typical IPSE and its formal specification.

(1) Problem Solving: $S \models E$ or $S \vdash E$ (subjective correct proposal)
(2.1) Possibly Incorrect Proposal: $T \neg\vdash E$ **(2.2) Incorrect Proposal:** $T \neg\models E$
(3) Stating Hypotheses: $E = E_{fix} \cup E_{mod}$
(4) Sound Completion Proposal: $T \vdash E'$ and $T \models E'$ with $E' = E_{fix} \cup E'_{mod}$ with **desirable but domain dependent monotony:** $T \models E_{fix}$ and: $T \models E'_{mod}$
(5) Self-Explanation: $S' \models E'$ or $S' \vdash E'$ with: $S = S_{fix} \cup S_{mod}$ and: $S' = S_{fix} \cup S'_{mod}$
(6) (Inductive) Knowledge Modification: $S \setminus S_{mod} \cup S'_{mod} \models E'$
(7) Used Symbols: S = subjective theory (personal knowledge), S_{fix} = proven knowledge, S_{mod} = modifiable knowledge T = specification of the task (=theory) E = solution proposal of the task, E′ = by the IPSE modified proposal E_{fix} = retaining part of the proposal, E_{mod} = modifiable part of the proposal

Table 1: IPSE-Specification

Table 1 illustrates the formal specification of an IPSE.

(1) The user has got a subjective theory S. According to this theory S he produces an artefact E either by intuition (|=) or by derivation (|-). Now the user asks the system about his proposal.
(2) The domain theory T is implemented in the expert system. If the solution proposal of the user is wrong, it cannot be produced or explained by this theory T. The system can give error feedback to the user's solution proposal. If the system's solution is generated by rules, it is possible, that the set of rules is incomplete. So there might be correct solutions that the system can't recognize.
(3) Now the user generates a hypothesis. He divides his proposal E into a part E_{fix} assuming that E_{fix} can be embedded in a correct solution and a part E_{mod}.
(4) If the system is able to generate a solution E' which contains E_{fix}, the hypothesis of the user is right. To explain the missing steps in the proposal of the user the solution part E'_{mod} can be used.
(5) According to the ISP-DL theory we expect some sort of self-explanation. To explain the correctness of the solution E' he has to generate the new knowledge S'_{mod}.
(6) Now the user modifies his knowledge inductively. This new knowledge S' contains old knowledge S_{fix} and new knowledge S'_{mod} and enables him to understand the proposal E'.

In the recent years we developed IPSEs in various knowledge domains. ABSYNT [23] (functional programming), PETRI-HELP (petri-net modeling), WI\KEA (room configuration for homely care), WULPUS [24] (management game), TAT [25] (constitutional formulas of reaction equations in organic chemistry), PULSE (pneumatic circuits), MSAFE (electrical control circuits), Patent-IT (assistance for patent proposers) [26].

2. Working Environment and Hypotheses Testing

The following part will give an overview of our working environment and of what hypotheses testing looks like in our IPSE.
The working environment (Figure 1, Figure 2) consists of several areas. In the task area (on the left side) the task settings and the information about circuit elements are shown; under Help, one can ask for the solution tree generated by the system (Figure 2, left). In the middle of the working area the circuit editor is located, where the student can work on the task circuit (set or change the properties of circuit elements, transform or simplify the circuit). The toolbar above contains the tools which can be used to manipulate the circuit. On the right side there is the worksheet area. This worksheet contains the solution proposal of the student. If he thinks the circuit he has edited belongs to the solution of the task, he can copy it into the worksheet. Also he can write formulas with a special formula editor implemented for the system and make textual annotations to his proposal. The system's feedback to hypotheses testing can be seen at the bottom of the window.
If the student loads a task, on the left the corresponding task specification is presented and possibly a circuit belonging to this task is shown in the middle. The worksheet just contains the initial state of the circuit. While working on the task, the student can edit this circuit and copy the different states of the circuit into the worksheet, he can write formulas and annotations into the worksheet and save it.
When he thinks that he has found a solution of the task (S |= E or S |- E), he may ask the system for an evaluation by marking his complete proposal even if the solution proposal is incomplete (Figure 1, on the right). The correctness of a hypothesis is proved by comparing

the student's solution with system generated solution (T |- E or T ¬|- E as our system is rule-based, see 2.1 Table 1). In case of an incorrect solution proposal the system gives error feedback (Figure 1, at the bottom). Now the user formulates a new hypothesis ($E = E_{fix} \cup E_{mod}$). He marks the parts of his proposal which he assumes to be correct (E_{fix}), that can be circuits as well as equations or both (Figure 2, right). The system gives positive feedback, only if E_{fix} can be embedded in a correct solution (Figure 2, at the bottom). In this case the solution trace of the program can be requested by the student (Figure 2, left). If the student lacks declarative knowledge, he can start the web-based learning modules of the partner universities. The appropriate links will be offered in the help tree.

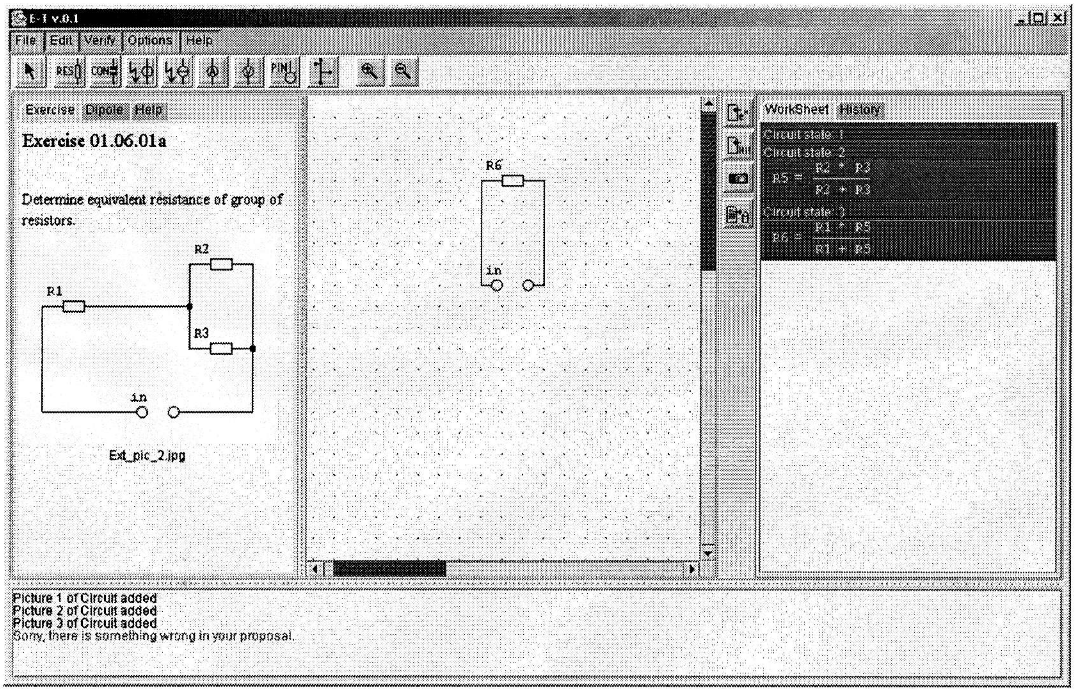

Figure 1. Incorrect Hypothesis

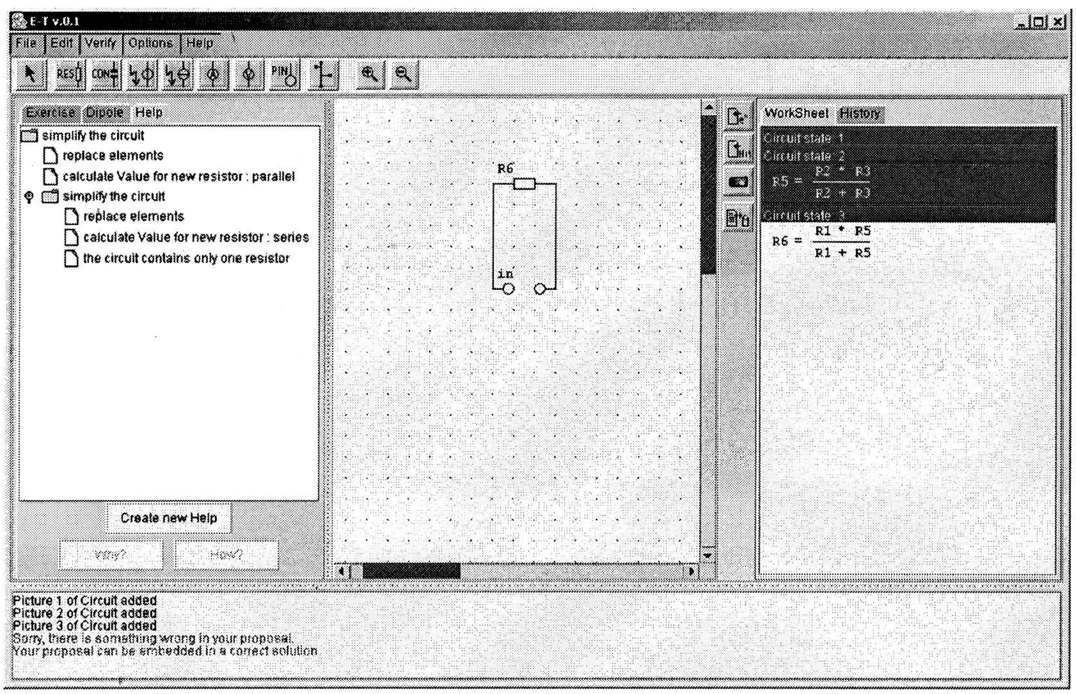

Figure 2. Embedable Hypothesis and Adaptive Help

The tasks will be saved in the WebTask-Database [27], which is established in Ilmenau and can be used online. Special XML-formats were elaborated to save the task specification and the user solution steps.

3. Knowledge Representation and Hypotheses Testing

3.1 Y-Model

Our detailed analysis of the task collections of our project partners brought us to the following elaborated Y-model for the task representation.

In our system supported tasks can be understood as relations over the sets of circuits (C), parameters (P), mathematical descriptions (M), and concepts (CF), see Figure 3.1.

The information about the circuit topology is stored in the C-quantity, symbolic and numeric parameter values - in the P-quantity, and mathematical equations - in M. According to a task goal several Y-Types of our tasks can be built. On the Figure 3.2-3.4. the initially given sets (P,C, or M) are light grey and the goal sets (P,C, or M) are grey. If in a task e.g. symbolic equations for given circuit are searched, this task belongs to the Y-Type 1 (Figure 3.2). Such tasks where some parameter for given circuits must be defined from other given parameters build the Y-type shown in the Figure 3.3. Tasks where any circuit must be created and some parameter of this circuit calculated are in Y-group shown in the Figure 3.4.

The quantities P, C, and M can be derived from each other by the application of CF. This derivation represents the task solving process of an expert and can be produced by our IPSE by means of gmr's.

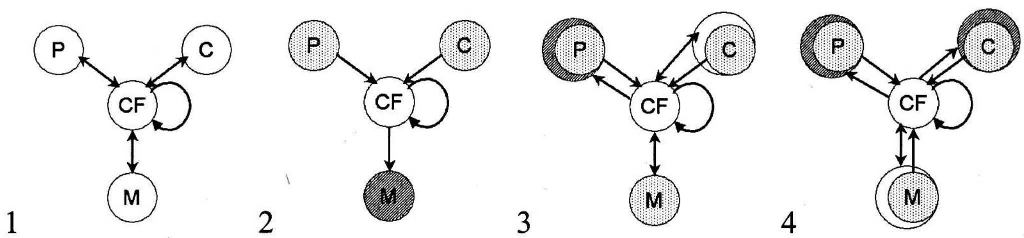

Figure 3. The Y-Model

3.2 Hypotheses testing with gmr's

To enable the testing of the hypotheses our IPSE contains a generative expert system which is able to solve the tasks. Therefore, the essential eletrical engineering knowledge has to be represented in the knowledge base of our system. For this purpose the goals-means-relations (gmr) are used. Gmr have been successfully implemented for varying domains: from functional programming (ABSYNT [19]) up to pilot modelling [28]. The gmr assigns the means (solutions) to the goals (tasks and subtasks). The means are dependent on different domains, for example they can be implementation parts, chemical elements (inclusive connections) or computations. In our IPSE the means are parts of the three sets circuits (C), parameters (P) and mathematical equations (M) of the former desribed Y-model. Additionally, there is a third parameter in the rules head. This is a informal text which describes the function of the rule. These texts can be used to generate help in the form of a trace of the solution steps of the program.

A special Meta-Interpreter interpreting the gmr-rules is implemented. Therefore, it is possible for the expert system to generate solutions for the tasks. In order to get many possible solutions the gmr-rules are very fine-grained. They can be combined to get different solutions, also "unusual" ones.
Owing to GMR the system will be able to examine hypotheses of the students and to complete solution drafts if necessary.
The example shows a gmr-rule for the calculation of an equivalent resistor. It is part of a gmr that simplifies the circuit. This task belongs to the Y-model shown in Figure 4 on the right side, because the circuit and some parameters are given and some parameters have to be calculated. First one has to find resistors of the given circuit that are connected according to the "ConnectionType" (e.g. parallel or series). The result of this search – the mean of this rule – is a part of the circuit (part of set C). As later on the resistance parameters of the resistors will be needed, the next step is to get them out of the set of parameters. Then the resistors are replaced through one new resistor, the mean of this rule is a part of the circuit. The last step is to calculate the new resistance parameter; the mean of this rule is a mathematical equation. In the case that the result of the equation is a numerical value, it also is a parameter.

```
gmr(simplify(_),_,['The circuit is simplified']):-
        gmr(searchOne(ConnectionType,resistor),(Elements,Points),_),
        gmr(getParameter(r,Elements),Parameters,_),
        gmr(replace(Elements,Points),[_,_,Points,[R,[Points,I],[Points,U],P]],_),
        gmr(calculate(ConnectionType,Parameters,R),Formula,_),
        gmr(simplify(_),_,_).
```

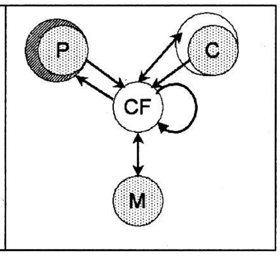

Figure 4. Gmr-rule example

4. Conclusion

Our learning environment is meant for the beginners, who do not have enough experience with the professional software packages such as CircuitMaker or Mathematica and who need support by solving the tasks. The problem solving environment is adapted to the tasks of the partners universities. It can be used however for the solution of different other tasks, as long as these tasks lie within the pre-defined topic fields. At the present tasks e.g. to such topics as basic electrical circuits, Kirchhoff's laws, equivalent circuits are supported. Further on, the working environment can be used task-independently. The student may use the system as an experimental work environment and explore any circuit from the knowledge domain implemented in the system.
Our learning environment can be used both in the direct study and for self learning, the explanations during the problem solving and the commentated solutions are helpful for exam preparations. A comprehensive testing and evaluation of our IPSE will take place at the partner universities after the first prototype is completed.

References

[1] www.tu-ilmenau.de/mile
[2] Iakimtchouk,V. Wagner, E., „Mathcad als Ingenieurwerkzeug in der multimedialen Lernumgebung" in 4. Workshop „Multimedia für Bildung und Wirtschaft" 28./29. 09. 2000, Ilmenau.

[3] Wagner, E., Hammer, S., Iakimtchouk, V., Aslanski, K., Learning Environment General Electrical Engineering Moduls. International Conference on Computer-Aided Engineering Education (CAEE'99) Sofia, Bulgaria 22.-24.09.1999
[4] SEIDEL, H.-U., WAGNER, E., Allgemeine Elektrotechnik Bd. 1, Carl Hanser Verlag München Wien, 3-446-21339-2, 1999
[5] MOEBUS, C., SCHROEDER, O. & THOLE, H.-J., Diagnosing and Evaluating the Acquisition Process of Programming Schemata, in J.E. Greer, G. McCalla (eds), Student Modelling: The Key to Individualized Knowledge-Based Instruction, Berlin: Springer (NATO ASI Series F: Computer and Systems Sciences, Vol. 125), 1994
[6] MOEBUS, C., SCHROEDER, O. & THOLE, H.-J., Interactive Support of Planning in a Functional, Visual Programming Language, in P. Brna, S. Ohlsen, H. Pain (eds), Proceedings AI-ED 93, World Conferences on Artificial Intelligence and Education, Edinburgh, 362-369, 1993
[7] MOEBUS, C., SCHROEDER, O. & THOLE, H.-J., Diagnosis of Intentions and Interactive Support of Planning in a Functional, Visual Programming Language, in D.M. Towne, T. de Jong, H. Spada (eds), Simulation-Based Experimental Learning, Berlin: Springer (NATO ASI Series F: Computer and Systems Sciences, Vol. 122), 61-76, 1993
[8] NEWELL, A., Unified Theories of Cognition, Cambridge, Mass.: Harvard University Press, 1990
[9] VAN LEHN, K., Toward a Theory of Impasse-Driven Learning, in H. Mandl et. al.(eds), Learning Issues for Intelligent Tutoring Systems, Berlin: Springer, 1988
[10] ANDERSON, J.R., Knowledge Compilation: The General Learning Mechanism. In: R.S. Michalski et. al., Machine Learning II. Kaufman, 1986
[11] ANDERSON, J.R., A Theory of the Origins of Human Knowledge, Artificial Intelligence, 1989
[12] HECKHAUSEN, H., Motivation und Handeln, Heidelberg: Springer, 1989
[13] GOLLWITZER, P.M., Action Phases and Mind-Sets, in: E.T. Higgins & R.M. Sorrentino (eds), Handbook of Motivation and Cognition, Vol. 2, 1990
[14] MOEBUS, C., SCHROEDER, O., The Acquisition of Functional Planning- and Programming Knowledge: Diagnosis, Modelling, and User-Adapted Help, in G. Strube, K.F. Wender (eds), The Cognitive Psychology of Knowledge. The German Wissenspsychologie Project (Advances in Psychology Series), Amsterdam: Elsevier (North-Holland), 233-261, 1993
[15] RUSSEL, St.J., The Use of Knowledge in Analogy and Induction, Research Notes in Artificial Intelligence, London: Pitman, 1989
[16] MOEBUS, C., Towards an Epistemology of Intelligent Problem Solving Environments: The Hypothesis Testing Approach, in J. Greer (ed), Artificial Intelligence in Education, Proceedings of AI-ED 95, Charlottesville: AACE, 1995
[17] MÖBUS, C., ALBERS, B., HARTMANN, St., THOLE, H.-J., ZURBORG, J., Towards a Specification of Distributed and Intelligent Web Based Training Systems , in: St.A. Cerri, G. Gouarderes & F. Paraguacu (eds), Intelligent Tutoring Systems, Proceedings of the 6th International Conference, ITS2002, Biarritz, France and San Sebastian, Spain, June, 2002, S. 291 - 300, Berlin: Springer, Lecture Notes in Computer Science, LNCS 2363, ISBN 3-540-43750-9
[18] MÖBUS, C., ALBERS, B., GARBE, H., HARTMANN, St., THOLE, H.-J., YAKIMCHUK, V., ZURBORG, J., Towards an AI-Specification of Intelligent Distributed Learning Environments , AI Journal Issue 1/03 "Knowledge Modeling and Knowledge Communication in Learning Scenarios" (to appear)
[19] MÖBUS, C., THOLE, H.-J., Interactive Support for Planning Visual Programs in the Problem Solving Monitor ABSYNT: Giving Feedback to User Hypotheses on the Basis of a Goals-Means-Relation, in: D.H. NORRIE, H.-W. SIX (eds), Computer Assisted Learning. Proceedings of the 3rd International Conference on Computer-Assisted Learning ICCAL 90, Hagen, F.R.Germany, Lecture Notes in Computer Science, Vol. 438, Heidelberg: Springer, 1990, S. 36-49.
[20] MÖBUS, C., SCHRÖDER, O., THOLE, H.-J., WULPUS - An Intelligent Problem Solving Environment Delivering Knowledge Based Help and Explanations in Business Management Simulation, in C. Frasson, G. Gauthier, A. Lesgold (eds), Intelligent Tutoring Systems, Proceedings of the Third International Conference ITS 96, Montreal, June 1996, Berlin: Springer (LNCS 1086), 1996
[21] THOLE, H.-J., MÖBUS, C., SCHRÖDER, O., Domain Knowledge Structure, Knowledge Representation and Hypotheses Testing, in: B. Boulay, R. Mizoguchi (eds.): Artificial Intelligence in Education Proceedings of AI-ED 97 World Conference on Artificial Intelligence in Education, Kobe, Japan, Amsterdam: IOS Press, 1997
[22] WILLMS, J., Konzeption einer intelligenten Problemlöseumgebung für die Patentanmeldung und –prüfung, Dissertation, Universität Oldenburg, Fachbereich Informatik, 2002, http://www.dissertation.de.

[23] MÖBUS, C., THOLE, H.-J., Interactive Support for Planning Visual Programs in the Problem Solving Monitor ABSYNT: Giving Feedback to User Hypotheses on the Basis of a Goals-Means-Relation, in: D.H. NORRIE, H.-W. SIX (eds), Computer Assisted Learning. Proceedings of the 3rd International Conference on Computer-Assisted Learning ICCAL 90, Hagen, F.R.Germany, Lecture Notes in Computer Science, Vol. 438, Heidelberg: Springer, 1990, S. 36-49.

[24] MÖBUS, C., SCHRÖDER, O., THOLE, H.-J., WULPUS - An Intelligent Problem Solving Environment Delivering Knowledge Based Help and Explanations in Business Management Simulation, in C. Frasson, G. Gauthier, A. Lesgold (eds), Intelligent Tutoring Systems, Proceedings of the Third International Conference ITS 96, Montreal, June 1996, Berlin: Springer (LNCS 1086), 1996

[25] THOLE, H.-J., MÖBUS, C., SCHRÖDER, O., Domain Knowledge Structure, Knowledge Representation and Hypotheses Testing, in: B. Boulay, R. Mizoguchi (eds.): Artificial Intelligence in Education Proceedings of AI-ED 97 World Conference on Artificial Intelligence in Education, Kobe, Japan, Amsterdam: IOS Press, 1997

[26] WILLMS, J., Konzeption einer intelligenten Problemlöseumgebung für die Patentanmeldung und –prüfung, Dissertation, Universität Oldenburg, Fachbereich Informatik, 2002, http://www.dissertation.de.

[27] www.tu-ilmenau.de/GETsoft

[28] LÜDTKE, A., MÖBUS, C., THOLE, H.-J., Cognitive Modelling Approach to Diagnose Over-Simplification in Simulation-Based Training, , in: St.A. Cerri, G. Gouarderes & F. Paraguacu (eds), Intelligent Tutoring Systems, Proceedings of the 6th International Conference, ITS2002, Biarritz, France and San Sebastian, Spain, June, 2002, S. 496 - 506, Berlin: Springer, Lecture Notes in Computer Science, LNCS 2363, ISBN 3-540-43750-9