ISBN: 978-989-8533-55-5 © 2016

FRAMEWORK FOR INTELLIGENT TEACHING AND TRAINING SYSTEMS – A STUDY OF SYSTEMS

M.Sc. Nikolaj Troels Graf von Malotky and Prof. Dr.-Ing. Alke Martens
University of Rostock, Institute of Computer Science
Albert-Einstein-Str. 22, D-18059 Rostock, Germany

ABSTRACT

Intelligent Tutoring System are state of the art in eLearning since the late 1980s. The earliest system have been developed in teams of psychologists and computer scientists, with the goal to investigate learning processes and, later on with the goal to intelligently support teaching and training with computers. Over the years, the eLearning hype appeared and vanished, and returned in a slightly different shape. However, Intelligent Tutoring Systems, as one of the most adaptive system types, remained in the focus of interest. Due to this, it is surprising that there exists no ITS architecture, no guideline how to implement these systems, and no structured, software engineered approach, which helps to develop comparable systems. In this paper, we will show the first steps toward such a system: a broad analysis of ITS over several years of research, with the resulting structured architecture.

KEYWORDS

Intelligent Tutoring System, Architecture, Framework

1. INTRODUCTION

Intelligent Tutoring systems (ITSs) are a special type of eLearning or teaching and training systems. The history of ITSs can be traced back to the early 1980s. One of the first ITSs is based on the medical expert system MYCIN, which has been developed between 1972 and 1980. The resulting ITS has been called GUDION (Clancey 1987). Expert systems were the main part of ITS. They integrated the idea of explicating the experts' knowledge, for example based on facts and rules, and sometimes extended by methods of artificial intelligence (AI), for example Bayes' Networks, in the system core. The expert knowledge was used as reference for the learner's learning progress and as basis for correcting the learner's actions. Using MYCIN and the early versions of GUIDON in medical education, it quickly became clear, that these systems can only support learning of finding the correct diagnosis, but not training of the diagnostic reasoning process. In the process of developing GUIDON, the pure expert knowledge base was extended by so-called pedagogical knowledge. This pedagogical knowledge should optimally add a teacher's knowledge regarding instructional design and supporting the learner to the existing expert knowledge. A bit later, as computer technology became quicker and storage became cheaper, learner models have been added (Lelouche 1999). Learner models allowed to generate an appropriate analysis of individual progress and recording of learner's activities. From the ITS architecture's perspective, a new component was added. Some extra ways have been investigated, for example working with bug libraries in BUGGY in the 1981, simulation in SOPHIE 1982 (an overview is given in (Nwana, 1990)). In the 1989, the growing of the commercial internet starts. The ITS quickly became web-based applications. Architectures in ITS started with client-server designs, and then moved to web-based applications, with online teaching and training material. From the system developers' and the content developers' perspective, the internet-based approach facilitates update of teaching and training material. In the years after the advent of the internet, ITS became mobile, interactive, game-based and ubiquitous pervasive.

All of these aspects have been reflected in changes of the ITS architecture in one way or the other. Whereas in the first years, the ITS architecture has been extended by integrating new components (e.g. from GUIDON to GUIDON2 (Clancey, 1988)), or completely redesigned based on a new trend (e.g. agent-based design), in the last years, the core architecture of ITS has not really been part of investigations. This is contrasted by the idea of software engineering, which itself has a long history in computer science. Starting in the 1960, and becoming a traceable branch of research in the 1980s, software engineering came with the claim that computer software design shall be an engineering task and not "art" (Gamme et.al., 2015). Due to this perspective, several authors have tried to analyze and construct the ITS architecture. However, up to date, most of the ITS developments start off from scratch. Even if ITSs are one of the oldest systems in the field of E-Learning, there exists no still valid framework, which is domain independent, and independent of the programming language.

Thus, our idea was to analyze existing ITSs, to investigate published frameworks and architecture for ITS development, and based on the insights of these steps to develop a generic framework for ITS design. The paper is structured as follows: in the next section, we will show some aspects of our analysis, first from the perspective of existing systems, second from the perspective of architectures and existing frameworks. In the third section, we will sketch our resulting approach for generic E-Learning Software design. The paper closes with a conclusion and outlook.

2. ITS ARCHITECTURE

The main perspective of the ITS architecture development has been the definition of interacting components. Over the years, the set of components has been extended. The name of the components vary, as can be seen in table 1, but the main functionality remains the same. As postulated in different papers, e.g. in (Harrer et.al., 2007) (Ruddeck et.al., 2010), the core architecture of ITSs consists of the following components:

- The expert knowledge base, which is the oldest component. This component is based on the idea that the mediated knowledge is (at least a part of) the knowledge of an expert of the application domain, e.g. medicine. The modeling of the domain knowledge is usually taking place in a combination of facts and rules, sometimes extended by AI approaches like Bayes' Networks. The main effort in developing an ITS is usually the modeling of the domain knowledge.
- The pedagogical knowledge model is the component, which is the most blurry in the set of components. Sometimes, e.g. in case-based training, the pedagogical knowledge consists of the case structure and the context dependent knowledge in contrast to the overall knowledge of the domain experts. Sometimes, the pedagogical knowledge model is realized as instructional facts and rules.
- The learner model is one of the younger components in the set. Nonetheless, the learner model reached an importance that has led to a whole branch of research, called user modeling. The components abilities range from simple recording and tracking of activities up to complicated guidance and help structures.
- The user interfaces also is related to an own branch of research, which is based in Human Computer Interaction (HCI). Moreover, we have found that the functionality of the user interface ranges from steering of the whole adaptation process up to pure showing of next possible activities.

One of the first descriptions of the architecture has been developed by Clancey (1984, 1987), who has defined the above mentioned components as part of an ITS. However, even if he has described the components, he has neither focused on the role and functionality of the components, nor on the components interaction. Thus, up to date, no reliable description of interaction of components, of tasks and of role and functionality is available. Literature review has revealed, that the above mentioned components can be treated as the common denominator in ITS (an extract of the analysis is shown in table 1). This insight has for example been formulated in form of an architecture by Lelouche (1999).

ISBN: 978-989-8533-55-5 © 2016

Table 1. Table of naming parts of the ITS (Martens,Harrer, 2006)

Reference	Expert Knowledge	Pedagogical Knowledge	Learner Element	User Interface
(Corbett et, all, 1997)	Domain Knowledge	Peagogical Module	Student Module	Poblem Solving Environment
(Lelouche, 1999)	Domain Expert Module	Tutorial Module (Pedagogy Expert)	User Model	User Interface
(Alpert et.al., 1999)	Expert Solver	Tutorial Module	Student Module	User Interface
(Melis et.al., 2004)	MBase (Course Generator)	Pedagogical Rules (Course Generator)	Student Model	Presentation Engine

However, one drawback has been found in the component based design: whereas the basic functionality and the structure of the components were more or less clear, the communication between the components and the role and functionality of the combined components were not. This can be seen in the following figure 1.

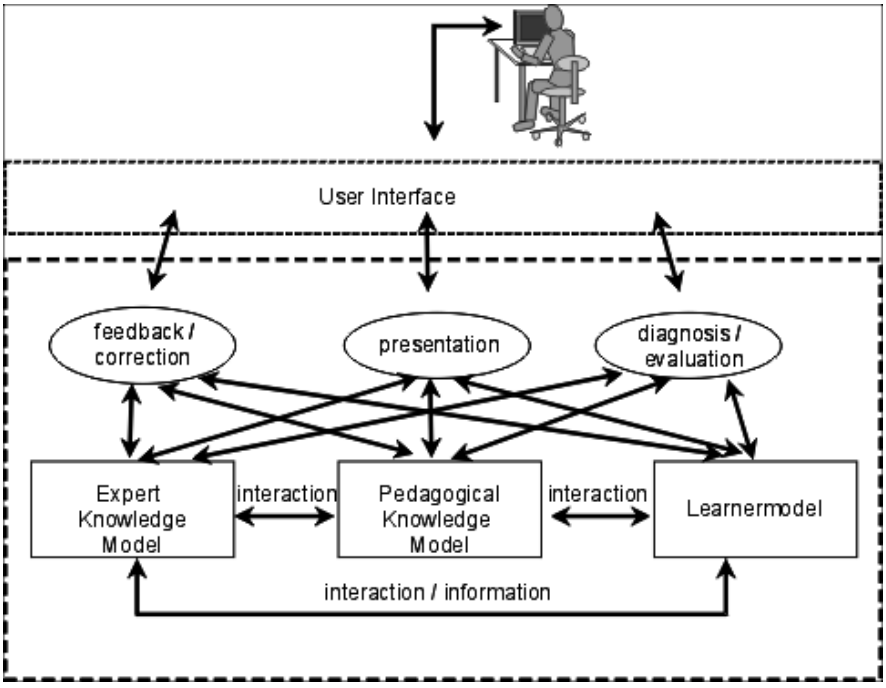

Figure 1. Abstract scheme of the classical IST architecture (Martens, 2003)

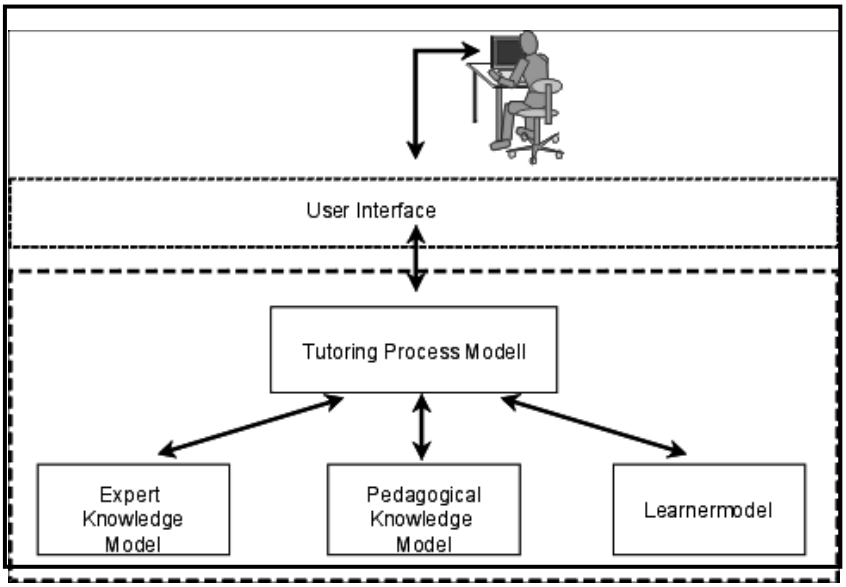

Figure 2. Architecture of ITS (Martens, 2003)

Thus, Martens et. al. chose to focus on the communication between the components. This has led to a design, where a central process steering component, the tutoring process model, takes over the adaptation functionality, whereas the tasks of the other components have been reduced to mere database functionality. This is sketched in figure 2. Harrer and Martens (2006) developed a pattern catalogue for ITS, based on the idea of centralized or externalized tutoring processes, where the role and functionality is described in detail. This pattern catalogue helped to develop ITS from scratch, however, even this approach did not help to design the communication between the components, which became clear, when we started with the effort to use our first framework as basis for implementation in 2001. So our next step in the last two years has been a deep analysis of system types, coming from the perspective of the ITS, but after all including all types of eLearning systems. We have learned, that we should better start with interaction scenarios. So we used task analysis and focused on the different modi of usage and interaction. The result has led to the framework, which we sketch in the following section.

3. ITS FRAMEWORK

As a first step, we analyzed the combinations in which students, teachers, experts, authors could interact with an ITS. The ITS has to be the only teacher in the knowledge transfer of the expert knowledge to the student. The suitable combinations showed us that there are multiple use cases for ITS aside from their capabilities as a teacher. The properties of the cases could then be split in two categories, these properties do sometimes have dependent other properties (sketched in figure 3).

The first category are properties which are the same for many learning programs and are more general: “Authoring” with the ability of automatic content creation, “Online connection” with the ability to send usage statistics, “Multiple users” and the ability to allow collaboration between them and at last a human “Supervisor” and the ability of live interaction with the student(s). “Online connection” is the property that the ITS or components of it can be accessed through the internet, which can be more up to date as a local one. Since one of the aims of the generic ITS framework is their evaluation and comparison, its usage should be accessible without help from the student itself, in a non-research environment. This is called “Send statistics” and can be met through sending the usage data over the internet back to the people which need it for scientific research or for review of the teaching content and is dependent on online connection functionality. “Authoring” means that there are authoring tools available to make it easier to experts and teachers to change and manage the teaching content, so that authoring is possible without knowledge of the internals of the ITS. Because of the always improving artificial intelligence automatic content generation is possible for selected domains and this will expand in the future, this is named “Content generation” which is

dependent on the “Authoring” functionality. “Multiple Users” defines that there can be more than one student who are taught by the same ITS instance, so it can manage multiple users and differentiate them and their progresses. The “Collaboration” property describes that multiple users can interact and learn with each other through the ITS and therefore depends on the “Multiple Users” property. The ITS should teach, so if there is need for a human who is not a student, he should only have the role of a “Supervisor”, a person to control the situation and help to interact with the ITS, he would not be the teacher, this could allow him to supervise more students than he could in the role of a teacher, it could be in a form of a forum or “Live” interaction through chats or live stream. Even if the supervisor is in the same room as the students, it could be sensible to still implement such a functionality to give the supervisor a more powerful user role, which in turn would allow him to interact with multiple students more efficiently than only with the physical presence.

Since the first category already specified if there are multiple users or an internet connection, we only need to consider the simplest use case for the second category. The second category of properties describes the inner functionality of a local, single user ITS without online capabilities or a supervisor. The components would be a good fit, but they are not defined precisely in the traditional ITS architecture. In research the usage of the naming and the functionality of the components is varying. The resulting architectures from the created ITSs have different capabilities and responsibilities for the components. So the resulting architectures descriptions are not compatible among themselves and therefore hard to compare. To analyze what does specify the inner functionality of the ITS we did not split it up into components of the general ITS system architecture, but extracted which functions the components fulfill. After analyzing the functionality of the components of the ITS architectures, we could find seven main functionalities which ITSs use, it is the core of the ITS. Two of them are obligatory to create a minimal tutoring system: “Deliver” and “Evaluate”. Expert information is not saved in a way which is easily understandable for the student. “Deliver” will construct an easily to access and view content unit from a given query, either from passive lectures or interactive exercises. “Evaluate” is a correctness check of the student's answer which saves the judgement into the student's progress. The ITS can then answer with the “Feedback” functionality to react accordingly to the student's answer and give appropriate response, adaptive to the student. This intelligence can also be used as a guidance for proactive or user activated hints, but since the functionality is different and not dependent from exercises it is excluded in the “Help” functionality. “Estimate” generates a profile from the student's collected activities, which can be saved or used immediately. Like in the case of the functionality “Propose”, it will return content matching the knowledge of the student. At last is “Summarize” as a functionality for completely analyzing the behavior of the student, to show the strengths and weaknesses for different categories, like day of week, knowledge subdomain and so on.

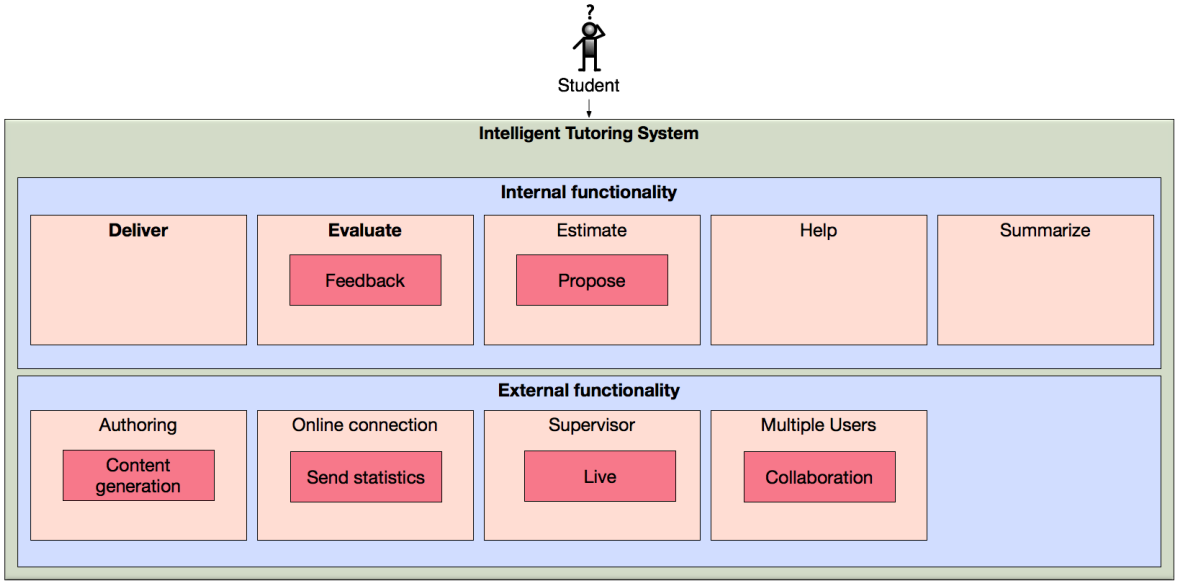

Figure 3. Suggested Architecture of ITS

After we have identified the tasks an ITS can and has to do, we could define the possible combinations of functional components for the internal and external functionalities. This allowed us to list all types of ITS and parameterize and therefore categorize an ITS (see figure 4). The system architecture uses five layers to split the business logic from database and user interface responsibilities. The business logic is what the core ITS functionality is, all the internal functionality is there. The databases have to be created accordingly to the functionality that was defined earlier, but instead of defining the concrete database, we instead use delegates to easily access information matching to one responsibility. The expert domain knowledge is what has to be transferred to the student and is split up into the passive and interactive content, named "Lectures" and "Training content", but also rules in the "Domain rule base" from the domain, formatted in a way so that the ITS can apply them. The "Student's progress" contains all the relevant steps the student did, so that the ITS can track what the student did relevant to learning. The ITS has to react to the students interactions with the learning content, to do that, errors should be easily recognized and the "Bug analysis" helps with that. Not all these database controls are necessary to create an ITS, you can for example just have "Lectures", "Reaction of Tutor" and "Student's progess", but it for a feature rich ITS it is mostly recommended to add all of them. These annotated categories are the 4 components often used in ITS: User interface, Learner model, Pedagogical knowledge and Expert knowledge. The user interface can vary drastic from one ITS to another. Still there are some separations of the user interface which makes sense in many cases. Given here is the example separation into a Statistics, Select Lecture, Learn and Train scene.

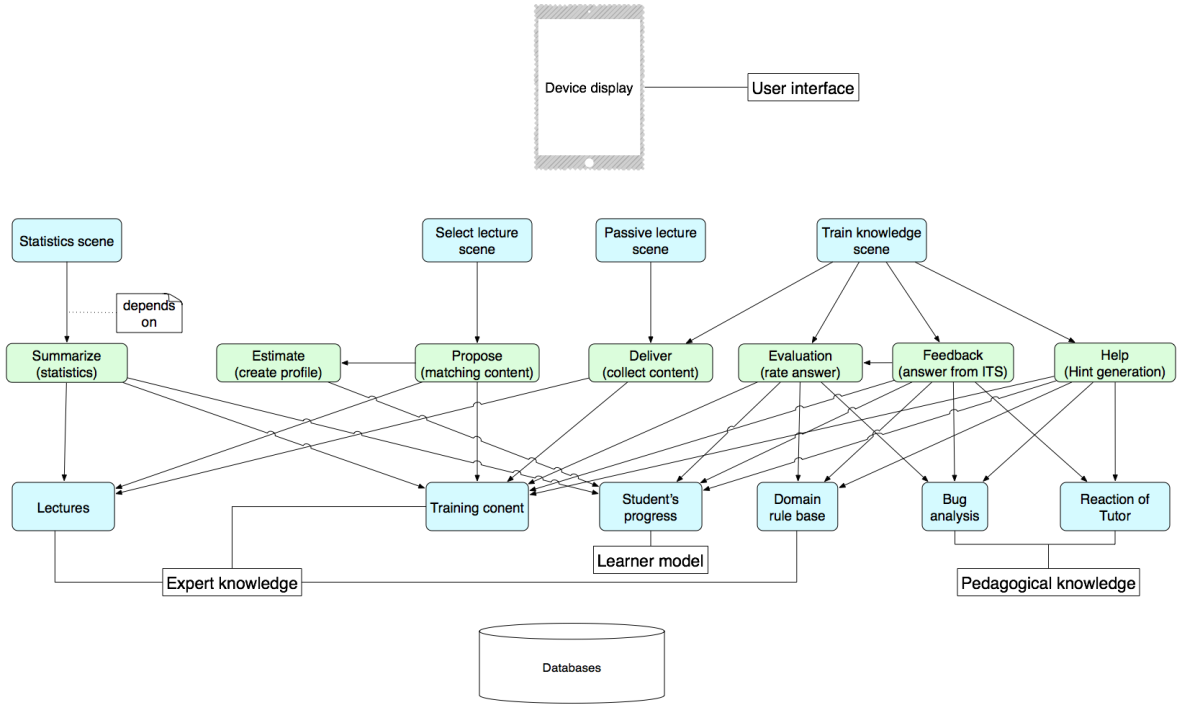

Figure 4. Dependencies of the functionalities

So these are the different components of an ITS and they are dependent from each other as shown. Instead of allowing any communication between them, the components can only communicate if they have a dependency relationship, the different parts of functionality are then eventually connected through the user interface.

4. CONCLUSION AND OUTLOOK

In our way to develop a framework of ITS, we started several years ago with a classical, component based approach. Over the years of research, we must learn that this approach was not deep enough to help in designing modern approaches in ITS. Thus, we focused on another aspect of ITS in particular, and eLearning in general, which is the role and functionality of the system in the interaction process with the learner. As a result, we developed a completely different approach to system design, which allows on the system kernel to realized the de facto existing portfolio of functionality. To cross-check our ideas, we developed a matrix of use-case scenarios, which has led us to the insight that our approach is flexible enough to allow the modeling and design of a plethora of different system types. Thus, as a next step, we go one step further and make a prototypical implementation of the concepts, and try to use this for a set of different ITS prototypes.

REFERENCES

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*, 10 (2), pp 183- 197

Clancey, W. J., 1984, Methodology for Building an Intelligent Tutoring System. In: Kintsch, W., Miller, J.R., and Polson, P.G. (eds.): Methods and Tactics in Cognitive Sciences, Lawrence Erlbaum Associates, Hillsdale, New Jersey, London, pp 51- 84

Clancey, W. J., 1987, Methodology for Building an Intelligent Tutoring System. *In: Kearsley (ed.): Artificial Intelligence and Instruction. Addison Wesley, MA.*

Corbett, A.T., Koedinger, K.R., Anderson, J.R. 1997, Intelligent Tutoring Systems. *In: M. Helander, T.K. Landauer, P. Prabuh (Eds.): Handbook of Human-Computer Interaction. Elsevier Science B.V., 2nd completely revised edition. Pp 849—874*

Gamma, E., Helm, R,, Johnson, R, Vlissides, J. 2015, Design Patterns. Elements of Resuable Obejct Oriented Software. *Pearson Education, US. 2nd revised edition.*

Harrer, A., Martens, A., 2007, An Integrative Approach for Teaching/Tutoring Process Models Using Meta-Models. *International Conference on Cognition and Exploratory Learning in Digital Age, CELDA 2007, Algarve, Portugal*.

Harrer, A., Martens, A., 2006. Towards a Pattern Language for Intelligent Teaching and Training Systems. *8th International Conference on Intelligent Tutoring Systems, ITS 2006, Chungli, Taiwan.*

Harrer, A., Martens, A. 2010, A heterogeneous pattern language for collaborative learning systems and intelligent tutoring systems. *in: P. Goodyear, S. Retalis (eds.)Technology enhanced learning: Design Patterns and Pattern Languages, pp. 153 –166, chap. 9, Sense Publishers, Rotterdam (ISBN: 978-94-6091-060-9).*

Lelouche, R. 1999, Intelligent Tutoring Systems from Birth to Now. *In: KI-Künstliche Intelligenz,* 4 ,pp 5-11.

Martens, A., 2003, Centralize the Tutoring Process in Intelligent Tutoring Systems. *5th International Conference on New Educational Environments, ICNEE 2003 Luzern, Schweiz*.

Melis, E., Siekmann, J. , 2004. Activemath: An Intelligent Tutoring System for Mathematics. In R. Tadeusiewicz, L.A. Zadeh, L. Rutkowski, J. Siekmann, (Eds.). *7th International Conference 'Artificial Intelligence and Soft Computing' (ICAISC) Lecture Notes in AI LNAI 3070. Springer-Verlag pp 91- 101*

Nwana, H.S. 1990, Intelligent Tutoring Systems: an overview, *Artificial Intelligence Review, 4, pp 251-277.*

Ruddeck, G., Martens, A., Maciuszek, D., Weicht, M. , 2011 Communication Patterns in Component-Based Intelligent Tutoring Systems. in: *The European Journal for Informatics Professional, Special Issue on "Engineering in eLearning Systems"*.