User Model Integration in a Distributed Adaptive E-Learning System

Peter Brusilovsky[1], Sergey Sosnovsky[1], Michael Yudelson[1],
Amruth Kumar[2], Sharon Hsiao[1]

[1]University of Pittsburgh, School of Information Sciences,
135, N. Bellefield ave., Pittsburgh, PA, 15260, USA
[2]Ramapo College of New Jersey
505, Ramapo Valley Road, Mahwah, NJ 07430-1680
{peterb, sas15, mvy3}@pitt.edu,
amruth@ramapo.edu, hyl12@pitt.edu
http://www.sis.pitt.edu/~paws/

Abstract. The integration of adaptive educational systems is changing from an interesting research problem into an important practical task. One of the major challenges that need to be accepted on the way is the development of mechanisms for student model integration. In this paper we propose an approach for aligning overlay models of students' knowledge collected by different educational tools relying on different domain representations. The approach is based on common protocols for evidence communication and manual mapping of underlying domain models. This approach has been applied for integration of two existing educational systems in the framework of undergraduate Java course.

1 Introduction

Over the last 10 years, a range of adaptive systems migrated from research labs to real life. Web recommender systems [1], mobile tourist guides [2], and adaptive e-learning systems [3] are now used by hundreds of real users. In some application areas the "density" of practical adaptive systems is reaching the point where several adaptive systems are available. Yet, in most of the cases, these systems do not compete, but rather complement each other offering unique functionality or content. This puts the problem of using several adaptive systems in parallel on the agenda of user modeling community. This problem has been explored over the last few years by several research teams, which considered it from several prospects: cross-system personalization [4], user model ontologies [5], and user modeling servers [6].

The main challenge of using several adaptive systems in parallel (or, as we may also say, a distributed adaptive system) is making the whole more then the sum of its parts. In this context, it means that each of the system should have a chance to increase the quality of user modeling and adaptation using integrated evidence about user, which was collected by all participating system. So far, it looks like the most popular approach to solving this user model integration problem is translation [7] or

mediation [8] from one user model to another. However, in a number of cases translation of the whole user model from one representation to another can be a relatively costly approach. For example, if two adaptive systems are used not in a sequence, but in parallel, the user models on both sides are being constantly updated. Thus, to take the joint information about the user into account, each of the system will need to translate the user model from another system before every adaptive decision is made.

A very good example of this situation is a distributed adaptive E-Learning system. In such a system, a student can work with educational activities provided by several systems. Each of the involved systems receives evidences about user knowledge and attempts to build the student knowledge model. To make this model reliable, each of the involved systems should take into account evidences about the student produced during her work with all systems. However, as our experience of work with distributed system shows, a student can switch from one system to another many times even within a single session [9]. To avoid multiple translation from one user model to another within the same session, we explored an alternative approach to user modeling in distributed adaptive systems, which we call evidence integration. With this approach, adaptive systems do not exchange user models, but instead exchange evidences received while working with the same student. In this case, the problem of user model integration becomes the problem of evidence integration. While evidence integration is a relatively simple task in some domains (i.e., user's ratings for a specific movie can be easily taken into account by multiple recommender system), it is not the case in E-Learning. In E-Learning, each educational activity (i.e., problem, quiz, or example) is typically described in the terms of a system's internal *domain model*. Using this knowledge and information of student success or failure while working with the activity, the user modeling component updates student knowledge model. In a rare case where the component systems share the same domain model [10] integrating evidences from two or more adaptive systems is a relatively simple problem. However, in reality two adaptive systems developed for the same domain (such as Java programming of SQL) have very different domain model. In this case evidence integration become a real problem, which requires some kind of translation from one *domain model* to another.

This paper presents a practical example of distributed user modeling, which involves two e-learning systems with considerably different domain models for Java programming language. The goal of our work is to stress the problems of distributed user modeling in the field of E-Learning and provide an example of conceptual and architectural integration, which was used in a real college-level course.

The rest of this paper is structured as follows. Section 2 describes the details of two educational systems for Java programming domain developed by different research teams and relying on different domain models: Java-Problets and QuizJET. Section 3 discusses the problems we had to overcome while integrating these two systems on the level of their domain and user models. Section 4 presents the implementation details of the integration mechanism. Section 5 finishes the paper with discussion and plans of future work.

2 QuizJet vs Problets: Two Systems and Two Domain Models

2.1 Ramapo College's Problets

Problets (www.problets.org) are problem-solving tutors on introductory programming concepts in C/C++/Java/C#. They present programming problems, grade the student's answer, and provide corrective feedback. Problets sequence problems adaptively [11], and generate feedback messages that include the step-by-step explanation of the correct solution [12]. Students can use Problets for knowledge assessment and self-assessment, as well as for training problem-solving skills. The reified user interface of Problets emphasizes the use of mental models students need to maintain, when solving the problems [13]. Fig. 1 presents the student interface of a Problet on if/if-else Statements in Java. The bottom-left panel contains the simple Java program. The students need to evaluate the program and answer a question specified in the top-left panel. The system presents student's answers in the right-bottom panel, and indicates the correct and incorrect answers by marking them in green, and red correspondingly. The detailed help on how to use the system, submit the answers and read system's feedback messages can be always opened in the right-top panel of the Problet interface.

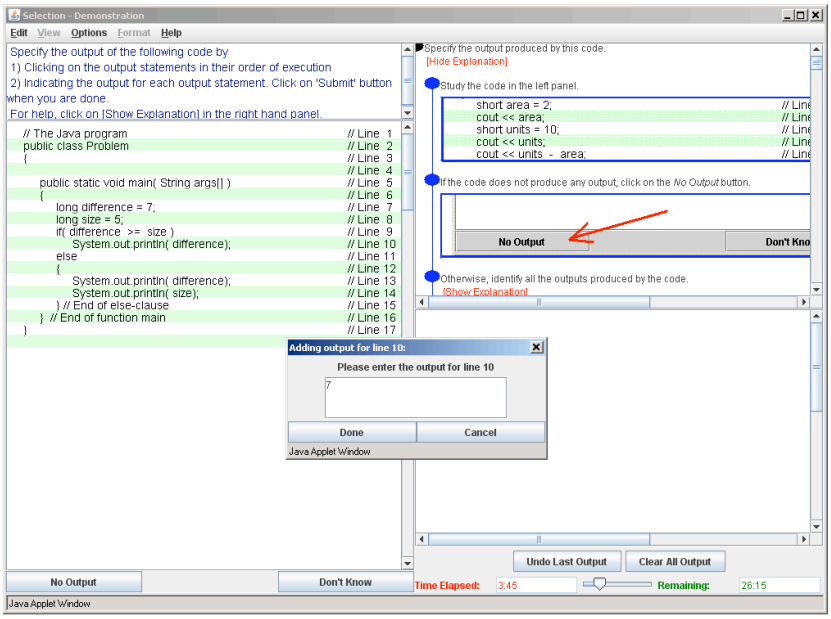

Fig. 1. A Problet on if/if-else Statements in Java.

Problets rely on the concept map of the domain, enhanced with pedagogical concepts called learning objectives, as the overlay student model [14]. Each learning objective is associated with the proficiency level calculated based on the student's

answers. The student model provides the basis for adaptive decisions made by the tutor, by associating a proficiency model with each learning objective. The system propagates the proficiency values to the top levels of the concept hierarchy. At any moment of the tutoring session, a student can observe the current state of her/his user model. Fig. 2 demonstrates an example of the user model snapshot for the if/if-else Statements in Java.

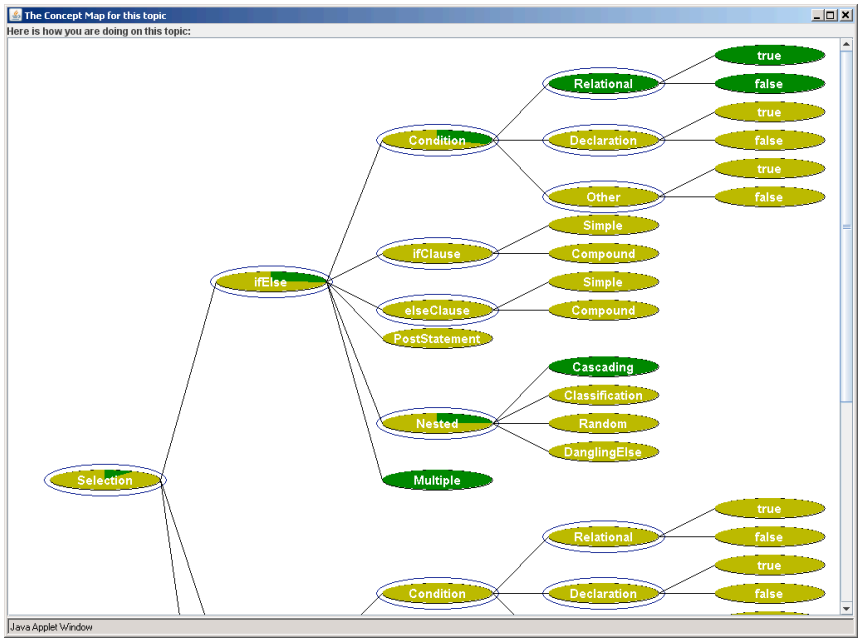

Fig. 2: A part of domain hierarchy on if/if-else Statements in Java. Learning objectives are associated with each concept in the hierarchy.

2.2 University of Pittsburgh's QuizJet

QuizJET (Java Evaluation Toolkit) is an online quiz system for Java programming language. It provides authoring and delivery of quiz questions and automatic evaluation of students' answers. A typical question in QuizJET is a simple Java program. The students need to evaluate the program code and answer a follow-up question, which can take one of two forms: "What will be final value of the marked variable?" or "What will be printed by the program to the console window?". Upon evaluation of the student's answer QuizJET provides short feedback specifying the correctness of the answer and the right answer in the case a student has make a mistake.

Fig. 3 demonstrates the student interface of QuizJET. The Java programs constituting QuizJET questions can contain one or several classes. For switching between classes QuizJet implements tab-based navigation. The driver class containing

the main function (the entry point to the program) is always placed in the first tab, which also presents the question itself, processes the student's input and presents the system's feedback.

The important feature of QuizJet is parameterized questions. One or more numbers in the code of a driver class are dynamically replaced with a random value every time the question is delivered to a student. As a result, the students can practice QuizJET questions multiple times, and every time the question will be different and have a different correct answer.

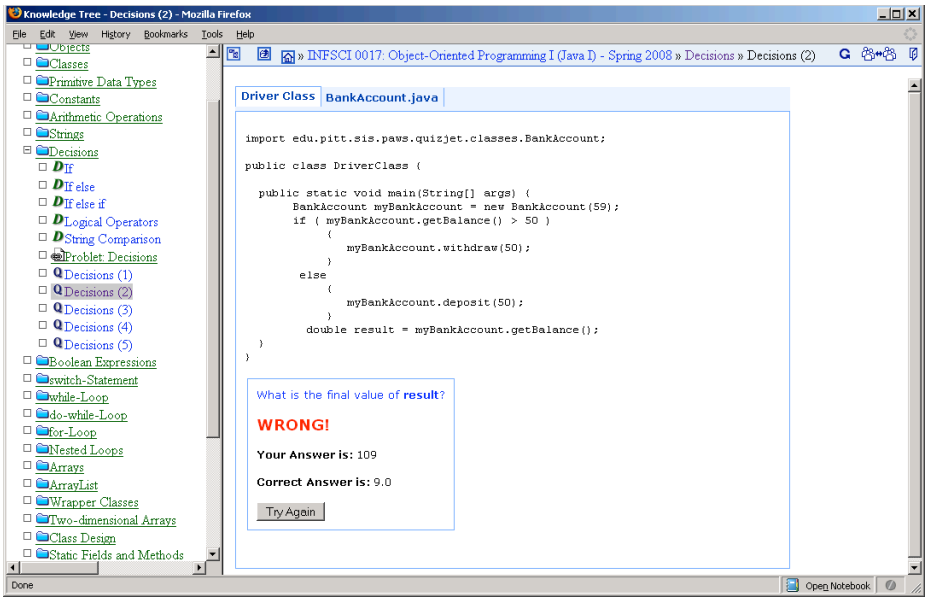

Fig. 3: An example of QuizJet question on Decisions in Java accessed through the Knowledge Tree Learning Portal.

Every QuizJET question is indexed by a number of concepts from the Java ontology. A concept in a question can play one of two roles: it acts either as a prerequisite for a question (if it is introduced earlier in the course), or as a question outcome (if the concept is first introduced by this question). Fig. 4 presents an extract from the Java ontology.

3 Integration: The Conceptual Side

Both Problets and QuizJet questions rely on conceptual content models that provide detailed representation of underlying domain knowledge. In order to provide consistent interpretation of the evidence reported by these two types of learning content, perform unifying user modeling and implement adaptive mechanisms taking

into account student's work with both systems we need to integrate the underlying domain models on the level of concepts constituting them.

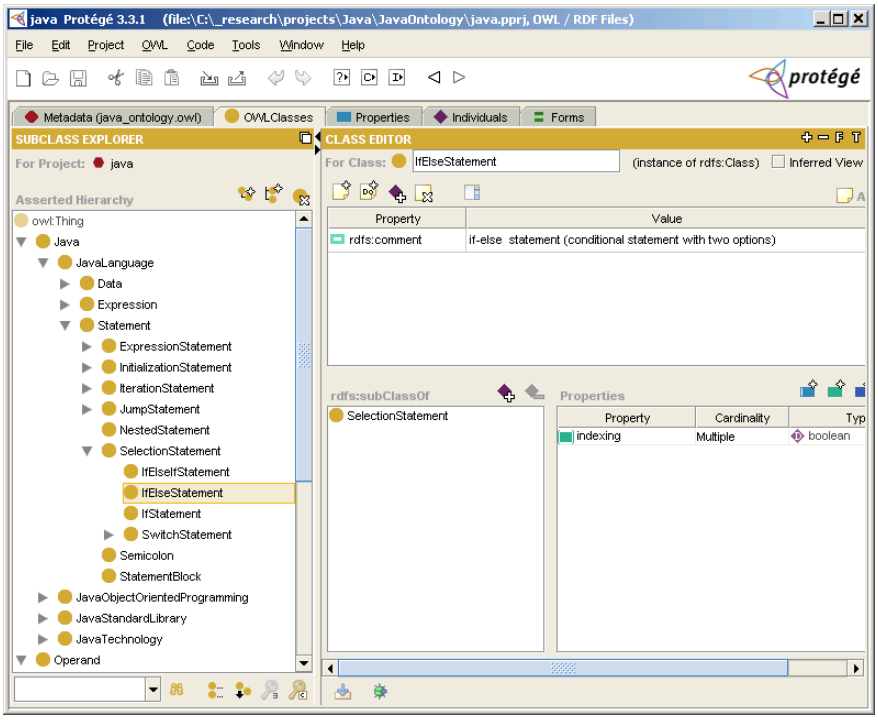

Fig. 4: An extract from the Java ontology.

The general task of domain model alignment potentially involves resolution of multiple model discrepancies on the two principle levels. The language–level mismatches, such as different syntax, expressiveness, or varying semantics of used primitives, need to be resolved in the first place. However, the more important are the model-level mismatches that occur due to the difference in structure and/or semantics of the domain models. Resolution of this kind of discrepancies involves dealing with such problems as:

- Naming conflicts (the same concept is defined in two models by different terms or the same term defines different concepts);
- Different graph structure (the models choose to connect relevant sets of concepts in different ways)
- Different scope (two models cover parts of the domain that only partially intersect or the scope of one model includes that of another model);
- Different granularity (the size of concepts differ across the models; a single concept of one model represents a piece of domain knowledge covered by several concepts of another model);
- Different focus (the models examine different modeling paradigms or adhere to different modeling conventions).

This list does not include the mismatches specific for the formal models employing advanced modeling primitives, such as typed relations and axioms (e.g. same entity can be modeled as a concept and as an attribute). In our study, we did not have to deal with this kind of discrepancies, neither we were resolving the language-level mismatches. Each of the problems listed above, however, occurred multiple times, while we were integrating the Java ontology and the learning objective models used in Problets.

Unlike QuizJet questions that are indexed with the concepts from the same ontology, each Problet relies on separate model of learning objectives. These models cover 6 large topics of Java programming language: (1)Arithmetic Expressions, (2) Relational Expressions, (3) Logical Expressions, (4) if/if-else Statements, (5) while Loops, (6) for Loops.

The combined scope of these topic models is several times narrower than the one of the Java ontology. At the same time the granularity of Problets' models is much higher. The total number of concepts in the Java ontology is about 500; the cumulative number of nodes in the Problets' models is more than 250. The most important problem we had to deal with is the difference on the modeling approaches (or different focus of modeling) used in Java ontology and Problets' domain models. Every learning objective models application of a concept in a particular learning situation (e.g. different objectives model the simple if clause in the if-else-statement and the simple if clause in the if-statement). In other words, a learning objective can be described as a concept put in a context. To properly map the context of a learning objective most of the time we had to connect one learning objective to several concepts from the Java ontology. To prevent too aggressive evidence propagation to the concepts modeling context of learning objectives, while mapping we also provided weights (from 0 to 1) that define how much knowledge of a particular concept define the proficiency of the learning objective. An example of mapping a learning objective into concepts is given by the Fig. 5. This terminal learning objective from the Selection topic defines the application of if-else statement, when the condition part of the statement evaluates to true value. To properly match this particular situation, we need to use three concepts from the Java Ontology. The assigned weights indicate that the main concept is still *IfElseStatement*, although the evidence of mastering this learning objective will slightly contribute to the knowledge of concepts *RelationalOperator* and *True*.

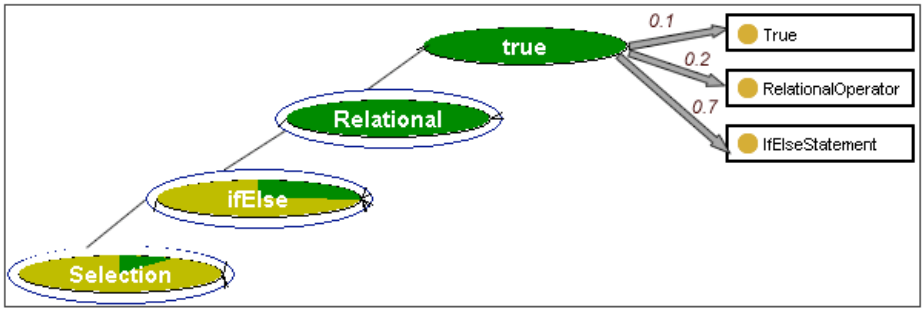

Fig. 5: An example of learning objective to concept mapping

4 Integration: The Implementation

To verify the practicality of the described approach we have administered a pilot study in the framework of an undergraduate Java course in the School of Information Sciences at the University of Pittsburgh. Both QuizJet quizzes and Ramapo College Problets are available to students via Knowledge Tree portal (Fig. 3). Knowledge Tree provides user authentication and authorization. It uses folder-document hierarchy to organize educational content. There are 6 Problets and 44 QuizJet questions are available to students of the introductory Java course taught at the School of Information Sciences, University of Pittsburgh.

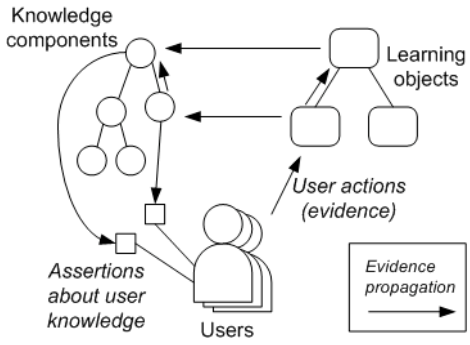

Fig. 6: Evidence propagation in CUMULATE [15].

Every time user submits an answer to the QuizJet question or Problet they notify user modeling server CUMULATE [15]. The notification is done via an HTTP request. Parameters of the request adhere to a simple protocol. Each report to user modeling sever should carry timestamp, group and user IDs, application and learning object the user worked with, and the result of the interaction (i.e., success, failure). Once evidence of new user action arrives, CUMLATE "propagates" it and updates overlay domain model of user knowledge. Special inference agents perform such propagation. Each agent is configured to deal with evidence from one or more external applications. Once new evidence arrives an agent is activated. For each learning object the agent retrieves corresponding metadata – concepts of the domain model (aka knowledge components). Based on the weights of the concepts for a particular learning object result of the interaction the agent updates assertions regarding user mastery of the concepts (Fig. 5). Note that the evidence propagation approach to user modeling applied by CUMULATE allows to take into account user work with both adaptive and non-adaptive learning objects – as long as concept-level metadata for these objects is provided by the content developer or any other stakeholder.

In the case of QuizJet and Ramapo Problets, the learning activities served by the systems are indexed with concepts of Java domain ontology described in section 3. This index along with weights (significance of each concept for the learning object) is cached by CUMULATE. The same inference agent handles evidence coming from

both QuizJet and Ramapo Problets. This agent increases the mastery level of the corresponding concepts in the case of successful interaction (correct answer) and does nothing otherwise. Since ontology mapping ensures that QuizJet and Problets use the same pool of domain concepts, successful interaction with quizzes and Problets have an identical effect on the level of student knowledge of domain concepts. Users work with either QuizJet quizzes, Ramapo Problets, or a combination of the two is reflected in the user model. The current "integrated" state of the user model is available for all approved components (including QuizGuide and Problets) and can be used to provide a more precise adaptation.

5 Discussion

In this paper we have proposed a lightweight solution for integration of user modeling information collected by different educational systems. The resulting infrastructure allows two applications developed by different research teams and relying on considerably different domain models to be used by the students of the same course. The applications separately collect the evidence about student knowledge and communicate it to the user modeling server, which allows us to support more holistic user models.

As the first step of the further investigation of this approach, we plan to evaluate the results of the pilot study conducted in the undergraduate Java course. The focus will be made on the assessment of the predictive validity of the obtained user models.

Although, the reported evidence is not currently used for adaptation, the necessary protocols have been implemented on the user modeling side. The Ramapo team is modifying the Problets in order to retrieve the aggregate user models from the CUMULATE server. On the University of Pittsburgh side a similar role will be played by the QuizGuide adaptive service [16].

We are also going to apply this solution in other learning domains (C, C++, SQL) and integrate with other adaptive systems, e.g. Canterbury University's constrained-based tutors [17] and University of Malaga's SIETTE [18].

References

1. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Neidl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) 291-324
2. Krüger, A., Baus, J., Heckmann, D., Kruppa, M., Wasinger, R.: Adaptive mobile guides. In: Brusilovsky, P., Kobsa, A., Neidl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) 521-549.
3. Brusilovsky, P., Peylo, C.: Adaptive and intelligent Web-based educational systems. International Journal of Artificial Intelligence in Education 13, 2-4 (2003) 159-172
4. Denaux, R., Dimitrova, V., & Aroyo, L. Integrating Open User Modeling and Learning Content Management for the Semantic Web. *In: L. Ardissono, P. Brna & A. Mitrovic*

(eds.), Proceedings of the 10th International Conference on User Modeling (UM'2005) (pp. 9-18), Edinburgh, Scotland, UK: Springer.

5. Dolog, P., & Nejdl, W. (2003). Challenges and Benefits of the Semantic Web for User Modelling. *In: P. De Bra (ed.), Proceedings of the AH2003 workshop*, Budapest , Hungary.
6. Niederée, C., Stewart, A., Mehta, B., & Hemmje, M. (2004). A Multi-Dimensional, Unified User Model for Cross-System Personalization. *In: Proceedings of the Workshop on Environments for Personalized Information Access at AVI'2004 (pp. 34-54)*, Gallipoli, Italy, from http://www.di.uniba.it/avi2004/e4pia/EPIA2004_proceedings.pdf
7. Sosnovsky, S., Dolog, P., Henze, N., Brusilovsky, P., Nejdl, W.: Translation of overlay models of student knowledge for relative domains based on domain ontology mapping. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Proc. of 13th International Conference on Artificial Intelligent in Education, AI-ED 2007. IOS (2007) 289-296
8. Berkovsky, S., Kuflik, T., Ricci, F.: Cross-technique mediation of user models. In: Wade, V., Ashman, H., Smyth, B. (eds.) Proc. of 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2006). Lecture Notes in Computer Science, Vol. 4018. Springer Verlag (2006) 21-30.
9. Brusilovsky, P., Sosnovsky S., Lee, D.H., Yudelson, M., Zadorozhny V., & Zhou X. (2008). An open integrated exploratorium for database courses. In: E. Menasalvas & A. Young (eds.) *Proceedings of 13th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'2008), Madrid, Spain, June 30 – July 2, 2008* (accepted).
10. Trella, M., Carmona, C., Conejo, R.: MEDEA: an Open Service-Based Learning Platform for Developing Intelligent Educational Systems for the Web. In: Proc. of Workshop on Adaptive Systems for Web-based Education at 12th International Conference on Artificial Intelligence in Education, AIED'2005. IOS Press (2005) 27-34.
11. Kumar, A.N. A Scalable Solution for Adaptive Problem Sequencing and its Evaluation. In Proceedings of The 2006 International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2006), Dublin, Ireland, June 21-23, 2006.
12. Kumar, A.N., Explanation of step-by-step execution as feedback for problems on program analysis, and its generation in model-based problem-solving tutors, Technology, Instruction, Cognition and Learning (TICL) Journal, Vol 4(1).
13. Kumar, A.N., A Reified Interface for a Tutor on Program Debugging, Proceedings of Third IEEE International Conference on Advanced Learning Technologies (ICALT 2003), Athens, Greece, 7/9-11/2003, 190-194.
14. Kumar, A.N. Using Enhanced Concept Map for Student Modeling in a Model-Based Programming Tutor. International FLAIRS conference on Artificial Intelligence, Melbourne Beach, FL, May 11-13, 2006, 527-532.
15. Brusilovsky, P., Sosnovsky, S. A., & Shcherbinina, O. (2005). User Modeling in a Distributed E-Learning Architecture. Paper presented at the 10th International Conference on User Modeling (UM 2005), Edinburgh, Scotland, UK, July 24-29, 2005
16. Brusilovsky, P., Sosnovsky, S., & Shcherbinina, O. (2004). QuizGuide: increasing the educational value of individualized self-assessment quizzes with adaptive navigation support. In J. Nall, & R. Robson (eds.) *Proceedings of World Conference on E-Learning (E-Learn 2004), Washington, DC, USA, November 1-5, 2004*, 1806-1813
17. Mitrovic, A., Martin, B., Suraweera, P. Constraint-based tutors: past, present and future. IEEE Intelligent Systems, special issue on Intelligent Educational Systems, vol. 22, no. 4, pp. 38-45, July/August 2007.
18. Conejo, R., Guzman, E., and Millán, E. (2004) SIETTE: A Web-based tool for adaptive teaching. International Journal of Artificial Intelligence in Education 14 (1), 29-61.