# Using Numeric Optimization To Refine Semantic User Model Integration Of Educational Systems

Michael Yudelson1, Peter Brusilovsky1, Antonija Mitrovic2, Moffat Mathews2
{mvy3, peterb}@pitt.edu, {tanja.mitrovic, moffat.mathews}@canterbury.ac.nz
[1] University of Pittsburgh, School of Information Sciences
[2] University of Canterbury, Department of Computer Science and Software Engineering

**Abstract**. Nowadays, it is a common practice to use several educational systems in one domain. In this situation, each of the systems should be able to provide the best user modeling based on the integrated data about the user. However, differences in domain conceptualization complicate the ability of the systems to understand each other's user models and necessitate the use of labor intensive and time consuming alignment procedures that require involvement of knowledge engineers. While the latter are best at detecting associative links between the user model items, they fail to reliably identify the strengths of these associations. In this paper, we are proposing a method to improve the user model mapping by using a numerical optimization procedure. Our results show that numerical weight optimization helped to decrease the amount of manual work and improved the target model accuracy.

## 1 Introduction

Over the recent decade, many state-of-the-art user-adaptive systems evolved from research prototypes to real-life production applications. Social interaction, information access, and E-Learning are the areas where the user-adaptive content is booming. In some fields (educational systems for example), the concentration of user-adaptive systems is such that several are available for each topic. Despite their availability, adaptive systems do not compete but complete each other, implementing only part of an ideal fully functional system. They offer unique features unable to completely replace their rivals. This creates a problem in using them together to embrace the full range of features offered separately. To enable coherent use, systems should be able to mutually exchange and understand collected user data. In an ideal scenario, user models should be integrated.

There are two major approaches to user model integration. The most widespread and popular is translation [15] (or mediation [1]). Here, systems exchange complete snapshots of their user models and *convert* them into local representations. The operation of translation is costly. However, since translation/mediation is usually employed when the systems are used sequentially, these costs stay insignificant. When systems are used in parallel, holistic translation/mediation poses a problem when large chunks of user data need to be exchanged and converted (even after minor updates to one of the models).

In the situation of parallel systems' use, evidence-based integration is more appropriate, because user-adaptive systems do not exchange whole user models. Instead, they share results of elementary user actions. When one user model is updated, only the changes are

conveyed to the other system's user model. The cost of conversion is lower as fewer user model items require conversion. Although, in the case of the evidence-based approach, the problem of user model integration is of a seemingly smaller scale; the question of user model conversion is still quite complicated. This is especially true for the field of adaptive education systems (AES). In AES, each learning object (problem, example, test) is commonly described in terms of constructs of the domain model often called concepts. In reality, systems seldom share the representation of the models, even in closely related domains. Under these conditions, evidence integration becomes a challenging task.

Because of the differences in the domain model representations, concepts from one system could be related to several concepts from the other; they may feature different strengths of relations. To be able to convert user model data between systems, researchers often employ domain experts and knowledge engineers who manually identify links between domain models of the systems and assign strengths to them. Although human experts are known to be able to find related complex relations between domain models effectively, they do have problems assigning numerical weights for the strengths [6]. The very same expert could change his/her weighting decision if the process is repeated.

In this paper, we are addressing the problem of evidence-based user model integration for two educational systems in the area of databases: SQL-Tutor [11] and SQL KnoT [14]. Our prior work on integrating the user models of these systems has shown that expert involvement only partially solves the problem of integration [2]. While experts agree on the concept associations, assigning weights is extremely time consuming and often leads to conflicting results. To solve these problems, we are proposing an optimization procedure intended to aid experts in mapping domain models of the two systems. This procedure is designed to significantly cut the amount of efforts and time, optimize the mapping weights, and potentially refine the structure of the concept mapping links.

The rest of the paper is organized in the following way. Section 2 addresses related work. Section 3 talks about the ongoing integration of the two systems in question, identifies a problem, suggests a solution, and lists our hypotheses. Section 4 discusses the experimental work. And finally, section 5 finishes with discussion points.

## 2 Related Work On User Model Integration

The task of making two adaptive systems understand each other's user models is quite challenging. It requires aligning the domain vocabulary of one against the other, so that both systems can correctly interpret assumptions about the users from the partner system and employ user data to improve their own performance. Domain model alignment leading to a successful integration requires resolution of multiple inconsistencies. The less complex are the inconsistencies at the language-level. Among these are differences in syntax, differences in clarity, varying use of semantics, etc. The more complex inconsistencies are the model-level mismatches occurring due to the discrepancy in structure and/or semantics of the domain models. Resolving these kinds of inconsistencies entails dealing with: naming conflicts (the same concept termed differently in two models or the same term defining different concepts); different graph structure (relevant sets of concepts connected differently); different scopes (e.g. two

models covering different parts of the domain); different granularities (a single concept of one model covering a portion of domain knowledge represented by several concepts in another model); and adherence to different modeling paradigms and conventions.

**Common ontology**. User models can be exchanged and mutually interpreted when user models of two systems rely on a common domain ontology. A good example of common ontology integration is the OntoAIMS project [7]. Here, two separate systems OWL-OLM and AIMS are deployed with mutual regard for interoperability. Both systems represent their domain models in the form of ontologies and user models – as overlays of these ontologies. The shared user model is populated and read by both systems. Another approach to common ontology integration is to devise a central user modeling server (e.g. Personis [10], CUMULATE [4]). Such servers store domain models, perform centralized user modeling, and deliver user data to participating adaptive systems.

**The intermediary ontology** is used when systems agree on a single ontology. Regrettably, the field of user-adaptive systems is far from producing ontologies, which would be widely accepted. It is often rather difficult to make an ontological commitment, as different research teams design ontologies with different views of the conceptualization of the domain. A good example of the use of an intermediary ontology is the M-OBLIGE architecture [12]. Here, a common reference ontology of SQL works as a multi-translator and facilitates the exchange of the user model information.

**Automatic ontology mapping**. The approaches presented above are practical solutions for semantic integration of multiple user-adaptive systems. However, their applicability is reduced by the need to either adapt to an alternative representation of the domain or to perform time-consuming and mostly manual knowledge engineering work. Nevertheless, cases exist where the use of ontologies for domain modeling permits automatic alignment them [9]. Automation helps to find matching concepts and relies on techniques of natural language processing, graph theory, information retrieval, and statistics to discover similar lexical patterns, conceptual sub graphs, and regularities in accompanying text. Fully implemented ontology mapping solutions are not yet known. But authors of [15], when investigating the applicability of automatic ontology mapping, have shown that it has the potential to be close to the best possible translation done by human experts.

**Evidence-based integration**. To successfully integrate user-adaptive systems, one does not have to align entire ontologies each time. Instead, integration could be performed on single units of user information: reports of user activity, assumptions about user knowledge, preferences, and goals – all described in terms of domain model concepts. This data arrives to the user model sequentially as a series of events. Translation is applied to one or a set of events as shown in Figure 1. Model values are often exchanged as soon as they are produced by one of the systems. An example of the evidence-based integration is the work on integrating Ramapo Problets and QuizJET systems [3]. Both systems have internal domain ontologies of Java programming and the ontologies are different in granularity and focus of modeling. Integration of user data is done on the side of QuizJET. QuizJET uses the ontology concepts in the classical descriptive metadata sense, while Ramapo Problets uses the concepts-in-a-context paradigm, where each concept is accompanied by a descriptor of its situational application (e.g. for-loop vs. for-

loop-executing-exactly-once). This leads to the situation when each of the Ramapo Problets' concepts-in-context is related to a sizable weighted set of QuizJET concepts.

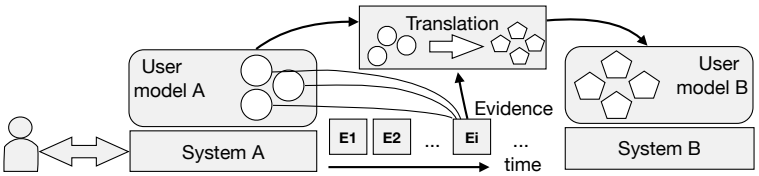


**Figure 1 Evidence-based integration of two user models**

The problem with all of the approaches above is that they heavily rely on time-consuming work on the part of experts to produce the mapping function. While experts are best for selecting the semantic relations between concepts of the domain models, the quality of their assignment of strengths to the inter-concept links is sub-optimal [6]. In our work, we are seeking to improve user model integration by utilizing an optimization procedure to either refine expert mapping weights or to produce them without human involvement, potentially altering the expert-suggested mapping links.

# 3 From Manual To Automated Student Model Mapping

## *3.1 Model Mapping For Evidence-Based Integration*

In our previous work, we have dedicated a lot of attention to evidence-based integration of user models in the SQL domain. This work revolved around two systems: SQL Knot and SQL-Tutor. SQL KnoT [14] is an external-loop tutor serving parameterized problems testing students' knowledge SQL. SQL KnoT represents domain model in the form of an ontology that has been developed by a team of human experts. Each problem is semi-automatically indexed with a set of ontology concepts. SQL-Tutor [11] is an ITS that presents problems to students and helps them to improve their knowledge of SQL. SQL-Tutor represents a domain model in the form of *constraints*. Constraints represent the fundamental principles of SQL and must be satisfied in any correct solution. Each SQL-Tutor problem has a set of relevant constraints. If the student solution violates these constraints, the solution is incorrect. The constraint set in SQL-Tutor contains about 700 constraints that assess the syntactic and semantic correctness of the solution.

The gist of the user modeling approach in both SQL KnoT and SQL-Tutor is quite similar: they construct an overlay of the domain vocabulary of knowledge items (constraints or concepts); nevertheless, the fundamental differences between the two domain models make the alignment of their models quite challenging. The unique nature of SQL-Tutor constraints makes the use of the known automatic ontology mapping techniques impossible. A constraint is not directly related to a single concept or a sub-tree of the ontology; instead, it models the syntactic or semantic relations between various concepts. As a result, the mapping is not one-to-one, but many-to-many.

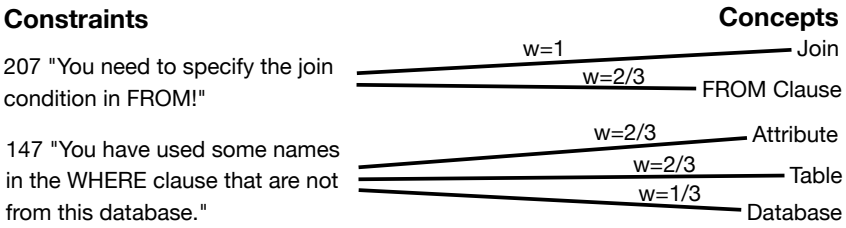


**Figure 2 A fragment of constraints-to-concepts mapping with weights**

We charged a number of experts with the task of mapping the two domain representations [13]. The resulting mapping contained a set of constraint-concept relations with the relevance weights: small (1/3), medium (2/3), or large (3/3=1). A fragment of the mapping is shown in Figure 2. The formula for computing the concept knowledge scores is shown in Equation (1). Namely, sum of weights between the concept and satisfied constraints minus sum of weights between the concept and broken constraints, divided by the sum of weights between the concept and all activated constraints.

$$S_{concept} = \left(\Sigma W^{+}_{(concept,constraint)} - \Sigma W^{-}_{(concept,constraint)}\right) \Big/ \Sigma W_{(concept,constraint)} \quad (1)$$

Both systems were deployed in an undergraduate database course at the School of Information Sciences, University of Pittsburgh during the Fall 2008 semester. Students had access to 300 SQL-Tutor problems and over 50 parameterized SQL KnoT problems. The student's progress was stored in one single long-term user model. SQL-Tutor was responsible for nearly a quarter of the students' problem-solving activity (SQL KnoT was responsible for the rest), despite that fact that SQL-Tutor was made available only in the middle of the semester and students were already familiar with SQL KnoT.

### *3.2 Problems Of Manual Mapping*

Despite tangible progress in integrating the two SQL problem-solving tools, there are still open questions pertinent to this particular integration effort and to user model integration in general. First, merging the user models leaves the issue of the merger quality unanswered. The fusion of the user models does not automatically improve the quality of the combined model. Second, we relied on expert opinion to come up with the mapping. From the literature, we know that the experts can reliably identify mapping relations between items of two domains [8]. However, assigning an appropriate weight – even on a simpler categorical scale – often poses a problem [6]. The same person can change his/her opinion if the weighting procedure is repeated. This means that, even if all relations between concepts and constraints were identified correctly, the assigned weights could still be suboptimal; as a result, the quality of the mapped model would be poor.

### *3.3 Towards Automated Model Mapping*

To answer these questions, we suggest using an optimization procedure to refine the expert-assigned constraint-to-concept weights. The idea is to employ the student logs of the source system (SQL-Tutor) to create and fine-tune a custom user model using least

square fitting method with the mean squared error as the criterion. Then, using the same student logs, we utilize the experts' mapping to compute the mean squared error of the now mapped concept-based user model. Performing the search in the weight-space, we are minimizing the error of the target model, thus refining the mapping.

Our hypothesis is that the suggested procedure would be able to, first, improve user model integration by optimizing expert weights. Second, starting with constant or random weights, produce the weights, without considering expert suggestion. Here, we expect the optimization results to be not as good as the one starting with expert mapping. And third, change the mapping links structure (by setting weights =0) for a more optimal one.

# 4 Experimental Study Of Automated Model Mapping

To verify the validity of the approach suggested, we conducted a set of experiments. First of all, to establish the baseline for mapping, we constructed the source constraint-based user model in such a way that modeling parameters minimize the modeling error. Second, the optimization was repeated. This time, the variables were the mapping weights and the objective function was the mean squared error of the mapped concept-based model.

Logs of the three database courses offered at the University of Canterbury in the 2006-2007 term were used for the experiment. We took the logs of the first course that contained 3544 transactions of 38 students. Each log entry contained user id, problem id, time the solution was submitted, solution correctness, list of confirmed constraints, and a list of broken constraints (if the solution is not correct). At the first stage of the experiment, we used user modeling approach that was different from the one deployed with SQL-Tutor, namely – Bayesian Knowledge Tracing (BKT) [5] – an established user modeling method in the area of intelligent tutoring systems. It the second stage, we used three sets of weights: supplied by the experts, equal constant weights, and the random weights. The details and the results of these procedures are given below.

## *4.1 Baseline Constraint-Based User Model*

As we have mentioned before, Bayesian Knowledge Tracing (BKT) was used as the baseline user modeling approach in SQL-Tutor. BKT assumes a two-state model of knowledge items (often called skills or rules) of a particular learning domain. The knowledge item (KI) is either in a learned or unlearned state. While interacting with the system, knowledge of KI can transition from the unlearned to the learned state. Even if a KI is in the learned state, a student can make a mistake. As in the unlearned state, there is a chance student can guess correctly. For each of the modeled KIs, BKT model maintains four parameter estimates: $p(L)$ – the probability that KI is in the learned state, the probability that KI is in the learned state prior to interacting with the tutor is $p(L_0)$; $p(T)$ – the probability that the KI will transfer to the learned state on next time user practices it; $p(S)$ – the probability the student will slip and apply the KI incorrectly even when it is in the learned state; and finally $p(G)$ – the probability that the student will apply the KI correctly despite it being in the unlearned state. For details on BKT models, refer to [5]

We adhered to the usual practice of BKT modeling and kept a unique set of parameters $p(L_0)$, $p(T)$, $p(S)$, and $p(G)$ for each KI and had a separate running estimate of $p(L)$ for each user-KI pair. Counter to the tradition, we didn't assume conditional independence of KIs and obtained estimates for all parameters together. Reason being the *special feature* of constraint-based model: constraints always spanned several domain concepts. One other usual BKT practice is using only the first chance to apply the knowledge item per problem attempt. This rule is often used in the so-called *inner-loop tutors* that walk students through each step of the problem solving activity. SQL-Tutor is an *outer-loop tutor*: it gives feedback only when a complete problem solution has been submitted. It is common to see same constraint both satisfied and broken in the problem several times not knowing which came first. We ignored repetitions, and when the constraint was both satisfied and broken, set the value of $p(L)$ to the arithmetic mean of $p(L)^+$ (assuming correct response came first) and $p(L)^-$ (assuming incorrect response came first).

**Table 1 Results of fitting source BKT model of SQL-Tutor constraints.**

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| p(L0) – a priori knowledge level | 0.642 | p(S) – probability of slip | 0.206 |
| p(T) – probability of transfer | 0.523 | p(G) – probability of guess | 0.187 |
| Mean Squared Error of modeling | 0.248 | | |

Out of about 700 constraints, we selected only those occurring in the user logs. In addition, we filtered out constraints that are always satisfied and never broken. This gave us 282 constraints with 4 parameters to estimate for each: a total of 1,128 parameters. Each constraint was treated as a separate KI. Mean squared error was used as an objective function of the optimization. To compute the BKT models, we used Matlab. Fitting was done using Matlab's implementation of linear square fitting procedure using trust region reflective algorithm. The parameters of the resulting source BKT model of constraints are shown in Table 1. Model parameters given are weight-averaged over all participating constraints (the number of constraint occurrences in the log weights its contribution).

## *4.2 Mapping Weights Optimization*

To optimize the model mapping weights, we modified the code used to construct the baseline constraint-based BKT model. User model parameters were held constant and mapping weights were used as variables. Objective function remained the same – mean squared error of modeling. The original constraint-based user model was updated as before, but for the computation of the user model error it was mapped to concept-based model using Equation (1). Since in the previous stage we reduced the number of participating constraints, the original number of 1,012 constraint-concept links/weights decreased to 576. Three sets of weights were used. The first set contained weights produced by experts: low, medium, or high (1/3, 2/3, and 1=3/3 respectively). In the second set, all weights were set to 0.5. In the third set, all weights were randomly chosen from 0.0 to 1.0. As in the case with BKT modeling of constraints, we did not assume conditional independence of sets of weights and all of the 576 weights were fit together. Matlab's constrained nonlinear multivariable optimization procedure was used with an active-set algorithm. Table 2 presents the results of the optimization experiments.

**Table 2 Results of weight mapping refinement**

| | Expert weights | Constant weight = 0.5 | Random 0.0 to 1.0 |
|---|---|---|---|
| Mean squared error before refinement | 0.454 | 0.453 | .463 |
| Mean squared error after refinement | 0.290 | 0.270 | .463 |
| Number of weights changed | 44 | 576 | 0 |
| Mean absolute weight change | 0.031 | 0.496 | 0 |
| Weights decreased / increased | 44 / 0 | 52 / 524 | 0 |
| Weights set to zero values | 34 | 52 | 0 |

As we can see, under all three starting conditions the initial mean squared error was roughly the same: 45%-46%. However, only the first two conditions (originating from expert weights and from constant weights) lead to improved modeling error rate. Optimization of the expert-supplied weights ended with a 29% error rate, only 4% worse than the original constraint-based BKT model. The mapping originating with constant weights of 0.5, counter to our expectations, beat that figure by 2% and reached a 27%. Optimization of random weights did not lead to any changes.

Although expert weights were closer to the found optimum (44 out of 576 weights were changed in the process of optimization and all of them were decreased) than constant weights (all weights were changed, 524 weights increased and 52 decreased), in both cases the optimization procedure arrived at similar results. In the process of optimization, 34 and 52 mapping links were removed in the expert and constant conditions respectively. Despite promising results, our solutions look like local optimums. Mean absolute difference of weights between expert-originated and constant-originated sets is 0.3581 and the number of removed links agreed upon is only 13.

## 5 Discussion

Our experiments have successfully shown the high potential of using numeric optimization to refine the expert's alignment of two user models in general. In our case the mapping of the source constraint-based mode of SQL-Tutor to the concept-based model of SQL KnoT was significantly improved. The original 45% error of the mapped model was reduced to 29%; only 4% shy of the source model's error. Taking into account principal differences between the two domain vocabularies mapped, the achieved result is impressive. The refinement of uniformly assigned constant weights, against our expectations, was able to achieve an even better 27% error. Although a 2% difference does not seem like a tangible one, what counts is that refining constant weights was comparable to the original constraint-based model. In the light of these results, in the future one could free experts from weight assignment entirely. The only thing left for experts to do is to set inter-model relations. The rest could be handled by the procedure we proposed. The optimization procedure also demonstrated the ability to not only change the weights, but also to drive them to zero as an indication of some of the model mapping links to be *redundant*. This could serve as an additional check of the mapping quality and as an aid to human experts in the by-hand iterative process of mapping.

Despite its merits, there are several limitations to this work. First, mapping links could only be removed, not added as in some automated mapping approaches. A possible remedy for this could be changing the link selection procedure when merging the experts' suggestions. Instead of using the relations agreed upon by the majority of experts, all suggested links could be considered and then the refinement procedure would do the necessary filtering. In general, additional expert verification of the refined mapping might be necessary. Computationally optimal mapping might not be pedagogically sound. Inappropriate mapping links might be emphasized by the use of maximal weights and important links could be removed. Second, we did not assume conditional independence of mapping weights and optimized all 576 weights at once. This posed a computation challenge for the optimization procedure. Also we used only a part of SQL-Tutor logs, one course worth of logs out of three available. A limited number of data points (3544) might have been a contributing factor for entrapment in the local minima of the objective function and possibly over-fitting. Mining large volumes of user data and enforcing conditional independence of the mapping weights could solve both of these issues. Third, the user model transformation formula inherently normalizes weights and scaling all mapping weights has no effect. For example, a set weights equal to {1, 2, 3} would be equivalent to {.1, .2, .3} or {.2, .4, .6}. However, optimization procedure treats scaled weights as completely different. Changing the transformation formula or the optimization procedure for the ones that don't suffer from this phenomenon could further improve the mapping. And finally, both the proposed procedure and the refined weights it produced could possibly be sensitive to a particular user modeling approach employed (Bayesian Knowledge Tracing in our case). At this point we cannot confirm whether the obtained weights would remain optimal, if modeling formalisms are replaced with alternative ones. We plan to continue this work and further investigate the issue.

## References


[1] Berkovsky, S., Kuflik, T., and Ricci, F. (2006). Cross-Technique Mediation of User Models. In V. P. Wade, H. Ashman, and B. Smyth (Eds.), 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2006), (pp. 21-30).

[2] Brusilovsky, P., Mitrovic, A., Sosnovsky, S., Mathews, M., Yudelson, M., Lee, D., and Zadorozhny, V. (2009). Database exploratorium: a semantically integrated adaptive educational system. In S. Berkovsky, F. Carmagnola, D. Heckmann, and T. Kuflik (Eds.), 7th International Workshop on Ubiquitous User Modelling (UbiqUM 2009).

[3] Brusilovsky, P., Sosnovsky, S., Yudelson, M., Kumar, A., and Hsiao, S. (2008). User model integration in a distributed adaptive E-Learning system. In S. Berkovsky, F. Carmagnola, D. Heckmann, A. Krueger, and T. Kuflik (Eds.), International Workshop on User Model Integration, (pp. 1-10).

[4] Brusilovsky, P., Sosnovsky, S. A., and Shcherbinina, O. (2005). User Modeling in a Distributed E-Learning Architecture. In L. Ardissono, P. Brna, and A. Mitrovic (Eds.), 10th International Conference on User Modeling (UM 2005), (pp. 387-391).

[5] Corbett, A. T. and Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction, 4(4), 253-278.

[6] Dawes, R. (1979). The robust beauty of improper linear models in decision making. American Psychologist, 34(7), 571-782.

[7] Denaux, R., Dimitrova, V., and Aroyo, L. (2005). Integrating Open User Modeling and Learning Content Management for the Semantic Web. In L. Ardissono, P. Brna, and A. Mitrovic (Eds.), 10th International Conference on User Modeling (UM 2005), (pp. 9-18).

[8] Giarratano, J. C. and Riley, G. (1989). Expert Systems: Principles and Programming. Boston, MA: PWS-Kent Publishing Co.

[9] Kalfoglou, Y. and Schorelmmer, M. (2003). Ontology Mapping: the State of the Art. The Knowledge Engineering Review, 18(1), 1-31.

[10] Kay, J., Kummerfeld, B., and Lauder, P. (2002). Personis: A Server for User Models. In P. D. Bra, P. Brusilovsky, and R. Conejo (Eds.), 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Second (AH 2002), (pp. 203-212).

[11] Mitrovic, A. (2003). An Intelligent SQL Tutor on the Web. Iinternational Journal of Artificial Intelligence in Education, 13(2-4), 173-197.

[12] Mitrovic, A. and Devedzic, V. (2004). A Model of Multitutor Ontology-based Learning Environments. Continuing Engineering Education and Life-Long Learning, 14(3), 229-245.

[13] Sosnovsky, S., Mitrovic, A., Lee, D. H., Brusilovsky, P., Yudelson, M., Brusilovsky, V., and Sharma, D. (2007). Towards integration of adaptive educational systems: mapping domain models to ontologies. In N. Capuano, D. Dicheva, A. Harrer, and R. Mizoguchi (Eds.), 5th International Workshop on Ontologies and Semantic Web for E-Learning (SWEL'2007).

[14] Sosnovsky, S. A., Brusilovsky, P., Lee, D. H., Zadorozhny, V., and Zhou, X. (2008). Re-assessing the Value of Adaptive Navigation Support in E-Learning Context. In W. Nejdl, J. Kay, P. Pu, and E. Herder (Eds.), 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2008), (pp. 193-203).

[15] Sosnovsky, S. A., Dolog, P., Henze, N., Brusilovsky, P., and Nejdl, W. (2007). Translation of Overlay Models of Student Knowledge for Relative Domains Based on Domain Ontology Mapping. In R. Luckin, K. R. Koedinger, and J. E. Greer (Eds.), 13th International Conference on Artificial Intelligence in Education (AIED 2007), (pp. 289-296).