Towards Integration of Adaptive Educational Systems: Mapping Domain Models to Ontologies

Sergey Sosnovsky[1], Antonija Mitrovic[2], Danielle Lee[1], Peter Brusilovsky[1], Michael Yudelson[1], Valeria Brusilovsky[1], Divyasheel Sharma[1]

[1]University of Pittsburgh, School of Information Sciences,
135 North Bellefield ave., Pittsburgh, PA, 15260, USA
{sas15, hyl12, peterb, mvy3, valeriab, dis2}@pitt.edu
[2]Dept. of Computer Science and Software Engineering, University of Canterbury
Private Bag 4800, Christchurch 8140, New Zealand
Tanja.Mitrovic@canterbury.ac.nz

Abstract. With the growth of adaptive educational systems available for students, semantic integration of user modeling information from these systems is emerging into an important practical task. Ontologies can serve as the major representational framework for such integration. However, not all adaptive systems rely on ontologies for representing domain knowledge. In this paper, we report an experiment on integration of domain models of two different adaptive systems.

1 Introduction

The expansion of WWW brings unmatched opportunities for dissemination of adaptive educational technologies. With the growing number of adaptive Web-based educational systems (AWBES) it becomes a realistic scenario to have several AWBES available to assist a student in the same domain. This opportunity comes with the challenge to consistently model users across several AWBES.

Traditionally, adaptive educational systems focus on the modeling of student's knowledge in the domain of learning, which includes a particular representation of the domain structure in terms of its elementary units and evaluation of how well a student knows these units. The mediation of such user modeling (UM) components will require target systems to achieve a certain level of mutual understanding of the domain semantics. Once the systems agree on the domain model, they can exchange overlay user information for equivalent or related parts of the domain and include it into the adaptation process. One of the first steps in this direction would be the implementation of the domain models with the help of ontologies, which are dedicated to express the shared view on the domain semantics. If two AWBESs rely on the common domain ontology, they can exchange their UMs and consistently interpret them when necessary. A good example of such a straightforward integration is given in [1]. Several research teams have generalized this approach to the level of architectures for UM semantic interoperability [2-4].

Unfortunately, the practice of AWBES is still far from the use of common ontologies. The designers of AWBES tend to employ different ontologies for the same domain, or develop their own. In this case, for semantic integration of the UMs, one can apply automatic ontology mapping techniques, and use the found mapping for UM mediation [5]. However, many successful AWBES do not utilize ontologies for domain representation. The adaptation and UM technologies used in these systems rely on formalisms, different from the conceptual networks that are the core components of ontologies. The semantic integration of such systems requires manual mapping of underlying domain models. Nevertheless, we argue that the ontologies can be used in these cases, as well, – as common denominators of such models and facilitators of future integration. In this paper, we attempt to make a first step towards this direction. We present an interesting case of domain model mapping for the field of database programming. To support students working with two AWBES for SQL language we have to integrate two very different models of student knowledge. One of the AWBES uses an overlay UM based on an ontology, while another employs constraint-based UM approach. We use the ontology underlying the first system as the reference model for constraint mappings. We also report the results of a small experiment evaluating the manual mapping provided by several experts.

2 SQL ontology used by SQL-Guide

SQL-Guide is an AWBES helping student to practice SQL skills by solving problems related to a subset of basic SQL concepts. A typical SQL-Guide problem description contains a set of predefined databases and a desired output, for which a student is asked to write a matching query. The system evaluates students' answers and provides simple feedback. To assist students in choosing the appropriate problem to practice, SQL-Guide adaptively annotates problem links with icons reflecting the student's progress. The system keeps track of the student's answers and computes the long-term model of his/her knowledge for the related concepts. A more complete description of the system can be found in [6].

The concepts indexing every problem template (and naturally every problem itself) come from the SQL Ontology. The ontology has been designed primarily to support the development of adaptive educational content for SQL and facilitate the integration of multiple educational systems in this domain, while ensuring the objective representation of SQL semantics. The level of granularity of the terminal concepts has been chosen to maintain the adequate modeling of students' knowledge with the necessary details. At the same time, our goal was not the comprehensive representation of the current SQL standard, therefore certain parts of the domain stay out of the scope if this ontology. The ontology is implemented as a light-weight OWL-Lite ontology. It specifies more then 200 classes connected to each other with one of the three relations: standard *rfs:subClassOf* (hyponymy relation) and a transitive relation pair *sql:isUsedIn* – *sql:uses*, which we have introduced to model the connection between two concepts, where one concept utilizes another. Fig. 1 gives an example of how these relations are used in the ontology. The ontology can be accessed at http://www.sis.pitt.edu/~paws/ont/sql.owl

```
<sql:WhereClause>        <rdfs:subClassOf>    <sql:Cluase>
<sql:SelectStatement>    <rdfs:subClassOf>    <sql:Statement>
<sql:WhereClause>        <sql:isUsedIn>       <sql:SelectStatement>
<sql:SelectStatement>    <sql:uses>           <sql:WhereClause>
```

Fig. 1. An extract from SQL Ontology

3 Constraint-based model of SQL-Tutor

SQL-Tutor is an intelligent tutoring system that helps university-level students to learn SQL. The system contains definitions of several databases, and a set of problems with the ideal solutions. At the beginning of a learning session, SQL-Tutor selects a problem for the student to work on. When the student submits a solution, the pedagogical module sends it to the student modeller that analyzes the solution, identifies mistakes and updates the UM. The pedagogical module uses the SM generates an appropriate pedagogical action. When the current problem is solved, or the student requires a new problem to work on, the pedagogical module selects an appropriate problem based on the UM. Several evaluation experiments have demonstrated the effectiveness and educational value of SQL-Tutor (see [7], for example).

The domain knowledge in SQL-Tutor is represented in the form of constraints specifying basic domain principles that must be satisfied by any correct solution [8]. A constraint consists of two conditions: the *relevance* condition specifies solution features for which the constraint is relevant, while the *satisfaction condition* specifies additional features which a solution must posses in order to be correct. Each condition may be a logical combination of simple tests applied on the solution. The constraint set in SQL-Tutor contains about 700 constraints checking the syntactic and semantic correctness of solutions. Fig. 2 illustrates two constraints (which we have used in the study reported in Section 4). Constraint 16 regulates the joint syntax of HAVING and GROUP BY clauses used in a query. Constraint 635 evaluates two semantically-equivalent alternative solutions for organizing nested queries: one based on the NOT IN predicate and another based on the EXISTS predicate.

The constraints are modular and problem-independent. They are matched to the student solution in parallel, and do not require complex reasoning. There is no one-to-one mapping between problems and constraints; a large number of constraints would be relevant for each problem. Constraint violations indicate errors in the student solution, and the system uses the feedback messages attached to constraints to provide feedback to students. The UM contains constraint histories, which are used to estimate student's knowledge. SQL-Tutor uses the UM to adaptively select problems at the right level of complexity for the student.

```
(p 16
(not (null (having ss)))
(not (null (slot-value ss 'group-by)))
"GROUP BY")
(p 635
(and (not (null (where ss))) (not (null (where is)))
   (match '(?*d1 ?a1 "NOT" "IN" "(" "SELECT" ?d5 "FROM" ?t ?*d2)
(where is) bindings)
   (not (member "IN" (where ss) :test 'equalp))
   (member "EXISTS" (where ss) :test 'equalp)
   (match '(?*d3 ??n "EXISTS" "(" "SELECT" ?a2 "FROM" ?t ?*d4) (where
ss) bindings))
(equalp ?n "NOT")
"WHERE")
```

Fig. 2. Two constraint examples

6 Semantic Integration Experiment

The fundamental differences in the domain models of two described systems make reliable automatic alignment of these models rather impractical. A well-established set of ontology mapping techniques cannot be applied for this task due to the unique nature of SQL-Tutor's constraints. A single 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 the concepts from the different parts of it.

To investigate the potential feasibility and effectiveness of the semantic integration between the two systems we have decided first to determine the best possible mapping provided by several experts for a limited set of constraints into concepts of the SQL ontology. Six experts participated in the experiment: two instructors teaching relational database courses to college students, two information science PhD students helping to teach SQL-related courses as teaching assistants, and two information science PhD students regularly using SQL for their programming projects.

Twenty constraints have been selected to cover both different parts of SQL domain and the variety of constraint types. Every expert has been asked to find the most relevant concepts for each of the constraints from the SQL-Tutor domain model. Experts have been provided with the hierarchical layout of the ontology. The task was to pick for every given constraint relevant concepts from the ontology and assign them with the ratings (low/medium/high) designating the importance of the relation between the constraint and the concept. No limitations were imposed on the time necessary to complete the mapping or the number of concepts used. Although the results of the manual mapping provided by experts are generally regarded as the golden standard comparing to the mapping acquired automatically, there is a number of important problems associated with human-provide expertise. The two, arguably, most important of these problems are: errors and subjectivity. As a result, there is generally a high level of disagreement even between the two experts.

The initial data analysis allowed us to confirm this phenomenon once again. Overall, experts used 61 different concepts from SQL ontology to map the assigned list of 20 constraints. Although, the average number of concepts used by a single expert to map a single constraint was only 3.15 with the standard deviation 1.16, the number of unique concepts used by all experts for mapping a particular constraint varied from 6 to 12. For 10 out of 20 constraints more than half mappings have been provided only by one of the six experts, which means, for the half of the constraints no agreement have been reached on more than a half of the mappings. Moreover, for two of the constraints no single mapping has been confirmed by the majority of the experts (4 out of 6). To numerically express the agreement between the experts, we computed the matching ratios for every pair. The ratio is essentially the percentage of mapping cases provided by the first expert, on which he/she has agreed to the second expert. For example, if the $exeprt_1$ found 100 mappings, out of which 50 have been confirmed by the $exeprt_2$, the rating of agreement of the $exeprt_1$ to the $exeprt_2$ is 50/100 = 0.5. The average rating of agreement among the expert pairs varied from only 40% to 66%. A more detailed analysis of the data showed that the main reason for such level of disagreement was that the experts approached the mapping task with two different strategies. While some experts provided very laconic set of mappings, others tended to over-specify the conceptual maps of the constraints. The total number of generated mappings varied from 40 to 81. Another important source of mapping mismatches was the difference in the usage of parent/child concepts. Some experts preferred to map a constraint into the set of sibling concepts, while the others moved instead one level up in the hierarchy and chose a single parent concept. In order to align the manually created mappings we are going to employ the taxonomic structure of the SQL ontology, which should allow to reduce the number of unpopular mapping though the basic ontological inference.

5 Discussion

We have presented an example of semantic integration of two separate AWBES employing different adaptation technologies and different domain models. The magnitude of diversity of the underlying domain models prevents us from using existing automatic techniques for semantic integration. Instead we rely on the mapping of domain models manually-provided by human experts. We have administered a small experimental study analyzing the results of the manual mapping. The experiments showed a very high level of disagreement between the experts. On the next step of the data analysis we plan to apply the ontological inference to automatically reduce the number of mapping mismatches. Another important direction of this project development is the implementation of the protocols for user authentication, remote application call, UM information exchange etc.

References

1. Denaux, R., Dimitrova, V., and Aroyo, L. Integrating Open User Modeling and Learning Content Management for the Semantic Web. In 10th International Conference on User Modeling (UM'2005), Edinburgh, Scotland, UK: Springer, 2005, pp. 9-18.
2. Mitrovic, A., and Devedzic, V. A Model of Multitutor Ontology-based Learning Environments. Continuing Engineering Education and Life-Long Learning, 14, 3 (2004), 229-245.
3. Niederée, C., Stewart, A., Mehta, B., and Hemmje, M. A Multi-Dimensional, Unified User Model for Cross-System Personalization. Workshop on Environments for Personalized Information Access at AVI'2004, Gallipoli, Italy, 2004, pp. 34-54.
4. Heckmann, D. Ubiquitous User Modeling. Berlin, Germany: Akademische Verlagsgesellschaft Aka GmbH, 2006.
5. Sosnovsky, S., Dolog, P., Henze, N., Brusilovsky, P., and Nejdl, W. Translation of Overlay Models of Student Knowledge for Relative Domains Based on Domain Ontology Mapping. In 13th International Conference on Artificial Intelligence in Education (AIED'2007), Marina Del Ray, CA, USA: IOS Press, 2007, pp. 289-296.
6. Sosnovsky, S., Brusilovsky, P., Lee, D.H., Zadorozhny V., & Zhou X. Re-assessing the value of adaptive navigation support in e-learning context. In Proceedings of 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2008), Hanover, Germany, July 30 – August 1, 2008 (accepted).
7. Mitrovic, A., Martin, B., Suraweera, P. Intelligent tutors for all: Constraint-based modeling methodology, systems and authoring. IEEE Intelligent Systems, special issue on Intelligent Educational Systems, vol. 22, no. 4, pp. 38-45, July/August 2007.
8. Ohlsson, S.: Constraint-based student modeling. In: Greer, J.E., McCalla, G (eds): Student modeling: the key to individualized knowledge-based instruction, 167-189, 1994.