

ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION FOR E-LEARNING

by

Sergey Sosnovsky

B. S. in Information Systems, Kazan State Technological University, Kazan, Russia, 1997

M. S. in Information Systems, Kazan State Technological University, Kazan, Russia, 1999

Submitted to the Graduate Faculty of
Information Sciences in partial fulfillment
of the requirements for the degree of
PhD in Information Sciences

University of Pittsburgh

2011

UNIVERSITY OF PITTSBURGH
INFORMATION SCIENCES

This dissertation was presented

by

Sergey Sosnovsky

It was defended on

November 30, 2011

and approved by

Peter Brusilovsky, PhD, Professor

Darina Dicheva, PhD, Professor

Daqing He, PhD, Associate Professor

Chad Lane, PhD

Michael Spring, PhD, Associate Professor

Dissertation Advisor: Peter Brusilovsky, PhD, Professor

Copyright © by Sergey Sosnovsky

2011

ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION FOR E-LEARNING

Sergey Sosnovsky, PhD

University of Pittsburgh, 2011

Conventional closed-corpus adaptive information systems control limited sets of documents in predefined domains and cannot provide access to the external content. Such restrictions contradict the requirements of today, when most of the information systems are implemented in the open document space of the World Wide Web and are expected to operate on the open-corpus content. In order to provide personalized access to open-corpus documents, an adaptive system should be able to maintain modeling of new documents in terms of domain knowledge automatically and dynamically. This dissertation explores the problem of open-corpus personalization and semantic modeling of open-corpus content in the context of e-Learning.

Information on the World Wide Web is not without structure. Many collections of online instructional material (tutorials, electronic books, digital libraries, etc.) have been provided with implicit knowledge models encoded in form of tables of content, indexes, headers of chapters, links between pages, and different styles of text fragments. The main dissertation approach tries to leverage this layer of hidden semantics by extracting and representing it as coarse-grained models of content collections. A central domain ontology is used to maintain overlay modeling of students' knowledge and serves as a reference point for multiple collections of external instructional material. In order to establish the link between the ontology and the open-corpus content models a special ontology mapping algorithm has been developed.

The proposed approach has been applied in the Ontology-based Open-corpus Personalization Service that recommends and adaptively annotates online reading material. The domain of Java programming has been chosen for the proof-of-concept implementation. A controlled experiment has been organized to evaluate the developed adaptive system and the proposed approach overall. The results of the evaluation have demonstrated several significant learning effects of the implemented open-corpus personalization. The analysis of log-based data has also shown that the open-corpus version of the system is capable of providing personalization of similar quality to the close-corpus one. Such results indicate that the proposed approach successfully supports open-corpus personalization for e-Learning. Further research is required to verify if the approach remains effective in other subject domains and with other types of instructional content.

TABLE OF CONTENTS

1.0	INTRODUCTION.....	1
1.1	THE PROBLEM AND MOTIVATION.....	1
1.1.1	Closed-corpus adaptive systems	2
1.1.2	Motivation: a closed-corpus Web-based adaptive educational system....	4
1.1.3	Problem: open-corpus personalization for e-Learning	6
1.2	OVERVIEW OF THE PROPOSED APPROACH	8
1.2.1	Three principle ways to semantic modeling of open-corpus content	8
1.2.2	Main dissertation approach: ontology-based open-corpus personalization for e-Learning	9
1.2.3	Algorithm for topic-to-ontology mapping	11
1.2.4	Combining it all together.....	11
1.3	RESEARCH QUESTIONS.....	13
1.4	DEFINITIONS.....	15
1.5	DISSERTATION OUTLINE.....	16
2.0	OVERVIEW OF THE RELATED RESEARCH FIELDS	20
2.1	INTRODUCTION	20
2.2	OPEN-CORPUS PERSONALIZATION FOR E-LEARNING	21
2.2.1	Extensible adaptive content	24

2.2.2	Term-based content modeling	25
2.2.3	Collaborative filtering	27
2.2.4	Social navigation	29
2.2.5	Semantic annotation	32
2.2.6	Information extraction	35
2.2.7	Semantic tagging	37
2.3	ONTOLOGY MAPPING.....	39
2.3.1	Ontology mapping based on a reference ontology	41
2.3.2	Ontology mapping based on lexical information	42
2.3.3	Ontology mapping based on ontology structure	44
2.3.4	Ontology mapping based on instance corpora	45
2.4	ONTOLOGICAL TECHNOLOGIES FOR PERSONALIZED E-LEARNING.....	46
2.4.1	Ontology-based representation of domain and student models	48
2.4.1.1	Ontology-based overlay user modeling and personal ontology views	50
2.4.1.2	Ontology-based user profiles.....	52
2.4.2	Ontology-based generation of domain and student models	54
2.4.3	Ontology-based open student modeling and adaptation	55
2.4.4	Ontology-based interoperability of adaptive educational systems.....	61
2.4.4.1	Integrating modeling approaches	62
2.4.4.2	Integrating domain representations	63
2.4.4.3	Integrating student profiles.....	64

2.4.4.4	Integrating modeling scales.....	66
2.5	CONCLUSION	67
3.0	EXTENDING THE SCOPE OF SEMANTIC MODELING OF OPEN-CORPUS CONTENT FOR E-LEARNING.....	69
3.1	INTRODUCTION	69
3.2	THREE APPROACHES TOWARDS SEMANTIC MODELING OF OPEN- CORPUS CONTENT	70
3.3	AUTOMATIC INDEXING OF STRUCTURED CONTENT.....	73
3.3.1	Automatic indexing of programming code-based learning objects.....	74
3.3.1.1	Parsing on-line C programming content	76
3.3.1.2	Prerequisite/outcome identification.....	77
3.3.2	Interactive support for refining results of automatic indexing	79
3.4	COARSE-GRAINED CONTENT MODELING AND PERSONALIZATION	82
3.4.1	Topic-based domain and content modeling in QuizGuide.....	83
3.4.2	Topic-based student modeling and adaptation in <i>QuizGuide</i>	87
3.4.3	Evaluation of topic-based personalization.....	91
3.4.3.1	Evaluation context and data collection	91
3.4.3.2	Learning effect.....	94
3.4.3.3	Motivational effect	96
3.4.3.4	Evaluation of topic-based user modeling: topic as an assessment unit	98
3.4.3.5	Evaluation of topic-based user modeling: predictive validity.....	102

3.4.3.6	Evaluation of topic-based adaptation.....	103
3.4.3.7	User behavior: a global picture.....	105
3.5	MEDIATION OF EXTERNAL MODELS	108
3.5.1	Manual mapping of a domain ontology and a pedagogical model	109
3.5.1.1	<i>Problets</i> and their pedagogical models	109
3.5.1.2	<i>QuizJET</i> and the Java ontology	112
3.5.1.3	Integration of <i>Problets</i> ' pedagogical models and the Java ontology	114
3.5.1.4	Summary.....	116
3.5.2	Manual mapping of a domain ontology and a constraint-based model	117
3.5.2.1	<i>SQL-Tutor</i> and constraint-based modeling.....	117
3.5.2.2	<i>SQL-Guide</i> and the SQL ontology	119
3.5.2.3	Semantic integration experiment.....	121
3.5.3	Automatic translation between two domain ontologies	124
3.5.3.1	Ontology mapping for student models translation	125
3.5.3.2	Evaluation of translation quality	128
3.6	SUMMARY: BUILDING UP THE MAIN DISSERTATION APPROACH	131
3.6.1	Automatic extraction of content models from structured and semi- structured content	133
3.6.2	Topic as a modeling unit	135
3.6.3	Ontology mapping as a basis for automatic model mediation	136

4.0	AUTOMATIC MODELING AND ADAPTATION OF OPEN-CORPUS CONTENT COLLECTIONS WITH THE HELP OF ONTOLOGY MAPPING.....	138
4.1	INTRODUCTION	138
4.2	STRUCTURED CONTENT ON THE WORLD WIDE WEB.....	139
4.3	PROPOSED CONTENT MODELING AND PERSONALIZATION WORKFLOW.....	141
4.3.1	Step 1: topic-based modeling of open-corpus content in terms of its structure.....	142
4.3.2	Step 2: mapping extracted model into the central domain ontology....	144
4.3.3	Step 3: mediated personalization of open-corpus learning material....	147
4.4	ADVANTAGES OF THE PROPOSED APPROACH.....	148
4.4.1	Independence of the human author.....	148
4.4.2	Independence of the domain	149
4.4.3	Independence of the adaptation technology	149
4.4.4	Independence of the content collection	150
4.4.5	Independence of the application field	151
4.5	CONCLUSION	152
5.0	ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION SERVICE.....	153
5.1	INTRODUCTION	153
5.2	THE PROOF-OF-CONCEPT IMPLEMENTATION SETTINGS.....	154
5.2.1	The subject domain and the instructional scenario.....	154
5.2.2	The core domain ontology	155
5.2.3	The open-corpus content collection.....	157

5.3	THE INTERFACE OF THE SYSTEM.....	158
5.3.1	Recommendation phase.....	159
5.3.2	Reading phase	160
5.4	THE ARCHITECTURE OF THE SYSTEM.....	162
5.4.1	Interaction Module	162
5.4.2	Modeling module.....	164
5.5	PERSONALIZATION IN <i>OOPS</i>	167
5.5.1	<i>QuizJET</i> exercise indexing	168
5.5.2	Modeling students' knowledge of Java programming	169
5.5.3	Two models of the textbook	171
5.5.4	Recommendation of reading material.....	171
5.5.5	Adaptive link annotation.....	173
5.6	CONCLUSION	175
6.0	ALGORITHM FOR TOPIC-TO-ONTOLOGY MAPPING.....	177
6.1	INTRODUCTION	177
6.2	MAPPING CONCEPTS AND TOPICS.....	179
6.3	NAME-BASED MAPPING	180
6.4	INSTANCE-BASED MAPPING.....	185
6.5	STRUCTURE-BASED SIMILARITY MERGING	190
6.6	ORDER-BASED MAPPING REFINEMENT	194
6.7	CONCLUSION	199
7.0	EVALUATION OF THE ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION SERVICE.....	202

7.1	INTRODUCTION	202
7.2	EVALUATION METHODOLOGY	204
7.2.1	Three versions of the system	204
7.2.2	Experiment Design.....	207
7.2.3	Overview of study materials	210
7.2.3.1	Textbook	210
7.2.3.2	QuizJET exercises	210
7.2.3.3	Pre- and post-tests.....	211
7.2.3.4	Post-questionnaires	211
7.2.4	Data collection	212
7.2.5	Subjects	213
7.3	LEARNING EFFECTS.....	214
7.3.1	General learning effect	216
7.3.2	Effect on learning complex material	218
7.3.3	Effect on weaker students	221
7.3.4	Effect on learning conceptual material.....	223
7.4	USING THE SYSTEM.....	225
7.4.1	Quality of recommendation	226
7.4.1.1	Recommendation acceptance rate	227
7.4.1.2	Average accepted recommendation rank	229
7.4.1.3	Browsing occurrence rate.....	230
7.4.1.4	Perceived value of recommendation.....	231
7.4.2	Effect on self-assessment exercises	232

7.5	SUBJECTIVE EVALUATION	234
7.5.1	Questionnaire: open-corpus vs. closed-corpus recommendation	236
7.5.2	Questionnaire: general questions	238
7.6	CONCLUSION	240
8.0	CONCLUSION AND DISCUSSION	243
8.1	SUMMARY OF THE RESULTS.....	243
8.1.1	Revisiting the research questions	243
8.1.2	Contribution to the field of open-corpus personalization	248
8.1.3	Contribution to the field of semantic web and ontology mapping	249
8.2	DISCUSSION.....	251
8.2.1	Limitations of the proposed approach	251
8.2.2	Future work directions	253
8.2.2.1	Further evaluation of the proposed approach.....	253
8.2.2.2	Improvement of the current implementation.....	254
8.2.2.3	Developing a bigger picture	255
	APPENDIX A	257
	APPENDIX B	259
	APPENDIX C	261
	APPENDIX D	262
	APPENDIX E	265
	APPENDIX F	277
	BIBLIOGRAPHY	285

LIST OF TABLES

Table 1. Comparison of open-corpus personalization technologies	23
Table 2: Students’ gender, experience and attendance distribution.....	94
Table 3: Basic statistics of using the system (adaptive vs. non-adaptive).....	98
Table 4: Quality of progress-based navigation: comparing the percentage of “good” navigational decisions.....	105
Table 5: Parameters characterizing the self-motivated activity of students with and without adaptive annotations.....	107
Table 6: Levels of relevant expertise	122
Table 7: Basic metrics of the mapped ontologies.....	126
Table 8: Example of terms extracted from naming labels with their idf scores	182
Table 9: Syntactic similarity between terms	183
Table 10: Java topics used in the experiment	208
Table 11: Design of the experiment.....	209
Table 12: Diagnostic tests and knowledge gain results (session 1)	215
Table 13: Diagnostic tests and knowledge gain results (session 2).....	216
Table 14: General learning effect statistics ($Score_{post-test}$ VS. $Score_{pre-test}$).....	217
Table 15: Weaker students statistics.....	223
Table 16: Statistics on learning conceptual material	225

Table 17: Basic recommendation statistics.....	227
Table 18: Extra recommendation statistics.....	228
Table 19: QuizJET statistics.....	233
Table 20: Post-questionnaire results.....	235

LIST OF FIGURES

Figure 1: A closed-corpus adaptive system	3
Figure 2: A closed-corpus adaptive system in the open-corpus settings	4
Figure 3: Open-corpus personalization in the context of e-Learning	5
Figure 4: Main dissertation approach and its relations to other contributions of this dissertation	12
Figure 5: Organization of the main dissertation chapters and relations between them	19
Figure 6: Research fields contributing to this dissertation.....	20
Figure 7: Classification of open-corpus personalization technologies	23
Figure 8: Extensible adaptive content: new documents can be added and modeled manually	25
Figure 9: Term-based content modeling: new documents are automatically modeled as “bags of words”	27
Figure 10: Collaborative filtering: documents are modeled based on the similarity of users’ experience with them	29
Figure 11: Social navigation: documents are modeled based on the history of user activity	31
Figure 12: Semantic annotation: pieces of new documents are automatically associated with the ontology concepts	34
Figure 13: Sources of information used by ontology mapping algorithms	41
Figure 14: Ontological technologies for personalized e-Learning	48
Figure 15: Ontology-based visualization of student models in <i>OWL-OLM</i>	57

Figure 16: Ontology-based verification of students' concept maps in <i>VCM</i>	58
Figure 17: Student model visualizations in <i>COMOV</i>	59
Figure 18: Visualizing conceptual user models in <i>VIUM</i>	61
Figure 19: Relationships between top categories of <i>IEEE PAPI</i> and <i>IMS LIP</i>	66
Figure 20: Modeling open-corpus content: automatic indexing of structured content.....	71
Figure 21: Modeling open-corpus content: coarse-grained domain modeling makes manual indexing easier	72
Figure 22: Modeling open-corpus content: model mediation utilized for content cross-modeling	73
Figure 23: Typical interactive example in <i>WebEx</i>	74
Figure 24: Typical self-assessment question in <i>QuizPACK</i>	75
Figure 25: Pseudo-code for prerequisite/outcome identification.....	78
Figure 26: Using navigable domain ontology to refine the results of automatic indexing.....	81
Figure 27: Topic-based content modeling in <i>QuizGuide</i>	83
Figure 28: Topic-based content modeling in <i>QuizGuide</i>	84
Figure 29: Topic-based structure of the C-Programming course: links represent <i>isPrerequisiteFor</i> relations.....	86
Figure 30: A topic-based adaptive system.....	88
Figure 31: Interface of <i>QuizGuide</i>	89
Figure 32: Topic-based annotations in <i>QuizGuide</i>	90
Figure 33: Non-adaptive access to quizzes.....	93
Figure 34: Topic-based learning curve	99

Figure 35: Time distribution of students' activity with QuizGuide and QuizPACK in the 4 th Semester	106
Figure 36: A <i>Probletem</i> on <i>if/if-else</i> statements in Java	111
Figure 37: A part of the domain hierarchy on <i>if/if-else</i> statements in Java; learning objectives are associated with each concept in the hierarchy	112
Figure 38: An example of <i>QuizJET</i> question on <i>Decisions</i> in Java accessed through the <i>Knowledge Tree</i> Learning Portal	113
Figure 39: An extract from the Java ontology	114
Figure 40. Mapping learning objectives to concepts	116
Figure 41: The interface of SQL-Tutor.....	118
Figure 42: Two example constraints.....	119
Figure 43: Interface of SQL-Guide.....	120
Figure 44: SQL Ontology	121
Figure 45. Relation between C and Java knowledge	125
Figure 46. The user models of knowledge for two concepts represented as 2-dimensional vectors	130
Figure 47. Classification of open-corpus content modeling technologies, including investigated and not-investigated technologies and the relation between them and this dissertation.....	132
Figure 48. Technologies for automatic open-corpus content modeling	134
Figure 49: Example of a topic-based model extraction based on the resource formatting	144
Figure 50: RDF-serialized extracted topic-based model.....	144
Figure 51: Mapping extracted models into central domain ontology	146
Figure 52: Meditated personalization of open-corpus learning material.....	148

Figure 53: The proposed approach in the spectrum of open-corpus content-based personalization technologies	152
Figure 54: Central Java ontology	156
Figure 55: Extracted model of the sample Java textbook	158
Figure 56: OOPS interface: list of recommendations	160
Figure 57: OOPS interface: reading a recommendation	161
Figure 58: OOPS architecture: communication with other components of <i>ADAPT</i> ²	164
Figure 59: The general workflow for modeling an open-corpus content collection	165
Figure 60: <i>QuizJET</i> exercise indexing	169
Figure 61: <i>CUMULATE</i> 's report on student's knowledge.....	170
Figure 62: OOPS uses model mapping to order and recommend open-corpus reading resources	173
Figure 63: Levels of adaptive annotation in OOPS.....	174
Figure 64: Adaptive link annotation algorithm in OOPS.....	175
Figure 65: The workflow of ATOM.....	178
Figure 66: Name-based mapping in ATOM	181
Figure 67: An extract from the page associated with the topic “ <i>The_break_and_continue_Statements</i> ”	186
Figure 68: Instance-based mapping in ATOM	188
Figure 69: Structure-based similarity merging in ATOM	192
Figure 70: Order-based mapping refinement.....	196
Figure 71: Example of learning goal distance matrix calculation	198
Figure 72: Three versions of the system and expected trend of effects.....	203

Figure 73: System interface: initial interaction (choosing a session and selecting a question)..	205
Figure 74: System interface: <i>experiment</i> -condition (QuizJET + OOPS-generated recommendations).....	205
Figure 75: System interface: <i>control</i> ⁰ -condition (regular QuizJET interface + the hard copy of the textbook)	206
Figure 76: System interface: <i>control</i> ¹ -condition (QuizJET interface + closed-corpus recommendation)	207
Figure 77: Experiment schedule	210
Figure 78: Subject screening protocol for the experiment.....	214
Figure 79: Knowledge gain vs. pre-test scores: <i>Left</i> - Session 1; <i>Right</i> – Session 2	222
Figure 80: Questionnaire: open-corpus recommendation.....	237
Figure 81: Questionnaire: closed-corpus recommendation	237
Figure 82: Questionnaire: general questions.....	239
Figure 83: Main components of <i>ADAPT</i> ²	258
Figure 84: Principle architecture of <i>CUMUALTE</i>	260
Figure 85: Presentation of a QuizJET exercise.....	263
Figure 86: Feedback in QuizJET	264

1.0 INTRODUCTION

1.1 THE PROBLEM AND MOTIVATION

Adaptive systems and technologies have recently started taking their place in our everyday life. Adaptive access to news and music, adaptive filtering of search results and social network feeds, adaptive recommendation of movies, books and other products are just several examples of mainstream applications of personalization. Despite this remarkable progress in adoption of adaptive technologies for general information access, adaptive educational systems (AES)¹ are still straggling with the transition from university labs to the world of commercial services. This is especially regrettable, as computer-aided instruction was one of the first application areas for the idea of user-adapted interaction (Carbonell 1970)).

Nowadays, the majority of AES are implemented on the World Wide Web (WWW). As an infrastructure for information and service delivery, WWW provides AES with unique opportunities in terms of user access and availability, inter-component connectivity, and the volume of information resources. However, it also introduces new challenges. One of the problems of AES that is especially important from the perspective of technology dissemination and uptake is open-corpus personalization. In the pre-Web era, the amount of learning resources available to a user of an AES was restricted to the pre-authored corpus of documents, learning

¹ See Definition 1 (Section 1.4)

problems, item descriptions, etc. The system owned its content and did not have access to any documents outside itself. On the Web, which is ultimately the global network of information items, every document is potentially available for any system. The challenge is how to provide adaptive access to new documents.

1.1.1 Closed-corpus adaptive systems

The goal of any adaptive information system is to tailor its content or behavior in order to support individual users in their information tasks and make their work more efficient. Traditional content-based adaptation relies on composite domain models and associations between content items (tutorial pages, problems, etc.) and domain elements. It allows adaptive systems to reason in terms of domain semantics instead of content and to generalize users' characteristics based on the history of their interactions with the system. For the closed-corpus applications operating on restricted sets of content items, it has been the role of a human expert to associate the content items with the elements of the domain knowledge at the authoring stage.

Definition 2 (Section 1.4) explains the most important characteristics of a conventional closed-corpus adaptive system, and Figure 1 graphically presents a general design for such systems. The adaptive content is embedded within the system along with all other components. Once the system is developed, no new documents can be added to it. Authoring tools and shells (Murray 1999) remedy this restriction to some extent. They greatly reduce the authoring time of new adaptive systems by reusing existing adaptation and user modeling components. However, every new system would still require manual development of the content model.

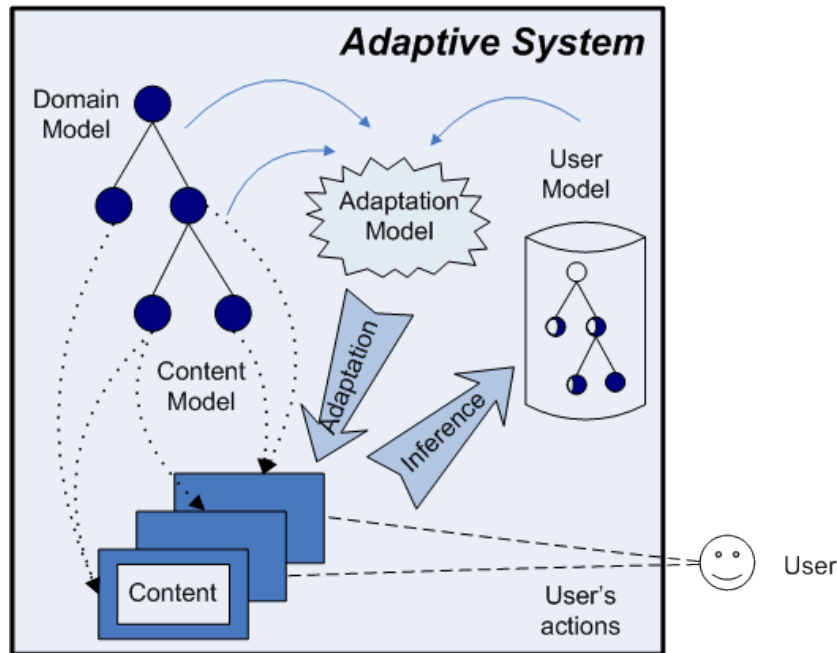


Figure 1: A closed-corpus adaptive system

In the global Web settings, this approach is no longer scalable, and, most importantly, it does not allow for coherent support of a modern user that accesses information from multiple sources and cannot be restricted to a limited set of content within a single system. Figure 2 represents a closed-corpus adaptive system facing the problem of open-corpus personalization. There is a large volume of relevant content available for the user on the Web that has not been processed by the developers of the system. The system cannot trace user's interactions with this content; neither can it include them in the adaptive inference to maintain the user model adequately.

It is virtually impossible to ensure adaptive access to the open-corpus content, as long as content models are created by human experts at the authoring time. For some information tasks on the Web, the closed-corpus personalization is inapplicable in principle because of the dynamic nature of the Web-content. For example, in the case of adaptive Web search or adaptive

news recommendation, new documents are discovered and existing ones can be modified in real time.

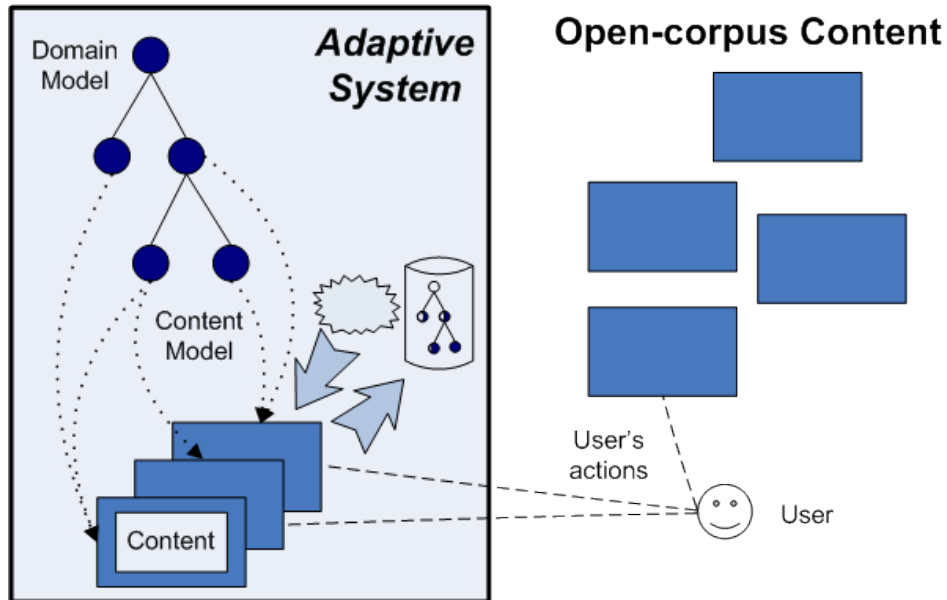


Figure 2: A closed-corpus adaptive system in the open-corpus settings

1.1.2 Motivation: a closed-corpus Web-based adaptive educational system

From the educational perspective, the WWW can be viewed as a very large collection of learning material. For many subjects, one can find online tutorials, textbooks, examples, problems, lectures slides, etc. Nowadays, teachers often do not have to create most of course materials themselves, instead they can reuse the best content available online.

For example, a teacher developing a course on Java programming might decide to use one of the Java Tutorials from (Oracle 2011), the electronic version of the chosen course book, PowerPoint presentations of the course lectures, an existing Web-based assessment system and online code examples. Although these resources are useful for students, they might get lost

in this amount of content without additional guidance. Organizing adaptive access to the course materials would help solve the problem. Appropriate tutorial pages can be recommended to the students based on their progress; adaptive navigation can be implemented to facilitate the choice of the most relevant example; an intelligent tutoring system can provide problem solving support and/or sequence learning problems adaptively. A teacher might be able to find a system implementing one of these technologies and providing adaptive access to one of the collections of learning material. However this system will not be aware of the rest of available content, unless it supports open-corpus personalization.

Figure 3 visualizes this scenario. A Web-based adaptive testing system has no knowledge about the external content available to students and can neither adapt it nor trace students' interaction with this content for better modeling of their knowledge. Figure 3 illustrates the main motivation of this dissertation: how can an adaptive system look beyond the pre-authored set of learning material, how can it model external learning content, present it to the students in an adaptive way and trace students' progress with this content.

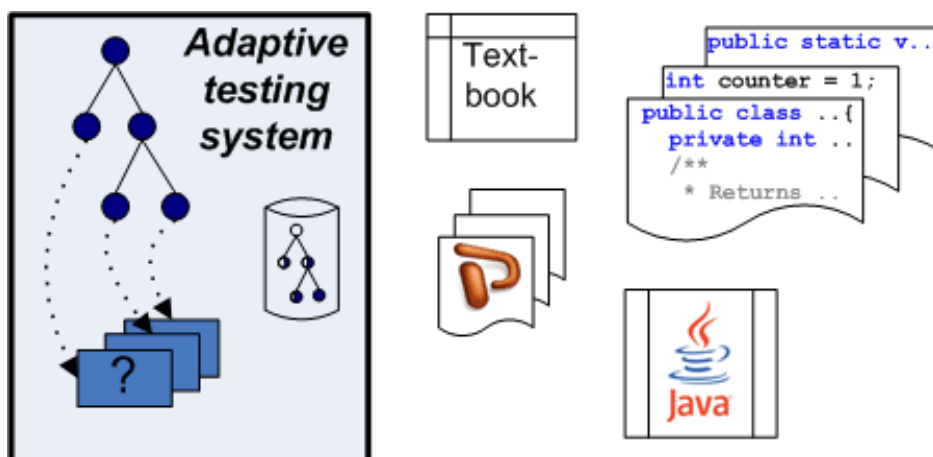


Figure 3: Open-corpus personalization in the context of e-Learning

The most important practical advantage of implementing open-corpus personalization is that it helps to eliminate (or greatly reduce) the need for manual authoring. Traditional content authoring for e-Learning systems is a very tedious and expensive procedure. Creation of a single learning object requires from the author both subject and pedagogical knowledge. In 2001, Downes estimated that development cost for a typical university-level course can be as high as \$100.000 (Downes 2001). In the case of AES, a new layer of complexity is added, as they rely on the development of sophisticated domain models and provision of semantic metadata for the learning content (Brusilovsky and Peylo 2003). Authoring of such semantic learning content will also require expertise in knowledge engineering and metadata standards.

1.1.3 Problem: open-corpus personalization for e-Learning

Definition 3 (Section 1.4) summarizes the main aspects of open-corpus personalization: unrestricted corpus of documents and content modeling at post-design time. Overall, the problem of open-corpus personalization is a multi-step problem. A full-scale solution to this problem in the context of e-Learning should include the following phases:

- discovery of open-corpus instructional content,
- identification of individual learning objects in open-corpus content,
- modeling of open-corpus content in terms of domain knowledge,
- connection of open-corpus learning objects to each other and to the preexisting content,
- providing adaptive access to open-corpus content,
- tracing of users' interaction with the open-corpus content,
- user modeling based on interactions with the open-corpus content.

Depending on the kind of adaptation technology and the supported information task, some of these phases can be more challenging and more important. For example, personalized search systems mostly rely on a well-defined methodology for automatic extraction of term-based content models (Micarelli, Gasparetti et al. 2007). Recommender systems using collaborative filtering to represent information items in terms of users' ratings (Konstan, Miller et al. 1997).

This dissertation focuses on the open-corpus personalization based on semantic content models, as the dominant personalization approaches in the field of e-Learning rely on representation of student knowledge and learning activities in terms of domain semantics. Thus automatic extraction of domain knowledge from the Web content becomes the central component of the problem addressed in this dissertation.

Unlike other fields, e-Learning has very strict requirements to the accuracy of adaptive systems. As a category of users, students are very susceptible to the system's failures and are rarely able to recover from those on their own. Wrong instructional material presented/recommended to a student can lead to confusion, frustration, lack of motivation and, essentially, poor learning. Therefore, the implemented AES not only should support adaptive access to open-corpus content, but also maintain a high level of adaptation accuracy.

Open-corpus personalization can greatly advance the reusability of learning content and adaptation components of e-Learning systems. However, it would require that the implemented technology for open-corpus content modeling is independent (at least to some extent) from the subject domain and the adaptation method.

To summarize, this dissertation addresses the problem of implementing an open-corpus personalization technology in the field of e-Learning. This technology should cover most phases of the open-corpus personalization workflow. It should rely on semantic content modeling. It

should maintain an acceptable level of adaptation accuracy. It should be independent of the subject domain and support different adaptation methods.

1.2 OVERVIEW OF THE PROPOSED APPROACH

1.2.1 Three principle ways to semantic modeling of open-corpus content

The first set of results of this dissertation comes from a detailed analysis of the principle approaches to the problem of creating semantic models of open-corpus content. This line of research is based on several projects undertaken by the author during his PhD study. In many respects, these results motivated the author to focus on the topic of ontology-based open-corpus personalization for e-Learning and helped to formulate the main dissertation approach.

Automatic indexing of structured content was applied to extract semantic models from the programming code of self-assessment exercises and interactive examples. Later these results were used to trace and model students' progress in the domain of C-Programming and implement adaptive services helping students to choose a proper learning object.

Relaxing requirements to domain modeling brought about a novel approach to student modeling and adaptation based on coarse-grained, content-driven topics. *Coarse-grained content modeling and personalization* has been implemented in the AES *QuizGuide* that adaptively navigates students to the best topic based on their knowledge and the current learning goal. *QuizGuide* has been thoroughly evaluated in real classroom setting in the domains of C programming, SQL and Java programming. Among other results, it has been shown that while considerably reducing the efforts of creating an adaptive system, *QuizGuide* allowed for

effective adaptive support of e-Learning in all the three domains. However, the topic-based student modeling suffered from low precision and poor predictive validity.

Utilization of external models was used to communicate the domain and students knowledge between several systems. Two experiments on manual model mediation and one experiment on automatic translation of student models based on domain ontology mapping were conducted. The later experiment showed that automatic ontology mapping can support model mediation of the quality statistically equivalent to the quality of manual model mediation.

1.2.2 Main dissertation approach: ontology-based open-corpus personalization for e-Learning

The main dissertation approach builds on the three solutions briefly described in the previous section to implement a fully automatic broadly applicable open-corpus personalization in the context of e-Learning.

An important motivation for this approach comes from recognizing the fact that there exists a large amount of high-quality online instructional material that is implicitly provided with semi-structured knowledge models. Tutorials, electronic books, digital libraries, manuals, etc. are the collections of such material. They have been authored by experts in the fields, who when creating them had certain knowledge organizations in mind. Most importantly, these organizations have been encoded within the collection in several ways: as tables of contents, indexes, headers of sections and chapters, links between the pages, and different styles of page fragments. All these structural and formatting elements have been introduced to order the material within the collection and make it easier to understand. Inescapably, they convey the model of the content and the domain, as the author understands it.

The extraction of this hidden semantic layer can be effective more or less depending on the exact formatting of the content and the domain-dependent techniques employed. However, even the least ambitious solution, relying only on top-level structural relationships between the pages will help to automatically extract a model of collection. The positive side of such approach is that it can be applied in many different domains and for a broad range of collection types. The weaker side is that the extracted model is not very detailed: a topic per page or section. Yet, as the *QuizGuide* example has shown such a content model can effectively support adaptive access to instructional material.

Extraction of topic-based models from open-corpus content is not enough for two reasons:

1. Topic-based structures cannot support student modeling of good quality, as demonstrated by the evaluation of *QuizGuide*.
2. A single topic-based model of a collection can be used to adaptively present only the content of this collection. The rest of the learning material used in a course stays isolated, as well as any other open-corpus collections that a teacher might want to use.

The solution proposed and implemented in this dissertation is to employ the central domain ontology as a reference model. It is used to assess students' knowledge and to translate between the topic-based structures of individual open-corpus collections. The connection between the harvested models and the central domain ontology is established based on the automatic mapping between the open-corpus model and the central ontology. To maintain adaptation of the open-corpus content, student's knowledge is mediated on the fly between the overlay of the domain ontology and the coarse-grained models of open-corpus collections.

1.2.3 Algorithm for topic-to-ontology mapping

The difference between the two models (a semiformal open-corpus content structure and a formal fine-grained ontology) necessitated the adjustment of the traditional mapping techniques and motivated the development of new algorithm for alignment of coarse-grained knowledge models to ontologies. Such an algorithm was developed (the details of its implementation are described in Chapter 6.0).

The Algorithm for Topic-to-Ontology Mapping (ATOM) estimates the equivalence between the individual concepts of the central ontology and individual topics of the collection structure. The output of the algorithm is the matrix of similarities between all concepts and all topics. To compute such a matrix, ATOM consequently employs four different mappers, each taking as input individual sources of information about the models: the naming labels, the instance corpora, the structural relations, and the ordering of model elements. The resulting matrix of similarities is used to assess the relevance of a topic to a target self-assessment exercise. This information together with the student's knowledge of ontology concepts helps to decide whether the topic should be recommended and what is the state of adaptive annotation for this topic.

1.2.4 Combining it all together

Figure 4 graphically represents the relations between the main approach of this dissertation and the rest of the contributing techniques. It shows how the main approach has built up as the fusion of several ideas.

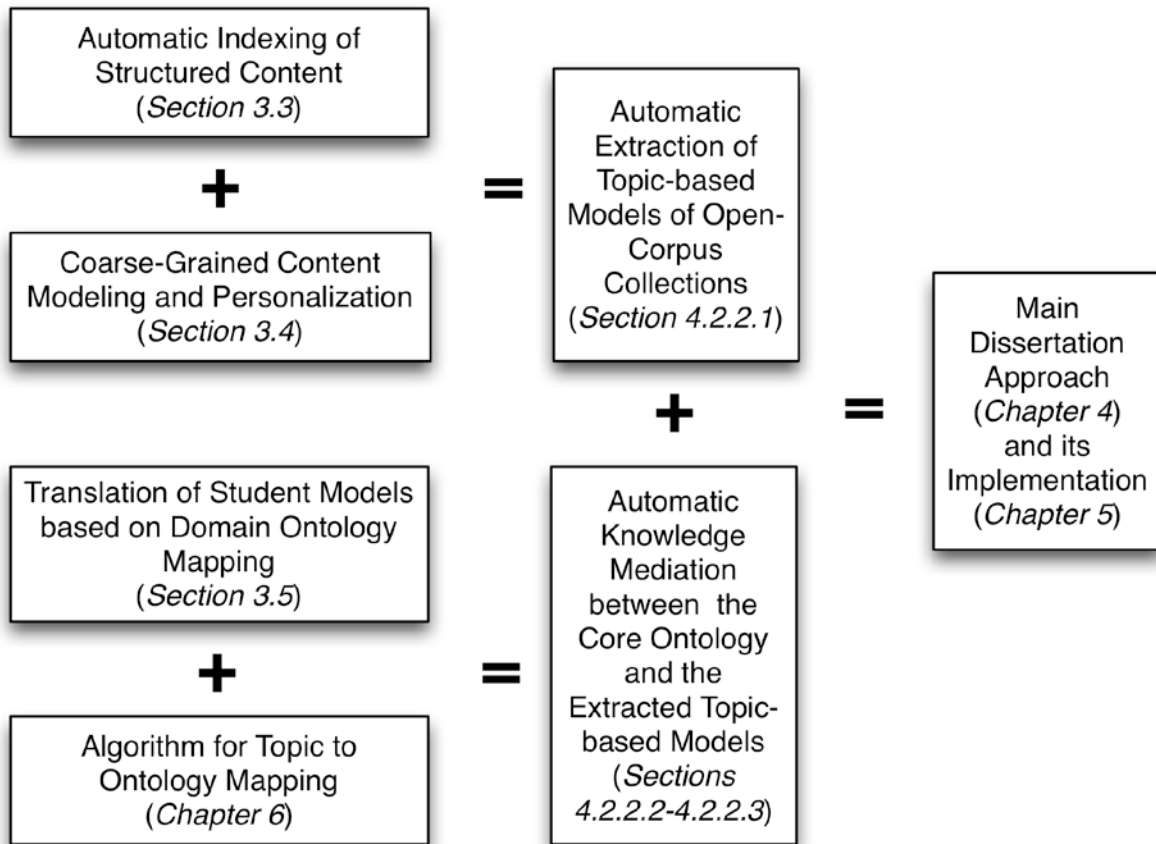


Figure 4: Main dissertation approach and its relations to other contributions of this dissertation

1.3 RESEARCH QUESTIONS

This dissertation focuses on the topics of open-corpus personalization and open-corpus content modeling. Special attention is paid to the following research questions (each question is accompanied by a brief overview of how it was addressed in the dissertation):

1. *What are the principal ways to implement semantic modeling of open-corpus content and what are their pros and cons?*

The three primary solutions to the problem of semantic open-corpus content modeling are presented in Chapter 3.0 together with the author's projects implementing these solutions. Their analysis helped to identify the weak points of these solutions, as well as the synergy between them. Based on this synergy the road map to the main dissertation approach has been built.

2. *What is the potential source of domain knowledge in Web-content, and how this knowledge can be automatically extracted and used for modeling open-corpus content on a large scale and across multiple domains?*

One of the central ideas of this dissertation is the reuse of topic-based structures of high-quality collections of educational content available online. These structures implicitly represent organization of knowledge in the domain by the author of the collection. A topic-based model, automatically extracted from such organization will result in a coarse-grained representation of the domain and the content of the collection.

3. *What techniques and sources of information can and should a mapping algorithm utilize in order to support mapping between ontologies and topic-based models?*

The developed mapping algorithm uses four dimensions of information about mapped models and implements four corresponding individual mappers. The name-based and the

instance-based mappers compute two separate similarity tables based on the naming labels of concepts and topics and the instance corpora associated with them. The structure-based mapper merges the tables and augments similarities with neighborhood-based information. Finally, the order-based mapper uses the instructional sequencing of concepts within the central system and topics within the textbook to refine the final similarity table.

4. *Can the main dissertation approach based on automatic extraction of topic-based models and automatic mapping of these models into the domain ontology be used to support open-corpus personalization for e-Learning?*

The main dissertation approach has been implemented as an e-Learning service for recommending the supplementary reading material to students solving self-assessment exercises. Java programming language has been chosen as a subject domain. The reading material comes from an electronic version of a Java textbook. The content of the textbook has been automatically processed; the topic model has been extracted and mapped to the central Java ontology.

5. *Will the implementation of the main dissertation approach be able to contribute to students' leaning?*

Evaluation of the developed service has been organized as a controlled experiment. The results of the pre- and post-tests have demonstrated that when using the open-corpus version of the system students have significantly improved their knowledge in the domain.

6. *What will be the main effects of the designed personalization service?*

Besides the general learning effect, adaptive recommendation of open-corpus reading content contributed to significant learning of complex learning material and theoretical

learning material. Weaker students have also benefitted more from the system equipped with the open-corpus recommender.

7. *Will the designed service be able to maintain the adaptation of comparable quality to the conventional closed-corpus approach?*

Comparison of open-corpus and closed-corpus versions of the system has shown that for every parameter characterizing the quality of recommendation, there was no significant difference between these two versions of the system.

1.4 DEFINITIONS

Definition 1. Adaptive educational system

An adaptive educational system (AES) is a computer application that helps students to perform learning activity in an adaptive way by adjusting the content of learning objects, their sequence, or presentation according to the individual characteristics of a student.

Definition 2. Closed-corpus adaptive system (adapted from (Brusilovsky and Henze 2007))

A closed-corpus adaptive system is an adaptive system, which operates on a limited corpus of documents, where documents, relations between the documents, and relations between the documents and the knowledge models of the system are defined at design time.

Definition 3. Open-corpus adaptive system (adapted from (Brusilovsky and Henze 2007))

An open-corpus adaptive system is an adaptive system, which operates on an unrestricted corpus of documents, where documents, relations between the documents, and relations between the documents and the knowledge models of the system are not known at design time and, moreover, can constantly change and expand.

Definition 4. Ontology (adapted from (Gruber 1995; Guarino 1998))

Ontology is a formal and explicit specification of a shared and intentional conceptualization.

In the case of AES, ontologies primarily serve as domain models; they specify the structure of the subject domain, formally represent its main entities and the relations between them.

Definition 5. Ontology mapping (adapted from (Euzenat and Shvaiko 2007))

Ontology mapping is the process of finding relationships or correspondences between entities of different ontologies.

Definition 6. Concept

Concept is an element of domain semantics. It helps to objectively and cohesively represent a single notion in the domain.

Definition 7. Topic

Topic is an element of content structuring. It helps to organize the content into a logical sequence or hierarchy of units.

1.5 DISSERTATION OUTLINE

The reminder of this dissertation is organized as follows. Chapter 2.0 reviews the existing work in the three research fields related to the theme of this thesis. It starts by outlining the main technologies of open-corpus personalization. It continues with a summary of the most important methods of ontology mapping from the point of kinds of information they use. Finally, it presents the state-of-the art in the emerging field of ontology-based personalization and concludes with a brief analysis of how this thesis advances all of the three relevant fields.

Chapter 3.0 introduces the first set of results of this dissertation. It provides a broad and detailed exploration of open-corpus content modeling. It analyzes three principle approaches to this problem: automatic indexing of structured content, relaxing requirements to domain modeling, and utilization of external models. All three approaches are illustrated with concrete technologies developed by the author. The chapter concludes with a comprehensive summary and a discussion bridging the first set of attempted technologies with the main dissertation approach.

Chapter 4.0 gives a conceptual explanation of the main dissertation approach to open-corpus ontology-based personalization for e-Learning. It starts with discussing the main ideas, principles and assumptions behinds the approach. It continues with the description of the content modeling and personalization workflow. It discusses the most important benefits of the approach that significantly broaden the area of its applicability from the points of target domain, implemented adaptation technology, chosen content collection, etc. In conclusion, it reflects once more on the lessons learnt from analyzing open-corpus content modeling and demonstrates how the main dissertation approach summarizes and advances the first set of technologies described in Chapter 3.0.

Chapter 5.0 presents the proof-of-concept implementation of the proposed approach in the form of Ontology-based Open-corpus Personalization Service (OOPS). It explains the technical details of the OOPS service, including its architecture and interface design, as well as used technologies for student modeling and adaptation. It describes the instructional settings and the domain, in which OOPS operates, including the core ontology representing the domain and the topic-based model extracted from the target content collection.

Chapter 6.0 introduces another important contribution of this dissertation – the ATOM mapping algorithm. It starts with explaining some features of topic-based models that make the mapping task of ATOM rather unique. The four following sections describe the four individual mappers constituting the algorithm. Every mapper is based on an individual source of information from the mapped models: naming labels of concept and topics, their instance corpora, structural relations within the models, and ordering of model elements. The chapter concludes with a brief summary of the main contributions ATOM makes to the field of ontology mapping.

Chapter 7.0 presents the design and results of the main dissertation study evaluating the effectiveness of OOPS. The study is organized as a controlled experiment comparing the open-corpus version of system with two other versions. The chapter presents several significant effects of OOPS on students' learning, the analysis of quality of open-corpus recommendation produced by OOPS, and the results of subject's evaluation of the service.

Chapter 8.0 summarizes the main results and contributions of this dissertation categorized by research fields. It also discussed limitations of the proposed approach and some questions left unanswered. The chapter concludes this dissertation with the description of potential directions for future work.

A list of appendices contains supplementary material, including texts of tests and questionnaires, as well as short descriptions of several systems used in the main study.

Figure 5 graphically represents the outline of this dissertation including the dependencies between the chapters and their parts and can help to choose the possible order of reading the dissertation.

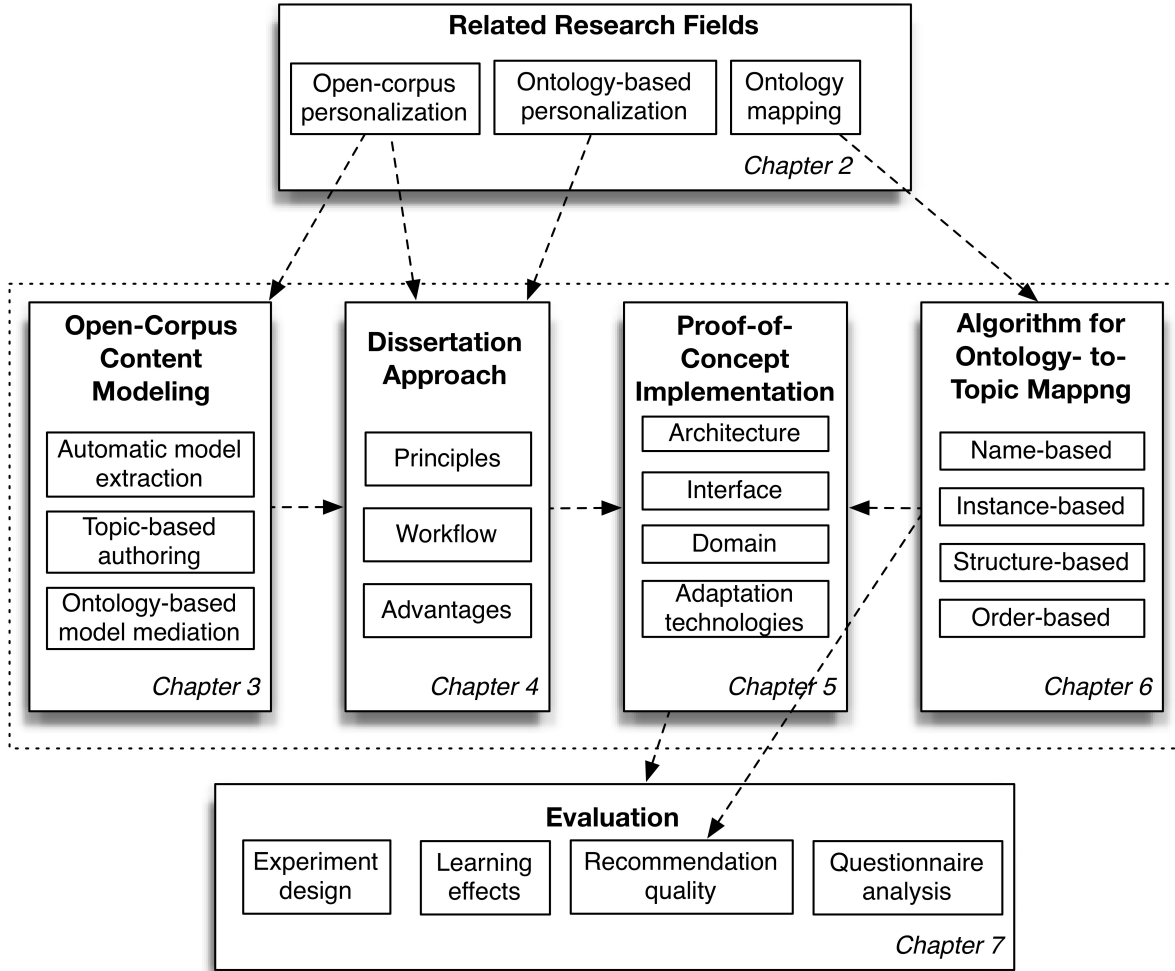


Figure 5: Organization of the main dissertation chapters and relations between them

2.0 OVERVIEW OF THE RELATED RESEARCH FIELDS

2.1 INTRODUCTION

The topic of this dissertation lies at the intersection of the three emerging research fields: open-corpus personalization, ontology mapping and ontology-based personalization (see Figure 6). This chapter provides an analytical overview of these fields. When analyzing open-corpus personalization and ontology-based personalization approaches, the main focus is made on the applications for e-Learning to emphasize their relevance to this dissertation. The field of ontology mapping is reviewed from the point of information integration and the sources of information available for mapping algorithms, to facilitate the description of the developed mapping algorithm.

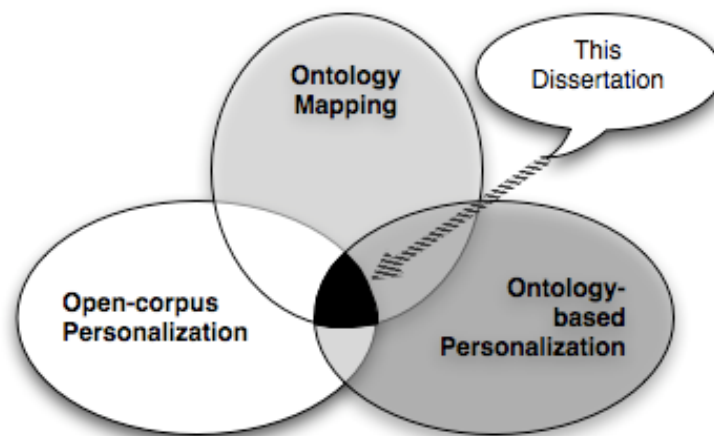


Figure 6: Research fields contributing to this dissertation

2.2 OPEN-CORPUS PERSONALIZATION FOR E-LEARNING

This section outlines the range of technologies that have been used for implementing open-corpus AES. They differ from each other in many respects, including the kinds of adaptation techniques they can support, and the source of knowledge they employ in order to model open-corpus content.

Table 1 summarizes these technologies. For every technology, it indicates the source of knowledge used for modeling content elements and users' characteristics, and gives examples of adaptive systems implementing it. The table also lists the strong and weak features of all technologies from the point of applicability in AES.

Figure 7 provides another perspective on the spectrum of existing open-corpus personalization technologies: it classifies them into two groups based on the type of information used to model open-corpus content. Collaborative filtering and social navigation form the user-based group, as they extract their models from the user activity. The remaining five technologies are combined into content-based group, as the modeling approaches employed here are content-driven.

Table 1. Comparison of open-corpus personalization technologies

Technology	Source of Knowledge	Systems	Pros	Cons
Extensible adaptive content	Domain ontology provided by the author of the system is used to manually annotate the content	<i>KBS</i> <i>Hyperbook</i> , <i>SIGUE</i>	High level of semantics, high quality of adaptation	Manual, therefore cost-effective and laborious
Term-based content modeling	Term-vectors are automatically extracted from the content	<i>Letizia</i> , <i>WebMate</i> , <i>NewsDude</i> , <i>MLTutor</i>	Fully automatic	Low level of semantics, low quality of adaptation to users' knowledge
Collaborative filtering	A table of similarities is computed based on users' behavior with the content (ratings, visits, etc.)	(Zaiane 2002), (Tang and McCalla 2003)	Fully automatic	"Cold start" of new users / content items, sparsity of similarity tables, inability to model idiosyncratic characteristics of the users
Social navigation	"Wisdom of the crowd": content items are ranked by the amount of users' activity (visits, annotation, etc.)	<i>Knowledge Sea II</i> , <i>CoWeb</i>	Fully-automatic	"Snowball effect", "Cold start" of new items, inability to guide users to the right content item at the right moment of time
Semantic annotation	Elements of domain ontology are matched to formatted fragments in the content based on a set of predefined patterns	<i>Magpie</i> , <i>COHSE</i>	High level of semantics, fully or semi-automatic	Recognizes only limited number of patterns, hence only limited number of concepts can be identified automatically
Information Extraction	Terms extracted from the content and their occurrences, labels and descriptions of existing concepts	(Li and Zhong 2006), (Roy, Sarkar et al. 2008), <i>Tangram</i>	High level of semantics, fully or semi-automatic	Fully automatic approaches have low precision, semi-automatic approaches require human author's participation
Semantic tagging	Domain ontology is used for restrictive tagging; or concepts of the ontology mapped to free tags	<i>CommonFolks</i> , <i>Relco</i> , <i>OATS</i>	High level of semantics, fully or semi-automatic	Either requires participation of taggers, or produces mapping of low precision

All technologies are roughly arranged along the two dimensions measuring their weak and strong points. *Applicability in e-Learning* estimates how broadly the technology can support different usage scenarios, subject domains, adaptation strategies, etc. *Semantics* versus *Automation* axis characterizes the extracted models of open-corpus content and the modeling process itself. For example, the *Extensible Content* approach supports the creation of semantic content models that can be widely applied in AES, but the models have to be created manually. On the other side, the Term-based approach is automated, but the resulting models do not capture the semantics of the domain, cannot adequately support modeling of students' knowledge and thus have a very limited applicability in AES. This figure is not intended to define the exact coordinates for all technologies in such a space, but rather to roughly estimate their relative positions.

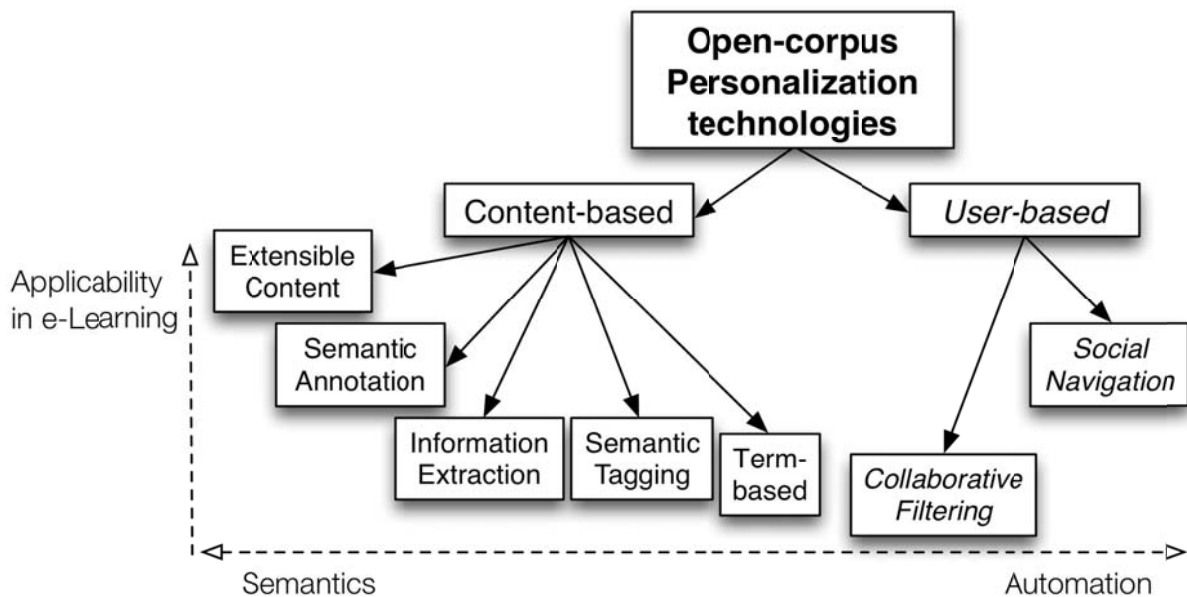


Figure 7: Classification of open-corpus personalization technologies

2.2.1 Extensible adaptive content

One of the first attempts to build an open-corpus adaptive system was made in *KBS Hyperbook* projects (Henze and Nejd1 2001; Henze and Nejd1 2002). The authors recognized the inability of adaptive systems to personalize the external content and proposed a solution based on the strict separation of the adaptive functionality from the domain model and the content it describes. The system relies on the representation of domain models as ontologies, which allows authors to employ a standard set of inference rules for building the adaptive component of the system. This ensures the compatibility of the system's logic with any potential domain model as long as its representation followed the same ontological requirements. For any new content added to the system, a new content model could be created. If an author wanted to update an existing document space, s/he could simply modify its content model by adding semantic markup for the new documents. The authors implemented this approach for developing an educational hypermedia system for learning Java programming.

A similar approach is taken in *SIGUE* (Carmona, Bueno et al. 2006), which is a system for authoring and delivery of adaptive hypermedia content. *SIGUE* allows teachers to compose their own adaptive courses from the existing learning material they find online. Instead of designing their own material, thanks to the open architecture of the system, any author can plug into *SIGUE* tutorials pages, Web-based examples, visualizations, etc., define the domain model of the course and annotate the learning material with the concepts of the model. *SIGUE* then will present the course material to the students adaptively, based on their progress in the course.

Figure 8 visualizes this approach. The main contribution of systems like *KBS Hyperbook* and *SIGUE* is the ability to extend an existing adaptive system with novel content. However, they provide no functionality for automatic authoring support. The human expert still had to

information interests or needs as vectors of terms extracted from the documents that the user has browsed or requested (sometimes such vectors were enriched with *tf*idf* values), e.g. *Letizia* (Lieberman 1995), *WebMate* (Chen and Sycara 1998), *NewsDude* (Billsus and Pazzani 1999), etc.

A big advantage of this approach is the automatic modeling of content based on well-developed information retrieval techniques, which provides the basis for open-corpus personalization. However, the terms support only shallow content models. Term-based modeling is not able to represent the true meaning of the content. Instead, it relies on statistical regularities within the text and implements the framework for retrieving statistically close documents. It provides no reliable means to address homonymy (multiple meanings of a word), or synonymy (multiple words expressing the same meaning).

In the context of e-Learning this issue becomes even more important. The traditional building block of domain models used by adaptive educational applications is a concept, which allows estimating student's knowledge on a fine-grained level and adapting to it accordingly. Every concept deterministically and unambiguously models a certain piece of domain semantics. A term itself does not represent anything except its lexical label and the frequency of its occurrences in the content. Term-based reasoning assumes that documents with similar term maps have similar meaning.

A number of successful heuristics have been developed to facilitate term-based models and make the term-based inference more reliable; however, they cannot guarantee a high precision. This is crucial in educational settings. If, for instance, an adaptive news recommender system presents a user with a relevant news item in 70% of the cases, it can be considered good enough. However, guiding a student to irrelevant learning material three out of ten times is not

an effective learning strategy. One of the few examples that attempted to apply term-based automatic content indexing for e-Learning is *MLTutor* (Smith and Blandford 2003). This system recommends hypermedia resources based on learners' interests harvested from the term vectors of the documents visited before. Although, *MLTutor* is not an open-corpus system, the implemented content authoring is fully automatic and hence provides the basis for adding new resources with no intervention of a human expert.

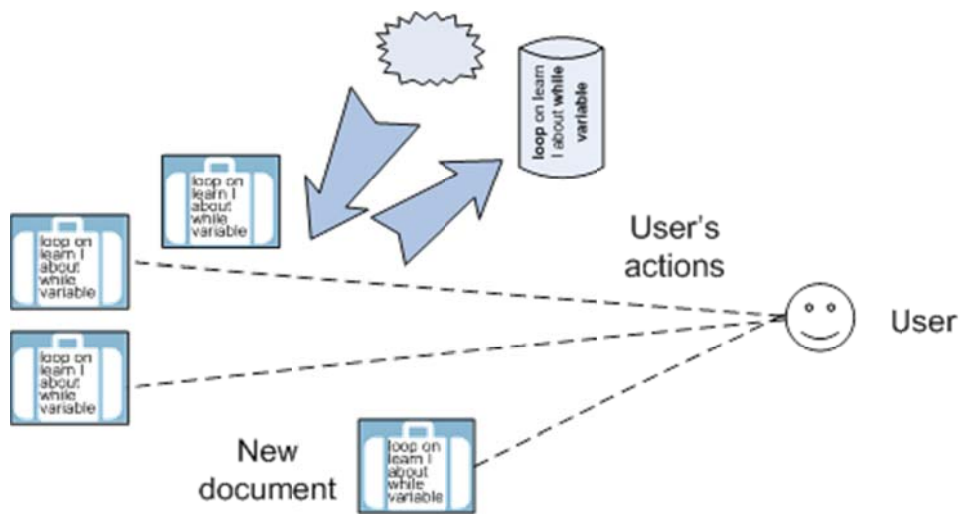


Figure 9: Term-based content modeling: new documents are automatically modeled as “bags of words”

2.2.3 Collaborative filtering

A very popular class of Web-based adaptive systems exploits a principally different set of technologies than the traditional content models. They are based on the users themselves, the similarities between users and the patterns of social behavior they follow on the Web. The most popular of them is collaborative filtering (CF), which is widely used in Web recommender systems. Essentially, all content-based approaches rely on the similarity of documents expressed

in their content metadata; content-based modeling represents a user as a product of her/his individual history of rating, purchases and page accesses. CF looks for the like-minded users expressing similar behavior (purchasing the same products, rating items similarly, etc.). For open-corpus adaptation, CF offers a potential solution (Figure 10). It does not require the authoring and support of expensive semantic content models to maintain the adaptation, however it has its own problems. Collaborative approaches require extensive offline data processing. A new document added to the system cannot be processed by the CF algorithm, since there is no activity registered for the item that could trigger personalization. A similar situation happens when a new user comes to the system. It will not be able to adapt to her/him without a record of activity needed to find similar users. The quality of CF drops if the user-item matrix is sparse, which is common in the Web settings and especially important in the context of e-Learning. Finally, collaborative systems provide an individual user with recommendations based on the evidence collected from other users, while learning is an individual process in many respects. The fact that a certain student accessed the learning resources in a particular order or answered a problem correctly only after two incorrect attempts does not imply that other students will follow the same pattern.

These reasons explain, why the success of CF in the field of adaptive recommenders did not transfer to the field of personalized e-Learning. Yet, a few attempts have been made to develop CF-based AES. (Zaiane 2002) has developed the *e-Learning Task Recommender* that combines CF and association rule mining in order to suggest learning actions to students. It analyses the log of pages visited by a student, looks for a similar pattern among other students' logs and recommends the next best task from a matching log. (Tang and McCalla 2003) have developed another educational recommender system that implements another modification of CF

approach. Their system recommends research papers to senior students taking a seminar course. The papers are crawled from the *CiteSeer*² system and classified based on their keywords / research topics, type (review, technical, etc.), length, etc. Once a student reads a paper s/he rates it, thus populating her/his student profile. The profiles of new students are matched to the existing ones based on rating similarity. The metadata extracted from the papers is used to constrain the search space for similar students (e.g. if a student has demonstrated interest in reading papers about user modeling, the system will try to match his profile to students with the same interests).

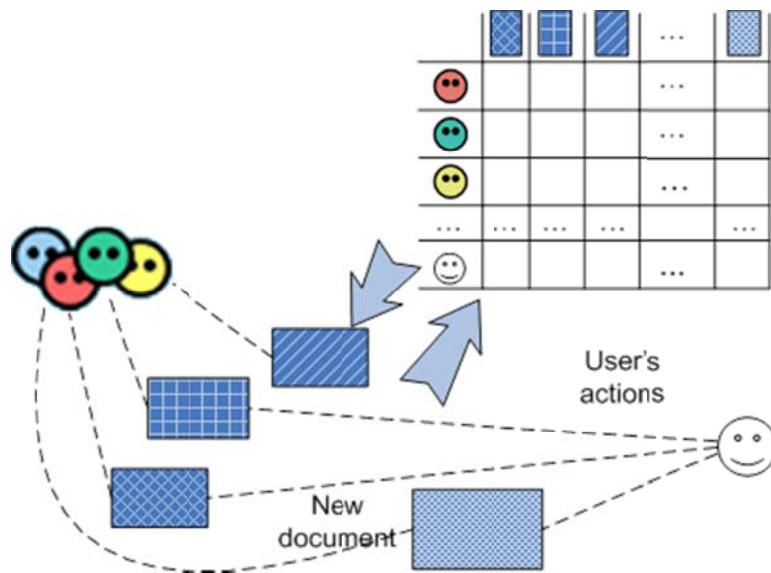


Figure 10: Collaborative filtering: documents are modeled based on the similarity of users' experience with them

2.2.4 Social navigation

Social navigation is another user-based personalization strategy. It exploits the natural behavioral regularity of human users to follow the crowd. If a certain resource attracts a lot of attention from

² <http://citeseer.ist.psu.edu/>

many users, this is regarded as an indicator of its quality. Should the system display such information to its users, they will tend to follow the footprints of each other and access the popular resources. A big advantage of this approach is that it requires no knowledge about the content of the documents. No additional models need to be created, which greatly facilitates the implementation of social navigation in open-corpus settings (Figure 11).

An example of social navigation system for e-Learning is *Knowledge Sea II* (Brusilovsky, Chavan et al. 2004). It provides students with navigation support for the reading material from several online tutorials. Tutorial pages are automatically clustered into the cells of a dynamic map according to their term vectors. *Knowledge Sea II* paints the map cells with different degrees of blue color based on the amount of traffic students have generated for the resources assigned to the cell. *Knowledge Sea II* also represents the individual traffic of a student by a small human colored with the same palette. It allows students to compare their own progress with the progress of the rest of the class and focus on the resources where they lag behind. In addition to traffic information, *Knowledge Sea II* allows students to annotate the tutorial pages and uses this information as an additional social navigation cue.

CoWeb is another application employing the social navigation technique (Dieberger and Guzdial 2002) to facilitate collaborative content creation. It has been applied in several domains, including e-Learning. *CoWeb* navigates its users through the wiki-based hyperspace by annotating links to the documents with social cues. These cues present three kinds of information: how recently was the document updated, how recently was it accessed and how much activity has the document received overall. In addition, it provides special markers for the new documents and the documents that haven't been accessed for a long time. These special markers try to remedy one important problem of social navigation – the so-called “snowball

effect”. Once an information resource receives some initial attendance, traffic-based navigation will drag more people to it hence increasing its traffic. Even if it is not a very good resource, users will come following the majority. Another way to solve this problem is by implementing more reliable mechanisms for indicating popularity of the resource. Annotations and rating have been suggested as possible solutions. Although they do allow better estimation of the resource quality, they also require bigger involvement and higher level of participation from the users of the system.

The most important problem of systems like *Knowledge Sea II* and *CoWeb*, and social navigation approach in general is that it has no mechanism of guiding a student to the *right* resource at the *right* moment. Essentially, social navigation allows differentiating between good and bad resources but it does not help a student to come to the most appropriate resource. As the other community-based approaches, social navigation is not capable of recognizing the idiosyncrasies of an individual user and adapt to them accordingly. The community wisdom works perfectly, when the task of a user is to buy a certain product or to watch a video on *YouTube*³; however, in educational settings, the individual differences between the users become more important.



Figure 11: Social navigation: documents are modeled based on the history of user activity

³ <http://www.youtube.com>

2.2.5 Semantic annotation

Another approach towards the problem of open-corpus content modeling is based on the Semantic Web (W3C 2001) technologies for knowledge discovery. Several recent projects exploit the ontology-based annotation (or semantic annotation) techniques for automatic indexing of textual Web-resources with semantic metadata. One of them is *COHSE* (Conceptual Open Hypermedia Service), the joint project between Sun Microsystems and University of Manchester (Bechhofer, Goble et al. 2003; Yesilada, Bechhofer et al. 2007). *COHSE* is based on the original idea of distributed link services (Carr, DeRoure et al. 1995) working as intermediaries between Web clients and Web servers and augmenting Web documents with dynamic links. Architecturally *COHSE* works as a proxy. It intercepts a client's HTTP request, retrieves the original HTML document, enriches its content with new links and delivers the modified document to the user's browser. The extra links added by *COHSE* are based on the contents of both the original source document and the target document. They are placed in the way to connect the key pieces of text in the original document with relevant HTML context elsewhere. Such knowledge-driven linkage is performed on the basis of an ontology serving as a source of consensual domain semantics. The *COHSE* components apply ontology-based annotation technologies to associate automatically the pieces of documents with ontology concepts. When a document is requested, its annotations act as links to the concepts and then to other documents annotated with these (or related) concepts.

A similar approach is implemented in *Magpie* (Domingue, Dzbor et al. 2004; Dzbor and Motta 2007). Unlike *COHSE*, *Magpie* works on the client side as a browser plug-in. It analyses the content of the HTML document being browsed on-the-fly and automatically annotates it based on a set of categories from an ontology. The resulting semantic mark-up connects

document terms to ontology-based information and navigates the user to the content describing these terms. For example, if *Magpie* recognizes that a particular phrase is a title of a project, it populates a menu with links to the projects details, research area, publications, members etc.

The main drawback of this approach is its reliance on a limited number of natural language patterns and heuristics that are often situational. Modern ontology-based text annotation techniques are capable of producing very good results in recognizing named entities, such as places (countries, cities), people, organizations etc. They also implement procedures for recognizing pieces of content with very specific formatting, like dates, times, coordinates, and citations. Annotation of very general content in terms of an arbitrary ontology may create a challenge for these tools. In the context of e-Learning, such limitations can satisfy a very small number of domains and types of content. Figure 12 presents a diagram explaining a general framework of semantic annotation.

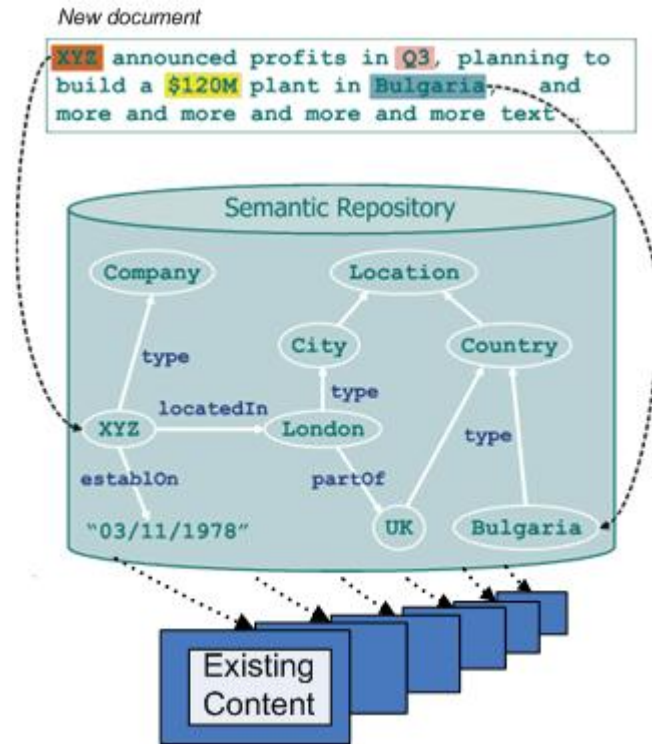


Figure 12: Semantic annotation: pieces of new documents are automatically associated with the ontology concepts⁴

Application of ontology-based annotation for open-corpus personalization is a promising direction of research. Combination of the capabilities of systems like *COHSE* and *Magpie* with some architectural solutions proposed by *KBS Hyperbook*, provides the functionality necessary for open-corpus personalization. Some work in this direction has been done recently for the task of adaptive recommendation. Systems like *Quickstep* (Middleton, De Roure et al. 2001), *Foxtrot* (Middleton, Shadbolt et al. 2003) and *MyPlanet* (Kalfoglou, Domingue et al. 2001) apply ontology-based annotation to automatically index dynamic textual content and recommend it adaptively.

⁴ A part of this figure has been taken from Popov, B., A. Kiryakov, et al. (2004). "KIM – a semantic platform for information extraction and retrieval." *Natural Language Engineering* **10**: 375-392.

2.2.6 Information extraction

As the amount of information available on the Web has grown, a new research field has emerged, that applied various computation methods in order to automatically harvest structured knowledge from the unstructured or semi-structured Web content. Unlike Information Filtering and Information Retrieval, Information Extraction (IE), first, seeks to discover the hidden semantics within the Web resources and then uses this new layer to deliver information based on its meaning rather than mere term vectors from the documents' text (Cowie and Lehnert 1996). It is sometimes hard to draw a line between the IE projects and those of semantic annotation, and term-based content modeling. Many methods of information extraction have been borrowed from these fields, as well as natural language processing, text classification, and data mining. Another area of research that is very close to IE in its goals and methodology is ontology learning (Shamsfard and Barforoush 2003); it also aims at automatic and semiautomatic acquisition of structured knowledge (ontologies) from less-structured information sources.

From the open-corpus personalization point of view, IE provides a complete framework for the creation of content models that can be employed for adaptive presentation/navigation/recommendation of the content. Several projects rely on IE for implementing adaptive systems; and, even though the methods they used, and the types of content they personalize are very different, the main idea stays the same – extraction of structured data from unstructured/semi-structured content.

(Li and Zhong 2006) proposed an algorithm that mines the content of documents that user has found interesting in order to learn “conceptual” representation of her/his information preferences. The term vectors harvested from documents are analyzed; common patterns are discovered and form concept lattices. Then a set of heuristics is applied in order to build a

hierarchy of discovered concepts. The resulting model is claimed to ontologically describe information needs of the user. However, the actual level of semantics achieved by such algorithms is not far from the term-based models. Supplied with a thesaurus and/or lexical ontology, such as *WordNet* (Miller, Beckwith et al. 1990), they can achieve a better precision, but still will be mainly based on the distribution of terms in the corpus. It is a proper model to represent users' interests or information needs, but it is not fit for describing students' knowledge.

(Roy, Sarkar et al. 2008) proposed a more elaborate approach for extracting semantic metadata from learning content. They develop a three-layer content model, with keywords (terms) being the lowest layer, ontology concepts forming the second layer, and course topics representing the third layer. On all three layers, the corresponding modeling elements are specified (this includes the dictionary of terms), and relations within a layer and between the elements of different layers are also defined. Hence, for example, for every concept they pre-author a set of corresponding keywords (with their weights). This architecture allows for very precise automatic matching of new documents to the modeling elements of all three layers. However, it requires a significant initial authoring effort, especially, for connecting keyword layer and the concept layer.

Tangram system applies a semi-automatic approach for providing semantic metadata for the parts of learning objects, namely *PowerPoint* slides (Jovanović, Gasevic et al. 2006). The metadata extraction starts with a teacher manually annotating the *PowerPoint* presentation file with some concepts from the domain ontology. After that, *Tangram* takes over and generates annotations for every individual slide within the presentation. It uses relations within the ontology, term vectors extracted from the slides and concept labels and descriptions, in order to

match slides to appropriate concept. The quality of annotation produced by *Tangram* depends on a very particular type of content. Presentation slides are fairly small and very condensed information items. They always have headers, very particular formatting and they are authored to consist only the most important text. Nevertheless, *Tangram* implements an interesting approach capable to produce good results and requiring minimal authors' participation.

2.2.7 Semantic tagging

Rise of the Social Web technologies has led to the emergence to multiple forms of user-generated content. One of them – social tagging – has become a powerful and engaging tool for regular Web-user to produce unstructured metadata for online resources they discover or create.

What makes social tagging so popular is:

- non-restrictive nature of tags: when annotating the resource with a tag a user is not constrained in terms of vocabulary or format; s/he does not have to learn a new schema or a set of rules and can create tags that represent individual meaning of this resource to this user;
- immediate, clear and powerful incentives: annotated resources can be retrieved and browsed by tags, shared tags and help access resources added by different users, tag-based interfaces (such as tag clouds) further facilitate resource discovery.

However such flexibility and popularity of tags as a tool for metadata authoring comes with a price. Tag-based content models bear most of the problems that a term-model would have. Except for tag concurrence there us no structural relationships between the tags, same tags can have multiple meanings (polysemy), and the same meaning can be expressed by various tags (synonymy). In addition, tag models are often subjective – users choose tags for themselves

disregarding if they have no meaning for the rest of the community. Yet, tagging have proven to be an effective technology for soliciting annotations from the users of social Web-sites; and several research projects tried to remedy the problems of tag-based metadata in order to make it more suitable for using in adaptive systems.

First approach towards improving tag-based annotations is based on restricting tag choices, discouraging introduction of new tags and promoting the reuse of existing tags and/or semantic labels from a predefined vocabulary. Open Annotation and Tagging System (*OATS*) implements such paradigm for e-Learning domain (Bateman, Farzan et al. 2006). *OATS* can work as a supplementary plug-in of any learning content provider. It allows students to annotate and tag interesting text fragments, thus providing basis for social navigation and/or collaborative content annotation. The tagging interface of *OATS* encourages students to reuse the tags introduced by others, hence the trying to harmonize the resulting annotations.

The second approach toward semantic tagging shifts the responsibility of providing the semantics from the users to the system. Systems implementing this approach accept any kind of tags from the users, and then apply various techniques in order to infer the hidden semantic layer from the tag-based annotations, often with the help of external ontologies. Two of the many examples of such systems are *CommonFolks* (Bateman, Brooks et al. 2006) and *Relco* (van der Sluijs and Houben 2009). *CommonFolks* allows authors of learning content to provide tagging-like annotation of created learning objects. Once the annotation is done, *CommonFolks* employs the lexical ontology *WordNet* in order to determine the actual meaning of tags and organize them into structured metadata. *Relco* uses a similar solution to obtain semantic annotation of multimedia documents. The initial input to *Relco* is a set of tags describing an information object. The solution proposed by the authors of *Relco* is to match these tags to the concepts of

the pre-existing domain ontology. The resulting infrastructure tries to combine the best of two worlds: the engraining interface of social tagging and the rich semantics of formal ontologies.

Semantic tagging is a promising emerging technology that can be helpful in addressing the problem of semantic metadata generation. However, its applicability is rather situational, especially in the field of e-Learning. It relies on active user involvement in tagging learning content, which will require a high level of motivation and/or an effective system of incentives. The quality of semantic tagging algorithms heavily depends on the amount and quality of tags, as well as the volume of the user base. This is a rather challenging requirement for e-Learning systems.

2.3 ONTOLOGY MAPPING

Ontologies are intended to represent consensual knowledge in the domain of discourse. Ideally, a domain ontology should be discussed and verified by multiple communities operating in the domain. Unfortunately, such a level of agreement is seldom achievable on the Web. Most often neighbor communities differ in perspectives on the same domain to such extent, that the common ontological commitment among all interested parties becomes impossible⁵.

As a result, for many domains there already exist numerous ontologies, and these numbers are only going to grow. The amount of overlapping ontologies (ontologies modeling the same set of concepts) is even larger. For example, the *Swoogle* search engine (Ding, Finin et al.

⁵ The rare examples are Health Care and Life Sciences Semantic Web Initiative W3C. (2007). "Semantic Web Health Care and Life Sciences Interest Group (HCLSIG)." from <http://esw.w3.org/topic/SemanticWebForLifeSciences>. and Gene Ontology Ashburner, M., C. A. Ball, et al. (2000). "Gene Ontology: tool for the unification of biology." *Nature Genetics* 25(1): 25-29..

2004) currently retrieves 867 RDF-documents referring to the concept “*student*”⁶. Some of these documents are ontologies defining their own versions of the concept “*student*”, reflecting different viewpoints, but still modeling the same notion. On top of it, not all ontologies use the label “student” to determine the concept, some refer to it as “*learner*” (68 ontologies retrieved by *Swoogle*), “*pupil*” (36 ontologies), “*trainee*” (8 ontologies), etc.

To find the correspondence between identical and relevant concepts defined in different ontologies multiple techniques for semantic integration of ontologies have been developed. These techniques are combined under the term “ontology mapping”⁷ (Kalfoglou and Schorelmmmer 2003; Noy 2004)).

Ontologies are very rich models, and as such they can provide mapping algorithms with multiple source of information. Figure 13 summarizes these sources. Concepts of ontologies *A* and *B* can be mapped based on:

- the external references to the upper ontology,
- structural similarities,
- matching textual labels,
- equivalent instances.

⁶ The query has been tossed on 08/02/2011

⁷ Terms “ontology alignment”, “ontology merging”, and “ontology matching” are sometimes used as synonyms to “ontology mapping”.

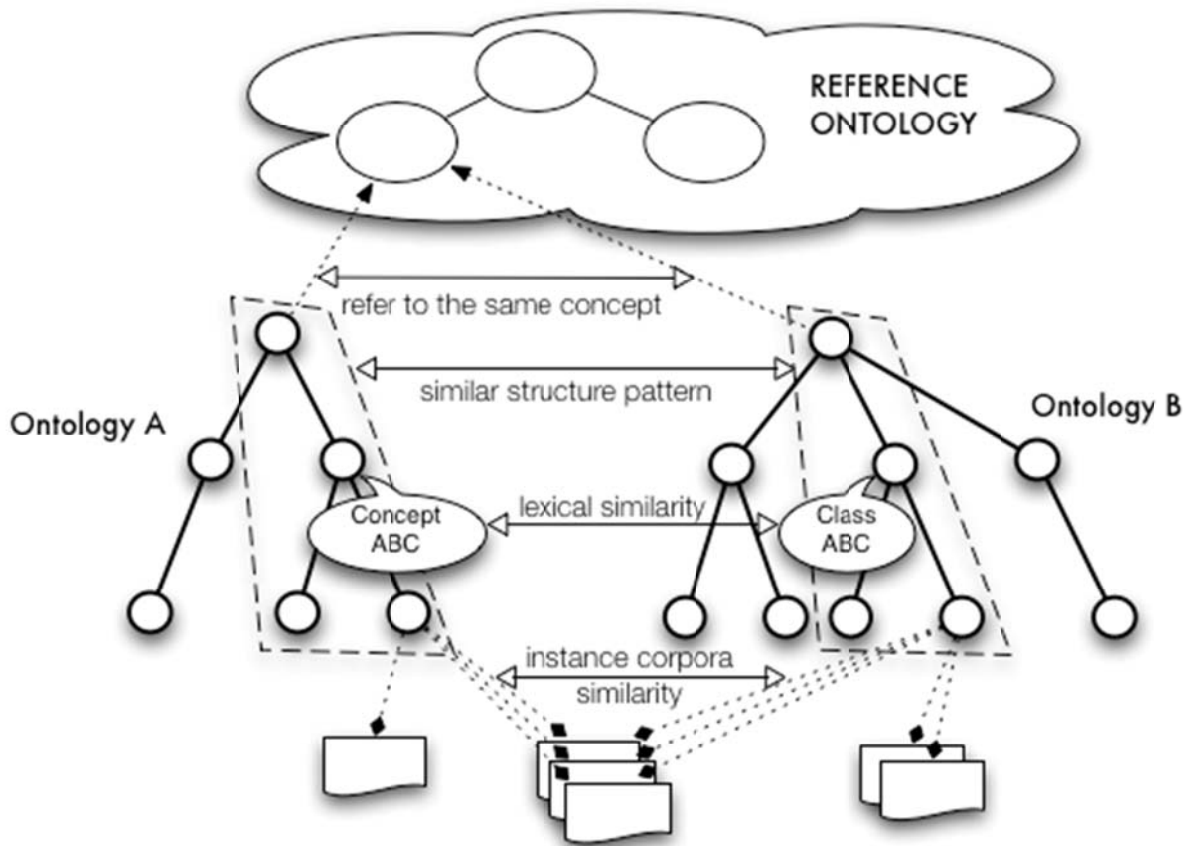


Figure 13: Sources of information used by ontology mapping algorithms

Next four sections give the details of these mapping techniques with the examples of system implementing them.

2.3.1 Ontology mapping based on a reference ontology

In a limited number of cases, the ontologies being mapped can contain references to a common ontology. Most often, the common ontology models very general notions of nature, such as time, space, process etc. Several initiatives to create general ontologies exist:

- *SUMO* (Suggested Upper Merged Ontology) (Niles and Pease 2001) has been developed as an IEEE standard for abstract and upper level knowledge modeling;
- *DOLCHE* (Descriptive Ontology for Linguistic and Cognitive Engineering) (Gangemi, Guarino et al. 2002) is formal upper-level ontology capturing linguistic categories and human common sense developed in the framework of European project WonderWeb;
- *CYC* (Lenat 1995) is a an proprietary project formalizing an upper-level foundation ontology and a set of domain “micro-theories”;
- *WordNet* (Miller, Beckwith et al. 1990) is a lexical ontology defining natural language terms and relations between them, such as synonymy, polysemy, antonymy, hyponymy, etc.

The practical purpose of a general ontology is to be extended by domain-specific ontologies. When the concepts of two ontologies are connected to the same or related concepts of the upper-level ontology, such references become a valuable source for the identification of correspondences between these concepts. For example, if two ontologies have their concepts “*trainee*” and “*student*”, correspondingly, linked with “*sameAs*” relations to the single concept “*learner*” of the upper ontology, it provides strong evidence, that these two concepts of different ontologies actually model the same entity in the world.

2.3.2 Ontology mapping based on lexical information

The most important kind of information for ontology mapping comes from the lexical labels given to the elements of the ontologies by their developers. Well-designed ontologies contain also natural language comments and definitions explaining the meaning of classes, relations and axioms constituting the ontology. Ontology developers try to maintain this lexical information

meaningful and consistent across the ontology, therefore lexically similarity of two elements usually indicates existence of semantic similarity. There are several techniques developed for facilitating lexical analysis of ontological vocabularies:

- Analyzed tokens undergo the process of string normalization:
 - upper-lower letter case normalization;
 - words extraction;
 - word stemming;
 - stop-words removal.
- For the resulting strings string distance metrics are computed:
 - Hamming distance;
 - Levenshtein edit distance, etc.
- Soundex (phonetic algorithms allowing to adjust to spelling mispronunciations).
- Thesauri (allows to adjust to such NL phenomena as synonymy and polysemy).

Utilization of lexical information is traditional for ontology mapping techniques. Even if the system applies other mapping methods, it somehow exploits the lexical information as well. For example, (Hovy 1998) describes a method for semi-automatic alignment of domain ontologies to a central ontology. The method is based on a set of heuristics for lexical comparison of concept labels and definitions across the ontologies. The lexical component of the method first breaks composite names into words, then analyses word sets for common substrings, finally it computes the ratio of common words. Finally, a limited analysis of taxonomical relations among the concepts is performed. *GLUE* algorithm (Doan, Madhavan et al. 2002) combines the results of two separate mappers, one of which is based on the lexical similarity of extended class names

(where an extended class name is a concatenation of a class name and names of all its super-classes up to the root class of an ontology).

2.3.3 Ontology mapping based on ontology structure

The structure of an ontology should represent the objective relationships among the element of domain semantics. If two ontologies of the same domain are designed in a meaningful way, the parts modeling the same piece of domain knowledge should structure it similarly. Therefore, another source of evidence that can signify the alignment among elements of two ontologies is the resembling structural patterns in the ontology graphs.

Most of the ontology mapping techniques rely on structural relationships among ontology elements to a certain degree. The simplest reasoning pattern would be similar to the following: if two concepts of ontologies are mapped, their super-classes are likely to map, same as their subclasses, same as their siblings. Similar heuristics is implemented in the *PROMPT* ontology mapping algorithm (Noy and Musen 2003). Once it finds a confirmed mapping, it starts harvesting the neighborhoods of the mapped elements to suggest new mappings.

PROMPT later modification *AnchorPROMPT* analyses ontologies as directed labeled graphs, where classes represent nodes and predicates – links. *AnchorPROMPT* considers two sub-graphs from the ontologies and compares possible paths through the subgroups restricted by anchors. If, as a result of the analysis, two classes often occur in the same parts of the graphs, they represent similar domain concepts.

The already mentioned *GLUE* algorithm (Doan, Madhavan et al. 2002) exploits ontological structures in two ways. First, it uses the chains of hierarchical relationships to compute the extended class names (see above). Second, it applies a technique known as

relaxation labeling on the final stage to refine the obtained mappings. Relaxation labeling postulates, that the label of a graph node is influenced by the node's neighborhood. *GLUE* quantifies the influence of the neighborhood on the node label as a probability function of the neighborhood features, where features represent some common sense constraints (such as “*If all children of node X match node Y, then X also matches Y*”). Then it finds the solution of the local optimization problem (the solution is characterized by the equilibrium point).

2.3.4 Ontology mapping based on instance corpora

Some ontology mapping algorithm try to utilize not only the content of ontologies themselves, but also the corpora of textual instances annotated with the ontology elements. An access to instances sets significantly increases the amount of data available for ontology mapping algorithm. To manage such volumes of information effectively ontology-mapping algorithms implement various probabilistic and machine learning methods. This allows them to benefit of well-developed methodology for statistical reasoning based on the unstructured textual data.

One example of such algorithms is implemented in the project *FCA-Merge* (Stumme and Madche 2001). It applies Formal Concept Analysis for bottom-up ontology merging. The combined set of class instances of the original ontologies undergoes lexical preprocessing and then used to create the merged concept lattice of the two ontologies. After the resulting lattice is automatically pruned, a human expert should analyze it and manually extract from it the resulting ontology.

GLUE algorithm (Doan, Madhavan et al. 2002) is another example of machine learning-based ontology mapping. Its instance-based mapping component takes as an input two ontologies along with the corresponding sets of instances. On the first step, for every pair of concepts from

the two ontologies *GLUE* estimates the joint probability of their distribution in the joint set of instances. For this purpose, *GLUE* trains a classifier for every concept using its instance set as positive (instance is annotated with the concept) and negative (instance is not annotated with the concept) cases; and then cross-classifies instances of the second ontology using just-trained classifiers. Then *GLUE* repeats the procedure with the ontology roles reversed. Thus, for every pair of concepts *GLUE* is able to break the joint set of instances into four subsets (instance has classified positively for both concepts, for one of them, or for none them). Obtained joint probability distributions are converted into the concept similarity estimations, using Jaccard similarity coefficient. As a final step, for every concept *GLUE* chooses the mapping candidate from another ontology, for which it has the maximum similarity value. The results of the instance-based mapper are combined with the result of the name-based mapper (see section 2.3.2) and then the combined mappings are refined using relaxation labeling technique (see section 0).

2.4 ONTOLOGICAL TECHNOLOGIES FOR PERSONALIZED E-LEARNING

The ideas of Semantic Web and knowledge representation with ontologies have become popular in many areas of Information Sciences, including the adaptive e-Learning. (Mizoguchi and Bourdeau 2000) list 10 problems of Intelligent Educational Technology and proposed how more formal system design principles relying on ontological engineering could help to overcome those problems. Based on the analysis of the existing research (Dicheva, Sosnovsky et al. 2005) developed an ontology of ontological technologies for e-learning and implemented a Web-portal driven by this ontology, which facilitates the description and discovery of on-line resources in

this area (such as papers, workshops, research groups etc.). (Winter, Brooks et al. 2005) analyzed the ways, Web-ontologies can help student modeling. Among the advantages of ontology-based student models they named “*formal semantics, easy reuse, easy portability, availability of effective design tools, and automatic serialization into a format compatible with popular logical inference engines*”.

Figure 14 visualizes the structure of this section. It shows the connections between the ontological technologies developed within the Semantic Web and the information technologies that are at the core of personalized e-Learning. Ontologies have been adopted on all level of information processing within adaptive educational systems. They help to unify and formalize representation of knowledge about students, domain, content, etc. They supply new techniques for mining, inference and interpretation of information about students. They help to enhance existing adaptation technologies, and lead to the creating of new ones. Finally, ontologies can facilitate the solution of Web-inspired challenges of personalized e-Learning, such as integration of AES’s, interoperability of student profiles, coherent modeling and teaching of students across multiple e-Learning platforms.

This section will give more details on the emerging field of ontology-based personalization, it will describe a set of technologies developed there and will provide examples of the system implementing this technologies in the context of e-Learning. It will not cover ontology-based content modeling and open-corpus personalization; the detailed description of this field has been given in the Section 2.2.

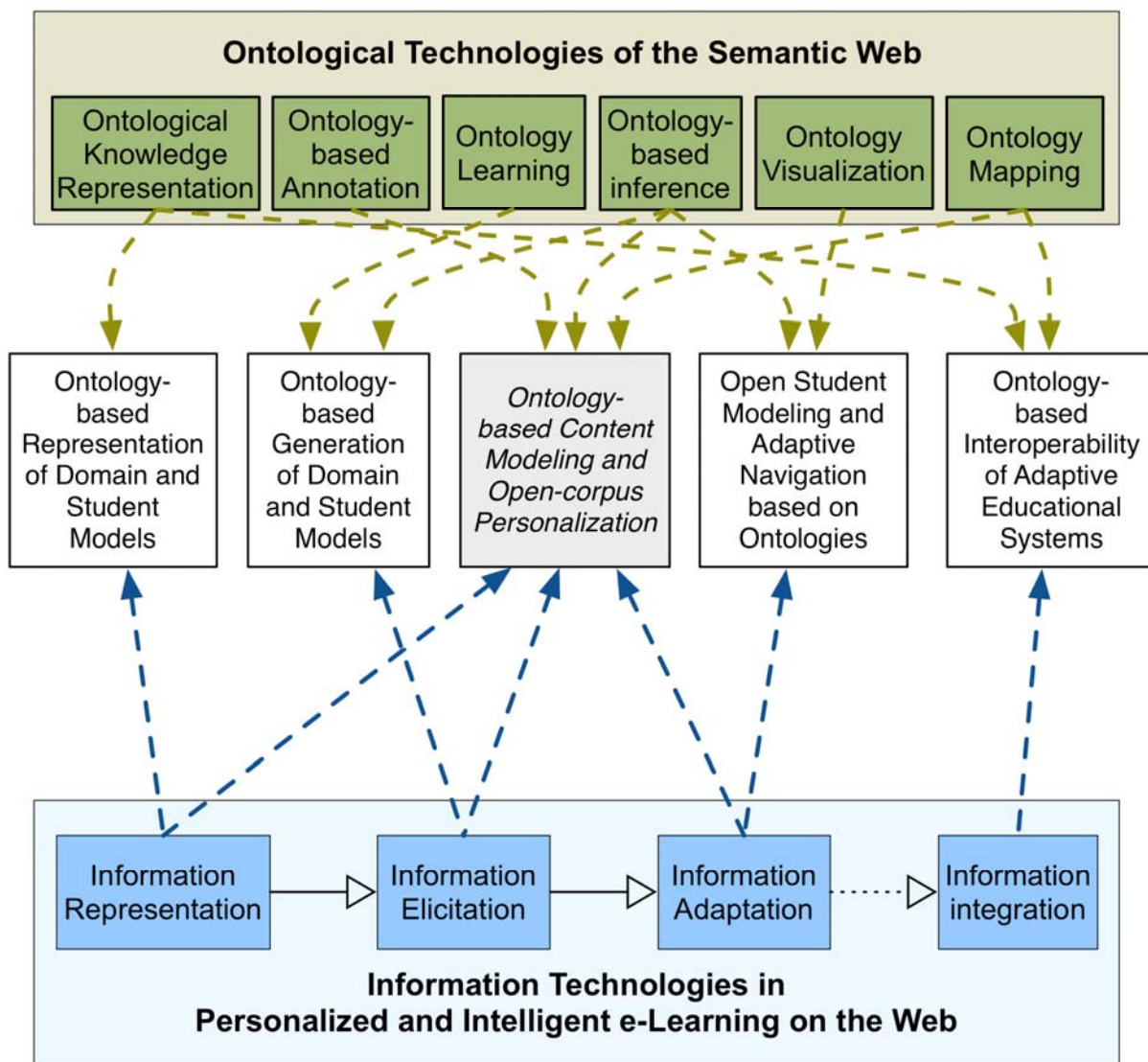


Figure 14: Ontological technologies for personalized e-Learning

2.4.1 Ontology-based representation of domain and student models

Ontologies are in the first place a knowledge representation technology. Thus, almost any user-adaptive system employing an ontology for student modeling and adaptation, on some level uses it for representation purposes. Even when the major goal for utilizing ontologies is to benefit of one of the derivative ontological technologies (e.g. using ontology mapping for user model

mediation or applying ontology-based learning for automatic construction of user model), the primary design decision made by the adaptive system developer would be to represent some part of system's knowledge as an ontology.

The diversity of ontological approaches for representation of student models springs both from the vast pool of research accumulated in the past for user model representation and from the opportunities, ontologies provide for the new generation of adaptive systems. However, two principle directions can be identified that summarize the whole spectrum of approaches. An adaptive system can employ ontologies for modeling the structure of the domain and represent atomic student characteristics based on the elements of the ontology. Another trend is to structure a complex student profile modeling several dimensions of her state as an ontology. The first direction inherits its main ideas from the overlay user modeling (Carr and Goldstein 1977), while the second approach has roots in stereotype user modeling (Rich 1979).

2.4.1.1 Ontology-based overlay user modeling and personal ontology views

The most conventional implementation of ontology-based user modeling is the overlay user modeling that relies on a domain model represented as an ontology. A simplified view on an ontology is a network of concepts. It provides a natural basis for modeling a domain of discourse following multiple examples in the area of AI and Intelligent Systems. Over the years several network-based knowledge representation formalisms have been explored, including Semantic Nets (Woods 1975), Concept Maps (Novak and Gowin 1984) and Bayesian Belief Networks (Pearl 1988). While similar with the mentioned modeling frameworks in the idea of network-based domain representation, ontologies differ in providing a basis for implementation of sharable and reusable models with rich built-in inference capabilities. Representation of a domain model as an ontology and modeling user's knowledge, interests, needs, or goals as a

weighted overlay on the top of it enables usage of standard representation formats and publicly available inference engines, as well as an access to a vast pool of technologies for ontology mapping, queering, learning, etc. Such straightforward overlay user modeling based on ontologies is implemented, for example, in the *ActiveMath* system (Melis, Goguadze et al. 2006).

The domain ontologies following principles of formal ontological theory also benefit of such properties of ontologies as intentionality and explicitness, which allows building unbiased and logically complete domain models (Guarino 1998).

Many projects using ontologies for modeling characteristics of an individual user go beyond the classic overlay approach. The student model can consist of a subset of concepts from the domain ontology. This is typical for recommender systems representing user interests. The presence/absence of a concept in the user model signifies the presence/absence of user's interest in the corresponding concept. The concepts can also have associated values (weights, verbal labels), characterizing the magnitude of the interest.

Another possible extension of the simple overlay user modeling based on a domain ontology is to model user characteristics relevant not only to the concepts of the ontology but also to its relations and axioms. It could be the knowledge of the fact that a "*human being is-a mammal*" (relation), or that every person "*has two and only two*" parents (axiom); it can be an interest in anything which is a "*part-of*" a particular Laptop model, etc. A somewhat related idea is also implemented in (Sicilia, García et al. 2002; Sicilia, Garcia et al. 2004). The authors introduce a concept of learning link for modeling relationships between learning objects and propose an ontology of such links. Concepts of this ontology are used to annotate the hyperlinks in a hypermedia tutorial. The designed prototype system adaptively navigates users taking into

account not only the concept knowledge but also the history a user has had for the particular type of links.

Finally, a student model can contain not a weighted mask over the domain model, but an individual network of concepts reflecting personal conceptual structure in the domain of discourse. For models of this kind (Huhns and Stephens 1999) introduces a specific term “*personal ontologies*” to designate a computation model reflecting an individual perspective on the world. The term has been criticized later by (Kalfoglou, Domingue et al. 2001) for its contradiction to the definition of ontology as a model representing a *shared* view on the domain. Accepting the use of the concept itself, however, they suggested another name – “personal ontology view”. Such term conveys the meaning of the concept more precisely, as the ontology-based user model is not an ontology itself, but its individualized permutation. Kalfoglou et al. introduced a new term for these models – “*personal ontology views*” (POV). From a cognitive perspective POVs are consistent with constructivist paradigm and cognitive models like Concept Maps (Novak and Gowin 1984), as they represent domain conceptualization from the point of an individual.

Such an approach is implemented in the *OWL-OLM* (Denaux, Aroyo et al. 2005; Denaux, Aroyo et al. 2005) – an adaptive system eliciting student knowledge by means of interactive dialog about the subject domain. Based on the student’s utterances a dialog agent builds a POV representing an individual student’s conceptualization. It may contain relationships not existing in the domain ontology, but reflecting student’s understanding of this part of the domain. By comparing the domain ontology and the elicited student conceptualization *OWL-OLM* identifies faulty relations, localizes the problem, and infers the pattern responsible for the error in the conceptualization.

2.4.1.2 Ontology-based user profiles

Sharability and reusability of user models require not only standardized domain representation but also common vocabularies for describing the user model itself. Therefore, another direction for application of ontologies in user modeling has become the design of ontologies describing sharable user profiles. The term “*user profile*” has been used traditionally to refer to the compound knowledge structure reflecting various static information about a user (such, as user’s demography, background, cognitive style, etc.). It is different from “*user model*” that usually represents a dynamic snapshot of a single aspect of a user (conceptual knowledge, interests, preferences, etc.). However, the distinction is rather conditional. These terms have been often used in the literature as substitutes. User models can have a compound structure and store static user information, as user profiles can include a dynamic component representing current state of a certain user characteristics.

A standardized user profile will make possible the implementation of interoperable adaptive systems sharing modeling information. Ontology-based user profiling is especially important for systems reasoning across multiple profiles (social adaptive systems) or for systems that can benefit from complex inference on multiple ontologies representing different knowledge (e.g., adaptive pervasive systems using ontologies to model domain and context knowledge).

The first steps towards user profile ontologies have been done in works of Murray and Mizoguchi. (Murray 1998) developed the ontology of a *Topic*, employed to assess all aspects of student’s understanding of a single knowledge element. Mizoguchi et al. developed the task ontology for AES (Mizoguchi, Sinitsa et al. 1996). This ontology defines concepts involved in the design and application of AES and has been intended to facilitate the development of reusable IES components. One of the top-level super-concepts is “*Learner's state*” that combines

various children concepts describing a student/learner in AES. (Chen and Mizoguchi 1999) proposed multi-agent architecture of AES and described a scenario of communication between a learner model agent and another agent based on this ontology.

The transfer of a large part of computer user activity to WWW along with the development of XML technologies for metadata definition created both: the need and the possibility for development of metadata standards for describing users on the Internet. There are two such metadata initiatives developed solely for the field of education: IEEE Public and Private Information (*PAPI*) for Learners (IEEE-LTSC 2001) and IMS Learner Information Package Specification (IMS 2001). Both standards provide XML bindings for their tags.

W3C recommended a more elaborate framework for metadata description – RDF. As a result, several RDF-based metadata standards for describing users have been proposed. The most widely-accepted metadata standard on the web, DCMI has several elements for defining different categories(roles) of users with respect to a document (DCMI 1999). W3C has issued RDF-binding for *vCard* (W3C 2001), a standard for exchange of information about a user in the form of electronic business cards. Brickley and Miller proposed *FOAF* – an RDF-based general-purpose model for description of user on the Web (Brickley and Miller 2005). The main motivation of the *FOAF* project is to provide a metadata schema for implementation of social networks on the Semantic Web.

Several attempts have been made to integrate previously developed models into a single synergetic representation of a user. Dolog and Nejdl designed an architecture for ontology-aware AES on the Semantic Web. One of the ontologies in this architecture is the learner profile ontology, developed as a combination of *IEEE PAPI* and *IMS LIP* (Dolog and Nejdl 2003). This ontology was extended with new concepts for the implementation of *Tangram* learning system

(see Section 2.2.6). (Ounnas, Liccardi et al. 2006) analyzed the applicability of several existing approaches (including the already mentioned *IEEE PAPI*, *IMS LIP*, Dolog and Nejd1's student profile ontology and *FOAF*) to the development of Learner Profile in Social Software. They decided to use the superset of these models taking *FOAF* as a basis.

2.4.2 Ontology-based generation of domain and student models

A number of research projects attempt to achieve a challenging goal of mining the structure of the domain/user model itself based on user activity with information resources. To facilitate the complete or partial model creation these systems apply a set of techniques from the field of ontology learning (Shamsfard and Barforoush 2003; Gomez-Perez and Manzano-Macho 2005). These techniques differ in the input data (structured – databases or other ontologies; semi-structured – dictionaries and encyclopedias; and unstructured – text), algorithms (degree of automation and interactivity, linguistic vs. statistic, etc.) and the resulting output (ontology vs. thesauri, only concepts vs. concepts and relations, etc.).

MECUREO is an example of a system learning domain ontology from semi-structured learning material (Apted and Kay 2004). It was developed for automatic construction of Computer Science ontology from Free On-Line Dictionary of Computing (*FOLDOC*). Unified format of *FOLDOC* pages and consistent rules of linking between them allow *MECUREO* to learn not only concepts but also 4 types of relations: parent, child, synonym, and antonym. The resulting ontology learnt by *MECUREO* has been then applied for visualization of open student models, for navigating students to the “right” learning material (Apted, Kay et al. 2003), and for automatic generation of learning object metadata (Apted, Kay et al. 2004).

MECUREO relies on ontology learning for extracting only domain model. Several systems try to go a little further and automatically learn the conceptual structure of the student model itself. *Willow* (Pérez-Marín, Alfonseca et al. 2006; Pérez-Marín, Pascual-Nieto et al. 2006; Pérez-Marín, Alfonseca et al. 2007) is a good example of such a system. It is an automatic assignment grading system assessing free-text student's answers. When a teacher creates assignments in *Willow*, s/he needs to specify to which of the top-level domain categories the assignment belongs. Every assignment also requires at least three sample answers provided by different experts to build a reference answer model and take care of possible rephrasing. When a student submit her/his own answer to a problem, *Willow* extracts the concepts covered in the answer and matches them against the concepts learnt from the reference model. As a result, it creates an individual conceptual model of a student reflecting the correctly reported knowledge and possible misconceptions. Based on the extracted student model, *Willow* generates a grade and a feedback containing the most important parts of the student's answer responsible for concept assessment and the reference answer used as an etalon. The evaluation of *Willow* shows that there is statistically significant correlation between grades given by the systems and actual grades received by the student in class.

2.4.3 Ontology-based open student modeling and adaptation

In addition to powerful representational capabilities, ontologies can provide a useful way to structure domain knowledge and present it to a user in comprehensible and navigable form. Several adaptive systems have exploited this feature to visualize the domain semantics for users and directly communicate to them the current state of their user models. Two popular ontology visualization layouts used by these systems are class hierarchy and concept maps; however, some

other techniques have been tried as well. Some systems used ontologies solely as a navigation aid, others implemented ontology-based interfaces to open user models to the users, finally a limited set of systems exploited ontology visualization to elicit user's characteristics in an interactive manner.

The *OWL-OLM* system previously discussed in Section 2.4.1.1 provides an example of using ontologies to elicit student's knowledge and visually present to the student her current conceptual state (Denaux, Aroyo et al. 2005; Denaux, Aroyo et al. 2005). The interface of *OWL-OLM* is presented on Figure 15. *OWL-OLM* organizes the process of student knowledge elicitation in the form of interactive dialog between the student and the dialog agent. The agent analyzes the current state of student knowledge model and infers a concept, not known by a student, or for which the student demonstrated a misconception. The agent maintains the dialog to assess student's knowledge of the entire neighborhood of the target concept, including the concept attribute and its relations with other concepts in the domain ontology. Figure 15 shows the part of the dialog about the file system in *Unix*. The dialog agent states its utterances in text form. The student creates respond utterances using the graphical editor by building gradually the target part of the ontology. The system converts student's actions into OWL statements and verifies it against the domain ontology. If it finds a mismatch between the student model and the domain ontology the dialog is altered to resolve this mismatch. The dialog ends, when the dialog agent fills that it has properly assessed all knowledge relevant to a particular concept, or it can be interrupted by a student.

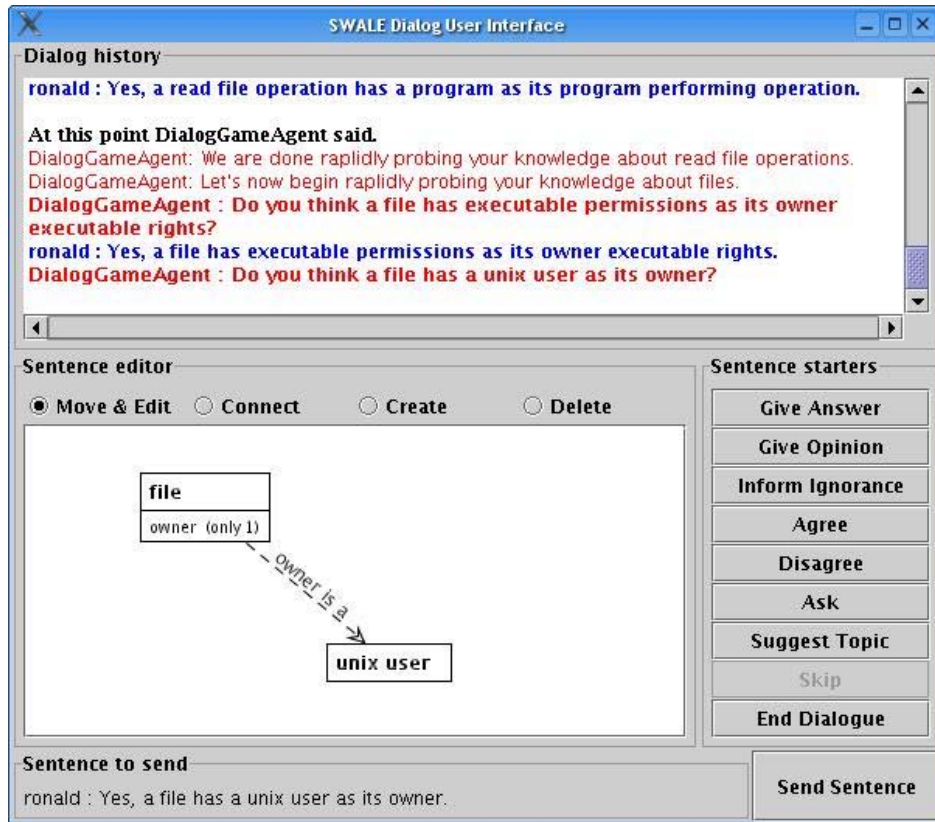


Figure 15: Ontology-based visualization of student models in *OWL-OLM*⁸

VCM (Verified Concept Mapping) system (Cimolino, Kay et al. 2004) follows a similar approach. Students working with *VCM* demonstrate their knowledge in the domain by building its concept map. The student interface of *VCM* (see left part of Figure 16) consists of the working space, where students have to draw their maps, the list of predefined concepts and the list of predefined relations. A teacher authoring the problem should populate these lists. If a student fills that some of the concepts and/or relations are missing, s/he can add their own concept map elements. After the student submits the map *VCM* checks its correctness against the etalon map created by a teacher according to the assessment rules that can be also specified by a teacher. If

⁸ From Denaux, R., L. Aroyo, et al. (2005). An approach for ontology-based elicitation of user models to enable personalization on the semantic web. 14th international conference on World Wide Web (WWW '05), Chiba, Japan, ACM, New York, NY, USA.

student's map has violated any of the rules, the system will generate the appropriate feedback. The correct passing of certain requirements is also included in the feedback. An example of the system feedback is shown on the right part of Figure 16. Both the student's concept map (student model) and the teacher-specified concept map (expert model) are represented and compared as ontologies.

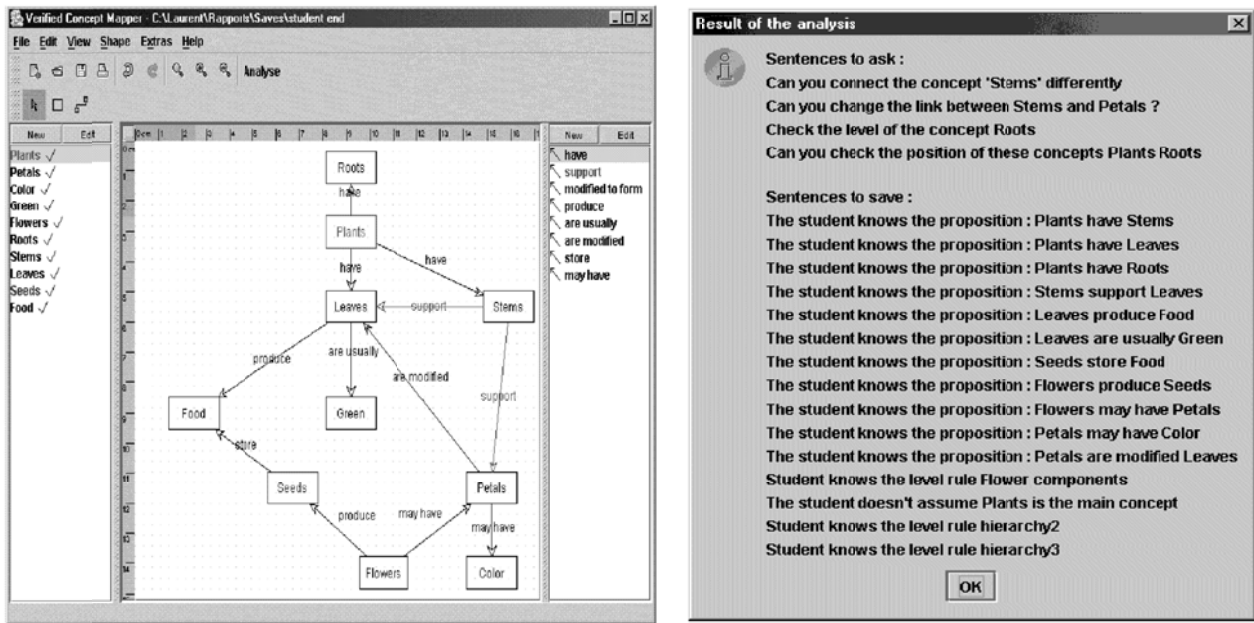


Figure 16: Ontology-based verification of students' concept maps in *VCM*⁹

COMOV (COnceptual MOdel Viewer) (Pérez-Marín, Alfonso et al. 2007) visualizes models of student knowledge extracted by the *Willow* system (see Section 2.4.2). Unlike many other systems, *COMOV* “opens” student models not for the students to reflect on them, but for a teacher to monitor the individual and group performance. *COMOV* can present the model of a

⁹ *Left*: student interface for concept maps authoring; *Right*: system feedback (from Cimolino, L., J. Kay, et al. (2004). "Concept mapping for eliciting verified personal ontologies." *International Journal of Continuing Engineering Education and Lifelong Learning* 14(3): 212 - 228.)

particular student and the aggregated model of the class. It experiments with four different kinds of visualizations: concept map, table, bar chart (or skillometer) and text summaries (see Figure 17). The node colors represent conceptual knowledge of a student or a class on the scale from red (no knowledge) to green (maximum knowledge). Knowledge levels of leaf-concepts are aggregated to propagate knowledge for top categories in the ontology. The evaluation of *COMOV* with real teachers demonstrated mainly positive attitude towards the system. Teachers appreciated the opportunity to observe student's performance on the conceptual level.

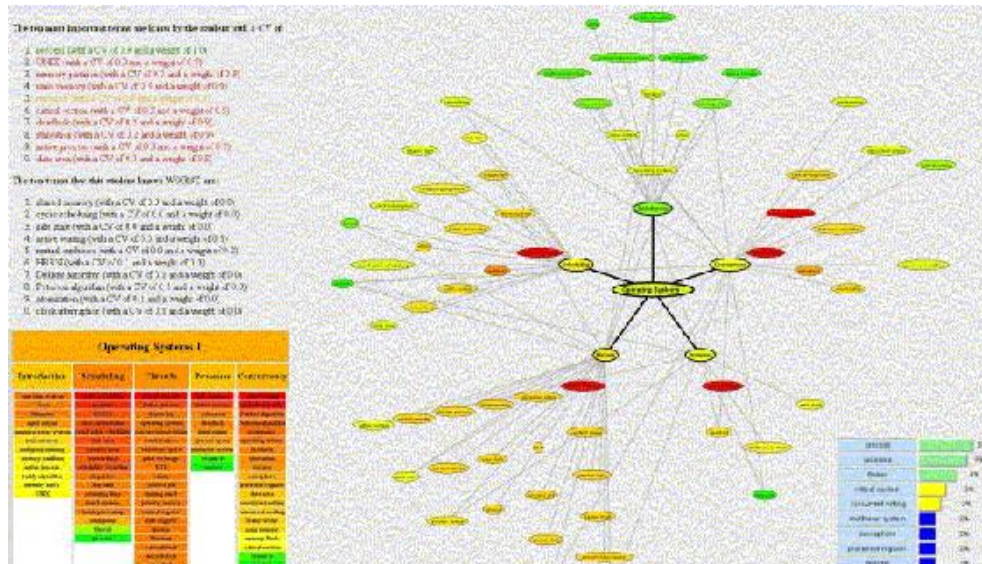


Figure 17: Student model visualizations in *COMOV*¹⁰

VIUM project (Apted, Kay et al. 2003; Apted, Kay et al. 2003; Uther and Kay 2003) proposes an interesting solution to the problem of visualizing large ontology-based user models.

¹⁰ From Pérez-Marín, D., E. Alfonseca, et al. (2007). *Automatic Generation of Students' Conceptual Models from Answers in Plain Text*. 11 International Conference on User Modeling (UM'2007), Corfu, Greece, Springer Berlin / Heidelberg.

When the ontology-based student model consists of several hundred concepts, presenting it to a student becomes a challenge. *VIUM* visualization (see Figure 18) allows users both to have a snapshot of the entire model and to concentrate on the concept that is currently in the focus. The visualization applet occupies a fixed area on the screen and can present as many concepts as needed. When a user clicks on the concept, *VIUM* highlights the concept by increasing the label font and reserving some space around it. The relevant concepts are highlighted in the same way, but with less magnitude. The rest of the concepts not relevant to the current user's focus are presented in the small font and very close to each other. However, when a user browses the hierarchy all concepts are clickable, once a user clicks on the concepts, the focus shifts to it. The relevant positions of concepts represent distance between them in the model; and the left indentation shows the level in the hierarchy. Essentially, the *VIUM* visualization is a smart modification of the traditional class hierarchy, which allows keeping all concepts on the screen and accessing them in one click. The color of concept labels represents user's interest in the concepts and changes on the scale from red (minimum) to green (maximum). The creators of *VIUM* used its interface as a part of adaptive navigation system for learning material.

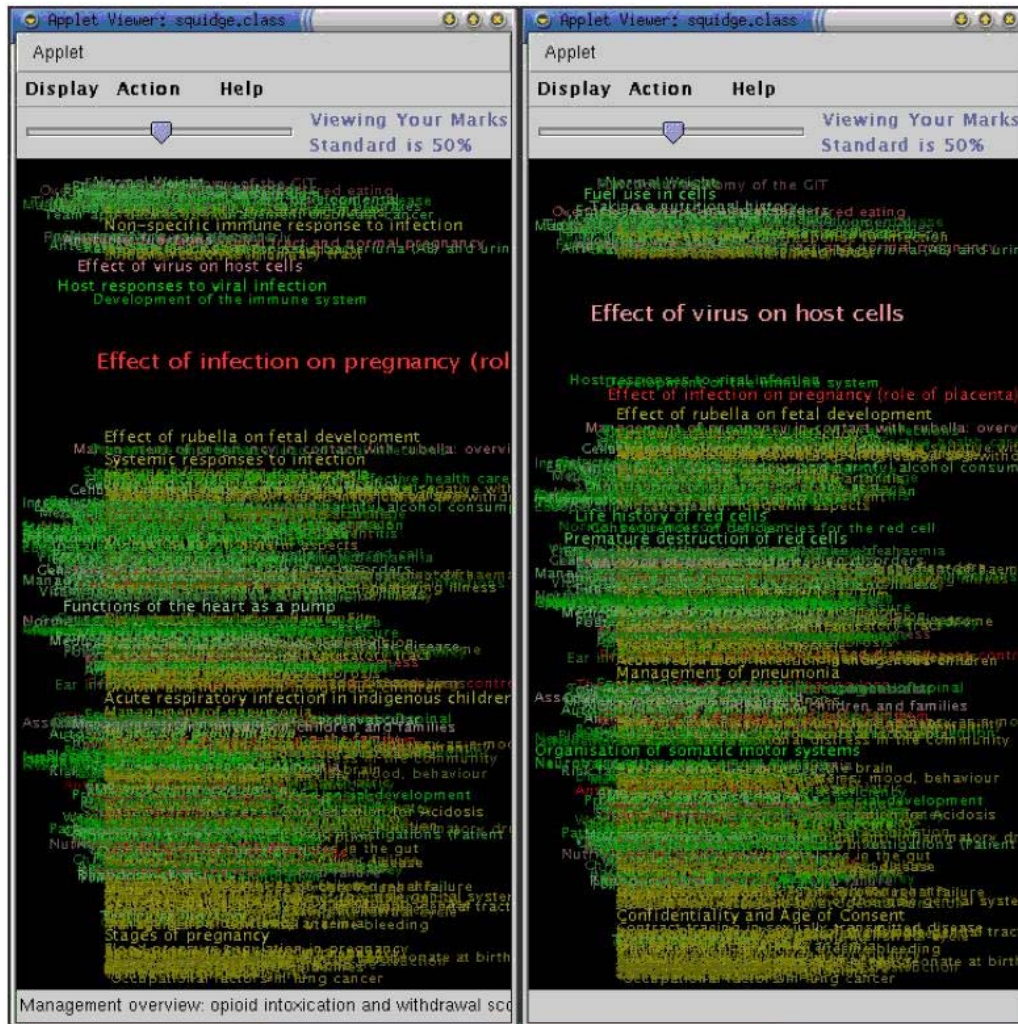


Figure 18: Visualizing conceptual user models in VIUM¹¹

2.4.4 Ontology-based interoperability of adaptive educational systems

Over the last 10 years, a number of adaptive educational systems have migrated from research labs to real life, and are now employed by thousands of users. In some application areas, the “density” of practical adaptive systems is reaching the point where several adaptive systems are

¹¹ From Apted, T., J. Kay, et al. (2003). Visualisation of ontological inferences for user control of personal Web agents. Seventh International Conference on Information Visualization.

available. Yet, in most cases, these systems do not compete, but rather complement each other, while offering unique functionality or content. This puts the problem of using several adaptive systems in parallel on the agenda of the AES community. This problem has been explored over the last few years by several research teams and from several perspectives: architectures for integrating adaptive systems (Brusilovsky 2004), cross-system personalization (Niederée, Stewart et al. 2004; Carmagnola and Dimitrova 2008), and user modeling servers (Kay, Kummerfeld et al. 2002; Kobsa and Fink 2006; Yudelson, Brusilovsky et al. 2007). This section overviews how ontologies can help to advance the development of interoperable adaptive educational systems by facilitating domain and student model integration.

The task of interfacing two student models involves resolution of modeling differences or discrepancies. There are four main sources of such discrepancies:

- different student modeling approaches;
- mismatches in domain models;
- different representations of user profiles;
- scale discrepancies.

2.4.4.1 Integrating modeling approaches

Adaptive educational systems can represent information about students in many different ways: concept-based, term-based, stereotypes, constraint-based, Bayesian networks, collaborative filtering, etc. The problem of translating between two fundamentally different representations of a student created by systems that understand only one of them is a great challenge. Ontologies can provide a solution in a limited number of cases. For example, the methods of ontology-based knowledge acquisition can be employed for transforming term-based models into concept-based;

the lexical ontologies as *WordNet* also facilitate extraction of semantic information from the term vectors. (Suraweera, Mitrovic et al. 2004) introduce project *ASPIRE* that utilizes ontologies in constraint-based tutors.

2.4.4.2 Integrating domain representations

Overlay student modeling based on conceptual domain representation is the most popular approach for modeling students' knowledge in AES. However, even when both integrated systems employ overlay models to represent students' knowledge in the domain, the underlying domain models of these systems are likely to be very different in many aspects (conceptual structure, granularity, representation technologies, etc.). Ontologies can facilitate integration of overlay models in several ways. First of all, semantic web ontology standards provide a uniform model for representing knowledge; hence, the system designers have an easier choice of basic technologies for modeling their domains. Secondly, ontologies are intended as shared models. Developers of different systems might be willing to agree on using a single ontology. In some domains such well-accepted ontologies already exist, e.g. *Gene Ontology* in the field of molecular biology (Ashburner, Ball et al. 2000). Finally, when the domain ontologies differ, ontology mapping techniques can be employed to resolve the differences. Once the mapping has been found the systems should be able to align their domain models and mediate users' information.

Several AES integration efforts have benefited of using ontologies for modeling domain and student models. The *OntoAIMS* project is an example of two separate adaptive systems integration based on a shared domain ontology: *AIMS* and *OWL-OLM* (Denaux, Dimitrova et al. 2005). *OWL-OLM* supports interactive elicitation of learner's knowledge represented as an OWL ontology and available for the *AIMS* system, which provides the learner with adaptive access to

available educational content. The model supplied by *OWL-OLM* is utilized by *AIMS* for adaptive recommendation of a next learning task. While the learner is browsing the task material and definitions of relevant concepts with *AIMS*, the systems updates her/his model as well. Next time a learner works with *OWL-OLM*, it will not try to elicit concepts learned with *AIMS*.

Two examples of AES integration based on domain ontology mapping are *Medea* (Trella, Carmona et al. 2005) and *M-OBLIGE* (Mitrovic and Devedzic 2004). *Medea* is a learning portal that provides students with access to multiple educational systems from a single point. To maintain consistent adaptivity across several applications, *Medea* needs a mechanism for mediating user's model between the applications. This task is achieved by manual mapping of domain models of integrated adaptive systems to the central domain model of *Medea*. Hence, student's knowledge assessed by any AES accessed from *Medea* are translated into the central domain representation and stored in the central student model of *Medea*. If a system needs to obtain the current state of the student's model it asks *Medea*.

The authors of *M-OBLIGE* consider a problem of mediating knowledge models of the same learner between several intelligent tutors teaching a learner different aspects of database management (entity-relationship diagrams, SQL, database normalization). Although, the domains of these tutors are different they share several concepts. For reasoning across tutor's local ontologies and mediating user's knowledge models *M-OBLIGE* introduces an ontology processor. The ontology processor uses the global ontology (e.g. the general ontology of data models) to link tutors' local ontologies and support cross-tutor personalization.

2.4.4.3 Integrating student profiles

Adaptive systems can collect diverse information about a student and maintain complex student profiles. Even though, most of the student's characteristics modeled by AES are typical,

the actual profile representation can vary from system to system. For example, the structures of the top categories from two standard learner profiles: *IEEE PAPI* (IEEE-LTSC 2001) and *IMS LIP* (IMS 2001) are very different. Identical categories have been named differently (e.g. *PAPI*'s "Personal" and *IMS LIP*'s "identification"). Some *PAPI* categories correspond to several *IMS LIP* categories (e.g. *PAPI*'s "Relations" and *IMS LIP*'s "affiliation" and "relationship") and some *IMS LIP* categories cover data that belongs to several *PAPI* categories (e.g. *IMS LIP*'s "activity" and *PAPI*'s "Performance" and "Portfolio"). Figure 19 represents the complete mapping of *IEEE PAPI* and *IMS LIP* as given in (Klamma, Chatti et al. 2005).

There were several attempts to develop a unified ontology for student profiles, and/or to map the existing ones (Dolog and Nejdil 2003; Jovanovic, Gazevic et al. 2006; Ounnas, Liccardi et al. 2006). Section 2.4.1.2 provides a brief overview of these efforts.

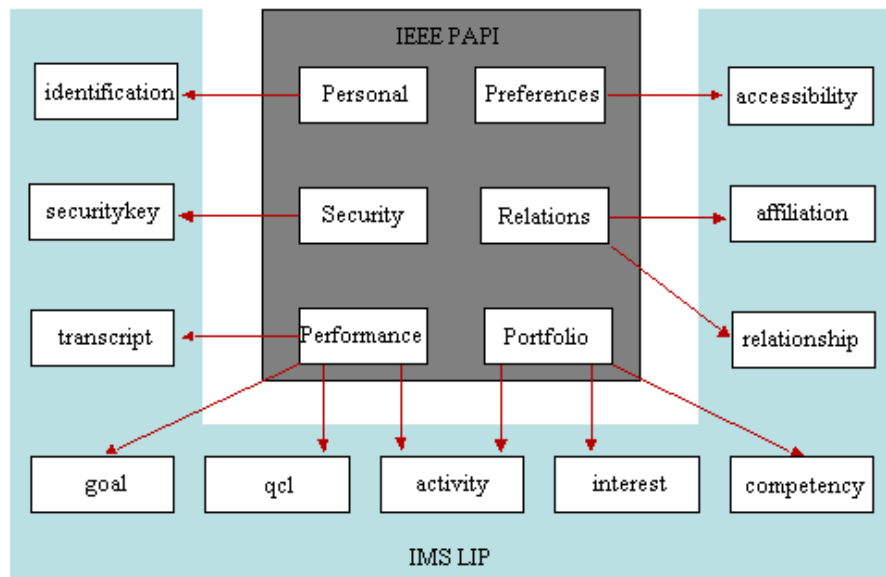


Figure 19: Relationships between top categories of *IEEE PAPI* and *IMS LIP*¹²

2.4.4.4 Integrating modeling scales

The final source of divergence between user models is different scales used for assessment of user's attributes. For example, two educational adaptive systems use overlay student modeling approach, operate in the same domain, and rely on the same domain ontology, yet they can model student's knowledge using different scales for assessment of her/his knowledge. Scales can be binary (knows / does not know), categorical (High / Medium / Low), numeric (from 1 to 5), probabilistic (certainty that the user knows the concept). Several problems can arise during the mapping of two scales:

- how to map two different categorical scales (when the category labels are different, when the numbers of categories are different);
- how to map probabilistic scale into categorical (categories thresholds);

¹² From Klamma, R., M. A. Chatti, et al. (2005). "LIP2PAPI Converter." from <http://dbis.rwth-aachen.de/lehrstuhl/staff/chatti/LIP2PAPICConverter/index.html>.

- how to align the internal inference mechanisms of two different systems (one system can use an asymptotic formula for knowledge level estimation, when another can use a Bayesian approach).

Scale ontologies could be developed to help solving the first two problems. If numeric values are expressed using XML standard data types, they become available for standard RDF reasoning engines, which can be used to provide automatic mapping to ontology-based scales. If systems' internal inference mechanisms are explicated and serialized in RDF, a set of rules could be developed to map the values inferred using different algorithms.

Although resolution of any type of discrepancies in practice will lead to the loss of modeling data, mediated user models stay a valuable source information about a user. Most of the time, the alternative to the non-precise user models translated from another system is an empty user model, which deviates from the actual user's state much more drastically.

2.5 CONCLUSION

The main topic of this dissertation is open-corpus personalization, but it is also related to ontology mapping and ontology-based user modeling and adaptation. As it is shown in the next chapters, the dissertation contributes to all three of these fields. For open-corpus personalization, it introduces a new principle approach for a fully automated creation of semantic content models and implementation of the complete personalization cycle: from tracing students' activities with open-corpus content to adaptive presentation of this content according to the current state of students' knowledge. For ontology mapping, a new algorithm has been designed that carries out a task of semantic linking between a traditional high-quality fine-grained domain ontology and

an automatically extracted topic-based coarse-grained model of an online collection. This algorithm implements several novel techniques and possesses a number of unique features. Finally, with regards to the area of ontology-based personalization, this dissertation provides another example of the good synergy between Semantic Web and Adaptive Web technologies, by solving a classic personalization problem with the help of ontologies.

3.0 EXTENDING THE SCOPE OF SEMANTIC MODELING OF OPEN-CORPUS CONTENT FOR E-LEARNING

3.1 INTRODUCTION

This chapter provides an in-depth analysis of the ways to support semantic modeling of open-corpus content and illustrates them with the concrete examples. All examples are taken from the projects carried out by the author during his PhD study. This chapter demonstrates the author's contribution to the field of open-corpus personalization and semantic modeling of open-corpus content. It also shows the evolution of the author's research on in this field that, at the end, helped in setting the goal and formulating the main approach proposed in this dissertation.

Generally speaking, the task of Web-based open-corpus personalization is not reduced to modeling open-corpus content. The relevant resources need to be discovered, preprocessed, and modeled according to the representation model used by the system. The new documents should be incorporated into the existing document space and users should be given an effective way to access them. However, from the research perspectives, as well as from the practical one, content modeling is the central part of content-based open-corpus personalization. There are only three principle approaches to address this issue:

1. Automatic indexing of structured content.
2. Relaxing requirements to domain modeling.

3. Utilization of external models.

The rest of this chapter is structured as follows. Section 0 describes in details all three principle approaches towards supporting semantic modeling of open-corpus content. Section 3.3 presents implementation of automatic indexing of structured content (Sosnovsky, Brusilovsky et al. 2004) and its refinement (Brusilovsky, Sosnovsky et al. 2005). Section 3.4 discusses coarse-grained topic-based domain and content modeling implemented in the system QuizGuide (Brusilovsky, Sosnovsky et al. 2004). It discusses the benefits this approach has from the point of content modeling (Sosnovsky, Yudelso et al. 2007), demonstrates the effectiveness of this approach from the point of adaptation (Brusilovsky and Sosnovsky 2005; Sosnovsky, Brusilovsky et al. 2008; Brusilovsky, Sosnovsky et al. 2009), and points out its shortcomings, when it comes to user modeling precision (Sosnovsky and Brusilovsky 2005). Finally, Section 0 presents two studies on integration of AESs based on manual alignment of domain models (Brusilovsky, Sosnovsky et al. 2008; Sosnovsky, Mitrovic et al. 2008; Sosnovsky, Mitrovic et al. 2008; Sosnovsky, Brusilovsky et al. 2009) and one experiment on model mediation based on automatic ontology mapping (Sosnovsky, Dolog et al. 2007). Section 3.6 concludes the chapter with a comparative analysis of presented approaches and their relationships to the main dissertation approach.

3.2 THREE APPROACHES TOWARDS SEMANTIC MODELING OF OPEN-CORPUS CONTENT

When content follows certain formal structural rules than can be automatically and unambiguously recognized, the content modeling component can take advantage of these

regularities and automatically associate identified constructs with the pieces of domain semantics. Figure 20 represents such situation: based on some regular structural patterns within the open-corpus content the system automatic indexes new documents with the concepts of the domain model. Such patterns can be specially formatted mini-data like currency and dates, or specially formatted text fragments like programming code.

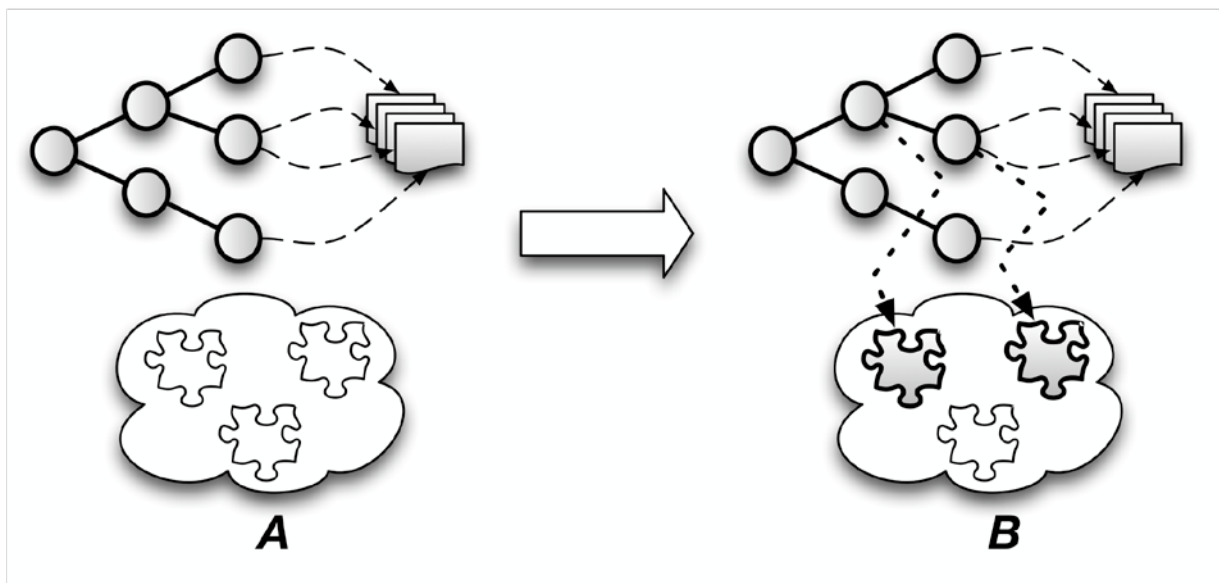


Figure 20: Modeling open-corpus content: automatic indexing of structured content¹³

Figure 21 visualizes solution number two. Conventional approach to content modeling assumes fine-grained domain models containing from tens to thousands of concepts. Description of learning resources with elements of such models is a very laborious task that also requires a basic level of expertise in knowledge engineering. Consolidation of fine-grained concepts into bigger topics helps to reduce the complexity of the modeling task. In fact, instead of many-to-

¹³ **A:** the domain model is connected only with internal set of documents, open-corpus documents are not modeled; **B:** new links are automatically generated based on the structure/formatting of open-corpus documents and a set of annotation rules

many indexing the author needs to perform many-to-one aggregation, which is a typical activity for a classroom teacher compiling a course.

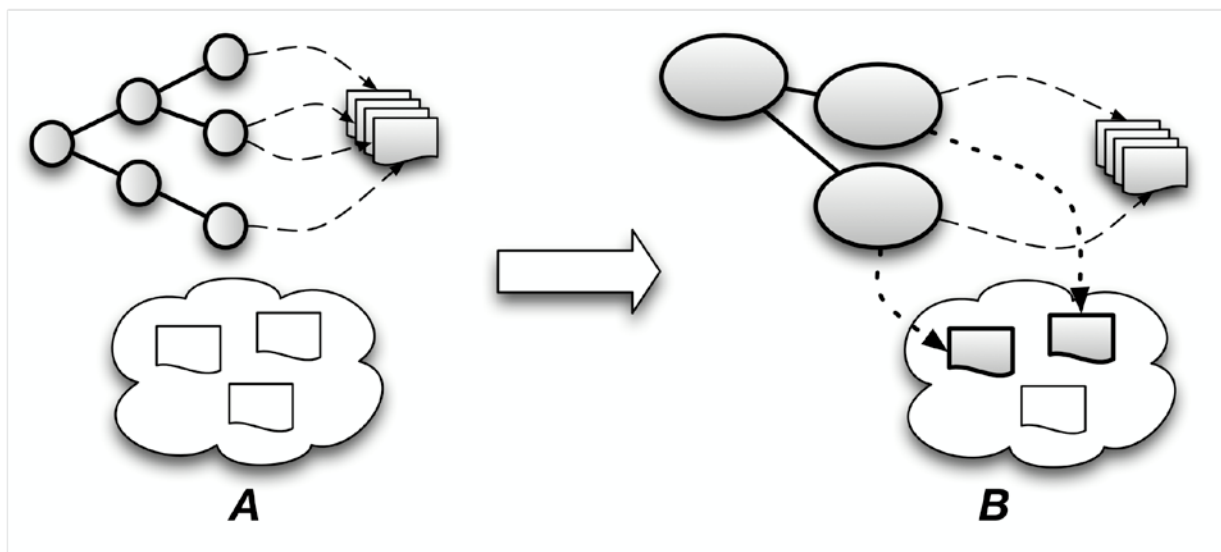


Figure 21: Modeling open-corpus content: coarse-grained domain modeling makes manual indexing easier¹⁴

The third solution is depicted on Figure 22. When external learning material comes supplied with its own content model, this model can be utilized for integrating the external material into the original system. Integration of two content-based AESs is one variation of such scenario. In this case, each system relies on some representation of the domain that allows for knowledge-based content modeling. If one can find a mapping between the two representations (either manually or automatically), it can be used as a bridge between the corresponding pieces of content models and would enable cross-modeling and cross-personalization. The external resources do not have to be provided by another system; they can be a part of learning object repository (or a digital library) annotated with semantic metadata.

¹⁴ **A:** fine-grained modeling of new documents is time and expertise consuming; **B:** simplification of domain models allows for easier content modeling

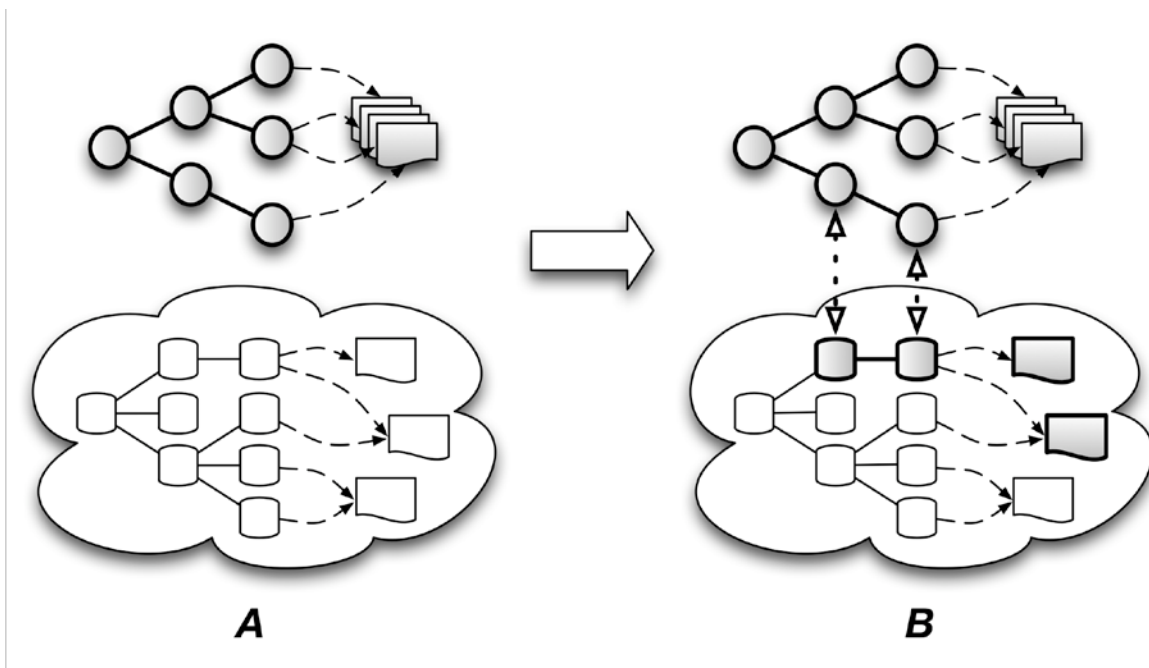


Figure 22: Modeling open-corpus content: model mediation utilized for content cross-modeling¹⁵

3.3 AUTOMATIC INDEXING OF STRUCTURED CONTENT

There are a few domains where the knowledge behind a learning object can be deduced automatically from its content based on the rules and the structure that the content follows in these domain. For example, it is possible to automatically recognize mathematical concepts behind a formula or parse programming concepts from a fragment of programming code. This section consists of two parts. First part presents a method for automatic extraction of concepts from learning material containing C programming code; it also supports identification of pedagogical roles these concepts play in learning objects (prerequisites/outcomes). Second part describes an improvement of this approach implemented in an authoring tool for programming

¹⁵ **A:** external content is indexed by another model; **B:** mapping between the original and the external models helps to link external content to the original model

exercises. The tool supported automatic indexing of created exercises and allowed authors to refine it by browsing the underlying domain ontology.

3.3.1 Automatic indexing of programming code-based learning objects

Two Web-based systems serving two different kinds of educational content have been used in this study.

The *WebEx* system provides access to interactive, explained examples of programming solutions. An author of an example or a later teacher can provide textual explanations for every line of the program code. The students can browse these comments at their own pace and order by selectively clicking on the commented lines (Figure 23). The first version of *WebEx* has been implemented in 2001 (Brusilovsky 2001). Since that time, *WebEx* has been heavily used in the context of an introductory C programming course.

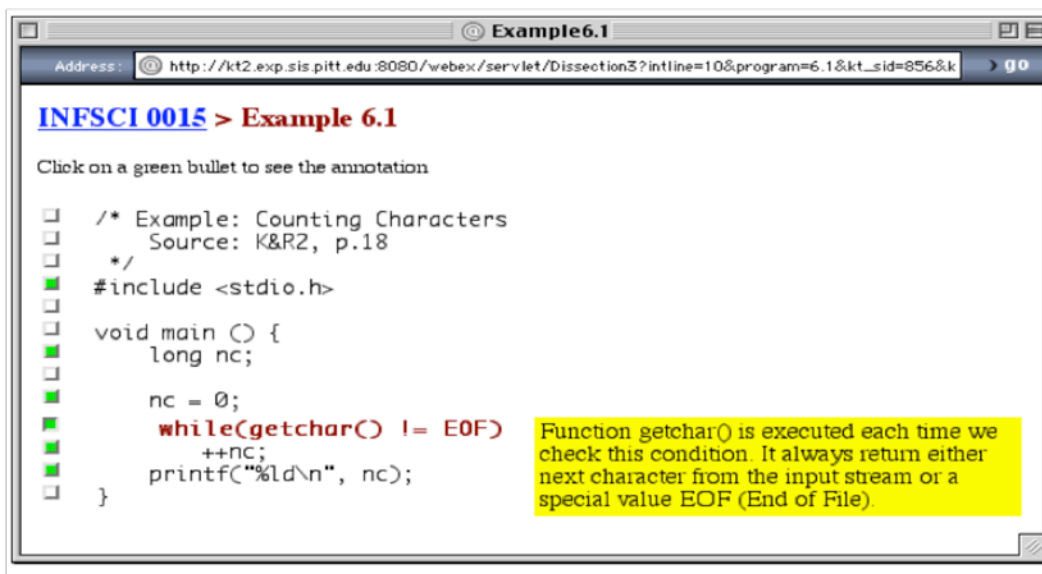


Figure 23: Typical interactive example in *WebEx*

Another application is *QuizPACK* - the system for authoring and delivery of parameterized web-based quizzes for C-programming courses (Brusilovsky and Sosnovsky 2005). Figure 24 demonstrates student interface of the system.

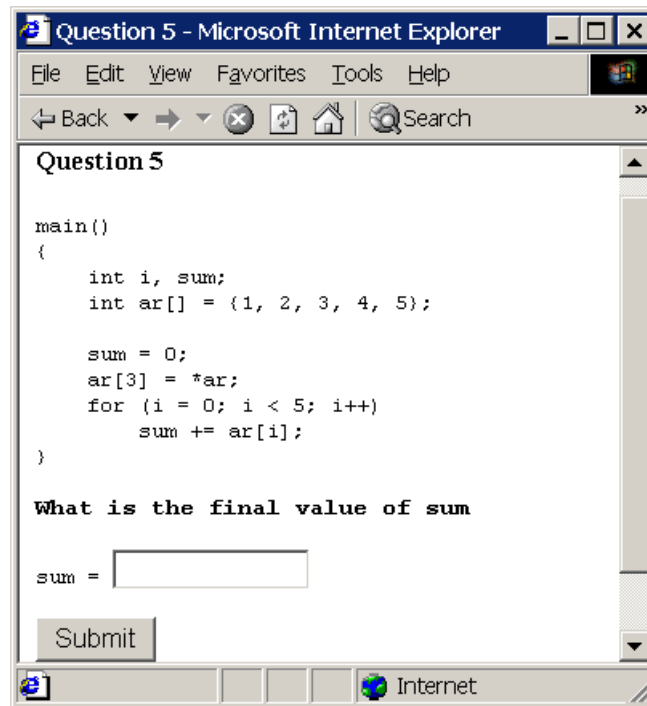


Figure 24: Typical self-assessment question in *QuizPACK*

The proposed algorithm for indexing programming code-based learning objects has two main stages. In the first stage, it extracts concepts from learning objects (examples, questions, presentation pages) combined by the teacher into an activity pool. For example, all *WebEx* examples form one pool; all *QuizPACK* quizzes belong to another pool. In the second stage, the prerequisite/outcome structure of the course is built in terms of concepts, describing content elements. This sequential structure, together with the indexed content, provides the basis for future content adaptation. The following two subsections explain both stages in details.

3.3.1.1 Parsing on-line C programming content

Extraction of grammatically meaningful structures from textual content and following determination of underlying concepts is a task for the special class of programs called parsers. For this approach, a parsing component was developed with the help of the well-known *UNIX* utilities: *lex* and *yacc*. This component could process source code of a C program and generate a list of concepts used in the program; the concepts come from a C ontology¹⁶. Each programming construct in the parsed content can be indexed by one or more concepts, depending on the amount of knowledge students need to have learned in order to understand the structure. For instance, the parser generates the following list of concepts for the program code of the WebEx example in Figure 23: “*include, void, main_func, decl_var, long, decl_var, assign, ne_expr, pre_inc, while, compl_printf*”. Each concept here represents not simply a keyword, found in the code, but a grammatically complete programming structure. For instance, concept *while* is recognized by the parser only after the entire *while*-loop (including the keyword “*while*”, the iteration condition and the body of the loop) has been found. This is why the concept *while* in the index list above follows the concepts *ne_expr* (*expression !=*) and *pre_inc* (*++pre-increment*).

The parsed code is accessed through the http-protocol. It can be represented as a simple source file in text format or as a properly formatted HTML-file that distinguishes the code samples from the rest of the HTML content with one of these commonly accepted tag pairs: `<code> – </code>`, `<pre> – </pre>`, or `<tt> – </tt>`. Other HTML-tags, used for formatting and presentation (e.g. ``, `<a>`, `<div>`) between the pair of code-tags do not influence parsing results. HTML escape-sequences (e.g. ` `; `<`; `&`;) are processed and converted into

¹⁶ Available at: http://www.sis.pitt.edu/~paws/ont/c_programming.rdfs

corresponding symbols. Hence, the developed indexing component can be used for indexing open-corpus on-line C content and help integrating it into an AES.

3.3.1.2 Prerequisite/outcome identification

The outcomes of the parsing stage are indices of learning objects. However, this is not enough, as these indices are not connected to each. In the next stage, concepts related to a single learning object are automatically assigned with prerequisite and outcome roles. Prerequisites are the concepts that the student needs to master before starting to work with the current learning object. Outcomes are the concepts that the learning object helps to learn, i.e. the learning goals of the element. These roles help to establish structural relationships between concepts (the same concepts play different roles within different learning objects) and pedagogical dependencies between learning objects, thus providing the basis for adaptive sequencing of learning objects.

The algorithm for the automatic identification of outcome concepts is presented in Figure 25. This algorithm is flexible enough to be influenced by an instructor-specific way of teaching the course. The source of knowledge for this algorithm is a sequence of groups of learning objects. Each group is formed by the objects introduced in the same lecture. Groups are ordered according to the order of lectures in the course, forming a sequential structure of learning goals in the course. The prerequisite/outcome identification is based on the following assumptions:

- All concepts associated with the learning objects that form the first group (first lecture) are declared the outcomes of the first lecture and are marked as outcomes in the index of all learning objects that form the first group.
- All concepts associated with the learning objects that form the second group (second lecture) are divided into lecture outcomes and lecture prerequisites. All concepts already listed as outcomes of the first lecture are defined as prerequisites of the second lecture.

They are marked as prerequisite concepts for each learning object in the second group. The concepts that were first mentioned in the second group become outcomes of the second lecture. They are marked as outcome concepts for each learning object in the second group.

- This process is repeated for each following group. The result of the process is a separation of prerequisite and outcome concepts for each lecture and each listed learning object. A by-product of this process is the identification of the learning goal (a set of introduced concepts) of each lecture.

The sequence of obtained learning goals represents the specific approach to teaching C-programming used by this specific instructor. Once the learning objects are indexed and the goal sequence is established, any extra piece of content can be properly indexed by the algorithm and associated with a specific lecture in the course. This would be the last lecture that introduces its concepts (i.e., the latest lecture, whose learning goal contains at least one concept from this learning object's index).

```
learnt_concepts =  $\emptyset$ 
for i = 1 to no_of_chapters
{
  for j = 1 to chapter[i].no_of_examples
  {
    chapter[i].example[j].prereq = learnt_concepts  $\cap$  chapter[i].example[j].all_concepts
    chapter[i].example[j].outcome = chapter[i].example[j].all_concepts  $\setminus$  learnt_concepts
  }
  for j = 1 to chapter[i].no_of_examples
    learnt_concepts = learnt_concepts  $\cup$  chapter[i].example[j].all_concepts
}
```

Figure 25: Pseudo-code for prerequisite/outcome identification

Although, the described indexing approach, using a parsing component, is specifically for programming and for the learning content based on the programming code (questions, code examples), the proposed general idea is applicable for a broader class of domains and types of content. The described approach has been implemented in a number of AES: *NavEx* (Yudelson and Brusilovsky 2005), *QuizGuide* (Brusilovsky, Sosnovsky et al. 2004), *SQL-Guide* (Sosnovsky, Brusilovsky et al. 2008), *Java-Guide* (Hsiao, Sosnovsky et al. 2010).

3.3.2 Interactive support for refining results of automatic indexing

Fully automated indexing is not always feasible. Some higher order concepts involve understanding programming semantics that might be hard to extract. In more advanced courses like *Data Structure* or *Algorithm Design*, pattern-oriented questions may be popular. For example, there are several modifications of the sentinel loop. The developed parser easily breaks such fragments of code into syntax concepts (which must be learned in order to understand the code); however, it will not be able to recognize differences in configurations of the sentinel loop. It also should be taken into account that the author might not fully agree with the results of automatic indexing. She may assume some extracted concepts to be irrelevant, or unimportant, or might want to add some other concepts.

Traditionally, there are two ways to support humans in performing complicated tasks: an AI approach (i.e., make an intelligent system that will do this task for the user) and an HCI approach (i.e., provide a better interface for the humans to accomplish the task). In the case of indexing, it means that one must either develop an intelligent system that can extract concepts from a fragment of content or develop a powerful interface that can help the teacher do this manually. While both approaches are feasible, this study advocates a hybrid approach – a “cooperative”

intelligent authoring system that splits the work between a human author and an intelligent tool. To put this idea into practice, a cooperative authoring system has been developed for the domain of C programming. The goal of this system is to allow authors collaboratively index interactive educational content (such as program examples or exercises) with domain model concepts while separating them into prerequisite and outcome concepts.

Figure 26 presents the interface for authoring *QuizPACK* parameterized questions. The main window is divided into two parts. The left part contains functionality for editing the text and different parameters of the question. The right part facilitates the elicitation of the concepts used in the question. It provides an author with non-exclusive possibilities: to extract concepts automatically and/or to use a visual indexing interface based on the visualized ontology of available concepts.

The developed ontology of C programming contains about 150 concepts. About 30 of them are meta-concepts (their titles are written in black font). An author cannot add meta-concepts to the index and may use them only for navigational purposes. Regular concepts can be present either in the index of the current question (blue font on the white background) or in the list of available concepts (light-blue squares). By clicking on labels of meta-concepts, an author navigates the ontology. By clicking on labels of regular concepts she adds them to the index, or removes if they're already there. The author can also use two lists below the ontology to alter index of a question. The concept lists and the ontology visualization are synchronized.

Figure 26 also tries to visualize the general workflow of the indexing process. First, the parsing component is used to automatically “Extract” the set of basic concepts in the question code. Then an author can “Edit” the set using the ontology-based interface. For example, on the figure the parser has been able to extract the concept *main-function*. The compound operator “{}”

is syntactically a part of the function definition; hence, the parser does not identify it as a separate concept. However, a teacher might want to stress that this is a particular case of compound operator and add this component by hand. The ontology-based interface for index refinement provides her with necessary functionality: concept *compound* is added to the index manually.

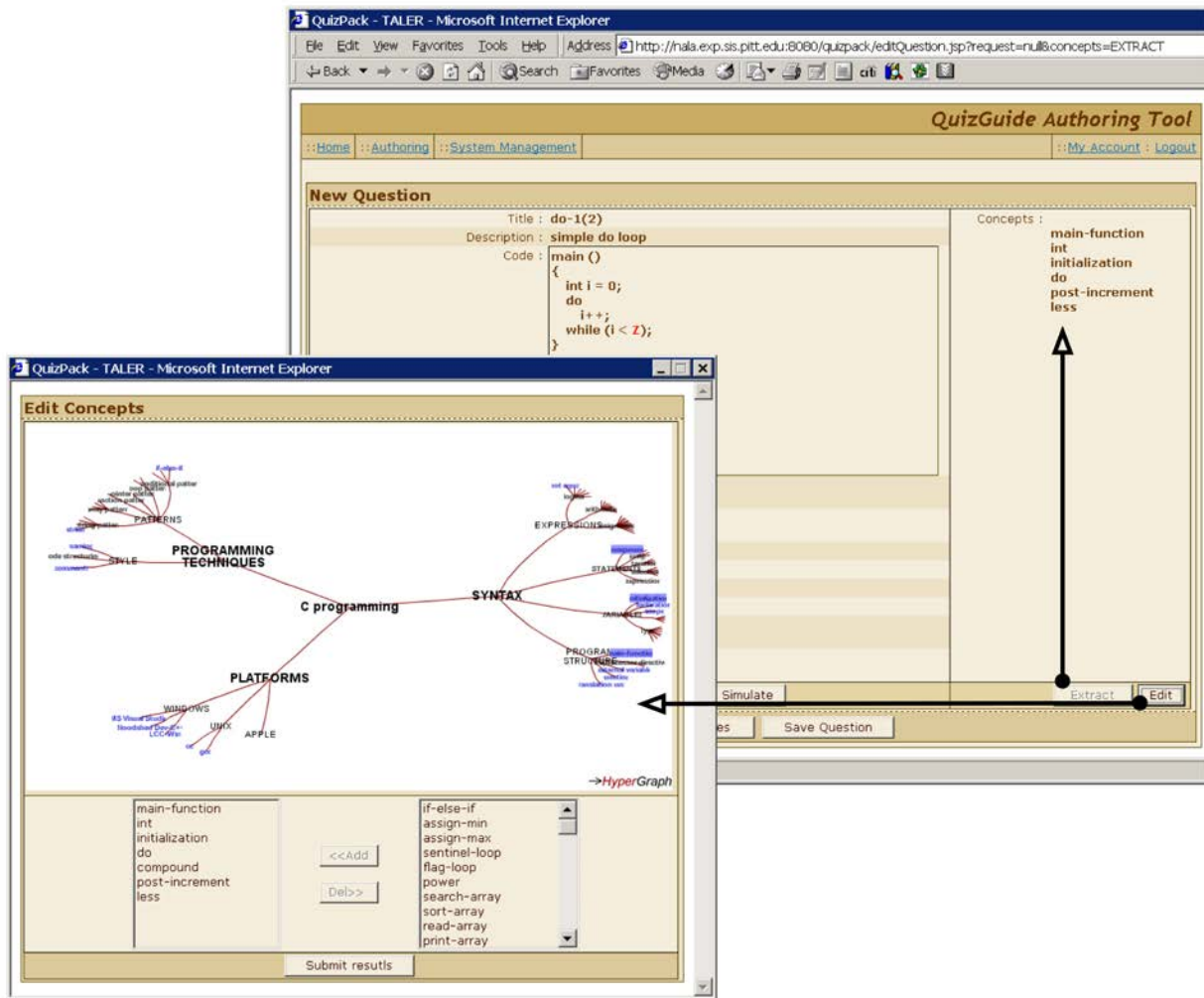


Figure 26: Using navigable domain ontology to refine the results of automatic indexing

3.4 COARSE-GRAINED CONTENT MODELING AND PERSONALIZATION

The complexity of adaptive content creation primarily originates in the complexity of the domain models used in modern adaptive educational systems. The more detailed and precise the modeling is, the more accurately a system can assess student knowledge; therefore, the more effectively it can potentially adapt its content to the individual student. However, the important questions are: Where does the “golden mean” lie? What is the best tradeoff between the model precision and model complexity, and between the effectiveness of adaptation and ease of development? Can we reduce the authoring effort without jeopardizing the quality of adaptation?

Collaboration with instructors interested in developing adaptive hypermedia content lead to an idea of exploring a rather minimalist *topic-based* approach towards domain modeling and content structuring. It is based on subdividing the subject to be taught into coarse-grain domain units (topics) and classifying each piece of learning content as belonging to exactly one of these topics. With this approach, the number of domain knowledge elements remains easy to manage, and the procedure of domain and content modeling essentially replicates the process of course design that many teachers follow. Separating the body of learning material into topics or lectures is a natural task for a classroom teacher preparing a course. It is also very common for a teacher to assign content (readings, problems, examples, demonstrations, etc.) to one of the lectures or topics. In fact, the success of modern Learning Management Systems (LMS), which offer a simple interface to create content folders and to sort all content into these folders, provide strong evidence that this approach is close to a teacher’s own conceptualization of the domain content.

3.4.1 Topic-based domain and content modeling in QuizGuide

An example of topic-based content modeling supported by the *QuizGuide* system is provided in Figure 27. Here, a course is structured into a sequence of *Topics* ordered along the course schedule. In simple cases, every *Topic* corresponds to a single lecture (or, in a more general sense, one of the top-level course *Goals*). Although, it is not a rule – a teacher can allocate as much time for a topic as required to cover a coherent set of ideas and concepts encapsulated in a topic. As a result, one Goal (lecture) can combine one or more smaller Topics. However, if necessary, a complex but coherent *Topic* can be spread over several *Goals*. In total, *QuizGuide*'s domain model consists of 22 *Topics* divided between 15 *Goals*. The educational content in *QuizGuide* is formed by a set of self-assessment *Quizzes* (their nature is explained in the next section). Following the topic-based content organization, each *Quiz* is associated with one of the course *Topics*. A *Quiz* is a complex educational activity – it consists of a sequence of individual traceable *Questions*. However, *Questions* can be accessed only sequentially within a *Quiz*; therefore, content assignment is done at the level of *Quizzes*.

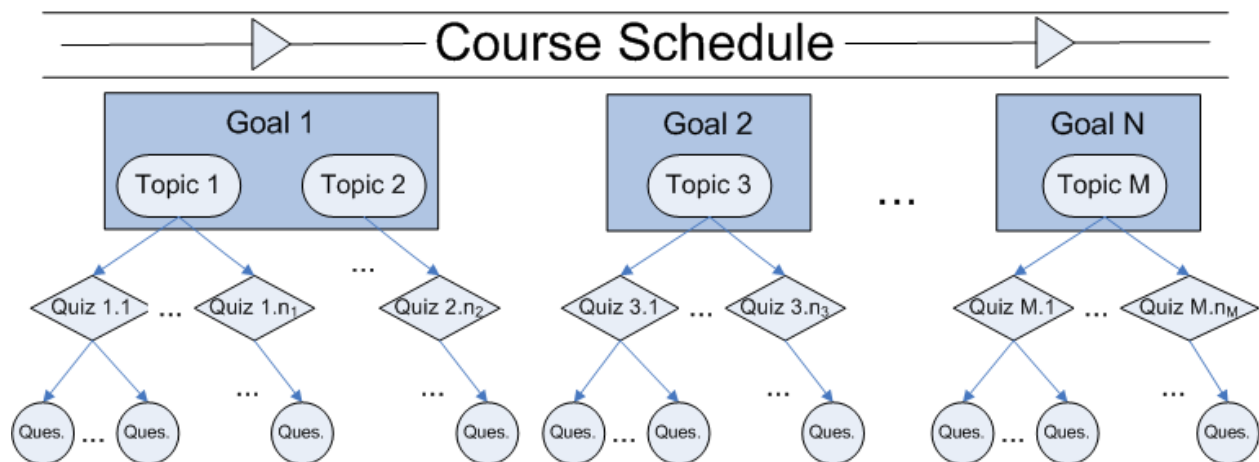


Figure 27: Topic-based content modeling in *QuizGuide*

Every topic in *QuizGuide* is essentially a bag of content (*Quizzes*). *Quizzes* can have various influences on the *Topics* to which they are assigned. *Quizzes* can differ in difficulty and coverage of the learning material important for mastering the topic. A teacher can model this contribution by assigning the percentage of the topical knowledge that a student can earn by completing a *Quiz*. Once all quizzes are completed the topic is mastered. Consequently, the content model of *QuizGuide* contains the list of *Topics*; each connected by weighted relations to corresponding quizzes (Figure 28).

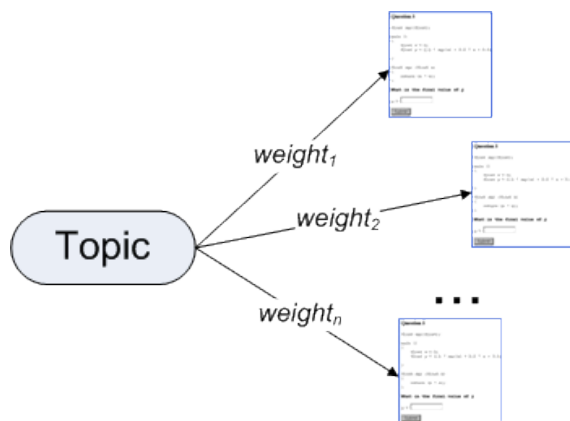


Figure 28: Topic-based content modeling in *QuizGuide*

The topic-based model of a course can be either a simple flat list of topics ordered along the course schedule or can have some structural relationships between the topics themselves. In order to take more advantage of goal-based adaptation (i.e., generate four goal states), the C-programming topics of *QuizGuide*'s model are connected to each other with the “*isPrerequisiteFor*”-relation. This relation does not model domain-driven structural connectivity, but pedagogically-driven dependencies between pieces of content. In other words, if *topic*¹ is connected to *topic*² with the “*isPrerequisiteFor*” relation, it means that some of the questions inside *topic*² require students to know learning material trained by some of *topic*¹'s questions.

For example, in the *QuizGuide*'s model, the topic “*loops (for)*” is a prerequisite for the topic “*arrays*” (see Figure 29). It does not imply that in every possible implementation of the C-programming domain, arrays require knowledge of for-loop; but for this particular course, students working with the quizzes on arrays need to have knowledge of for-loop. Topic-based modeling tries to speak in the language of a classroom teacher. It does not demand to understand the principles of domain semantics in order to connect topics to each other, but relies in teacher's ability to identify the pedagogical dependencies between topics used in the course.

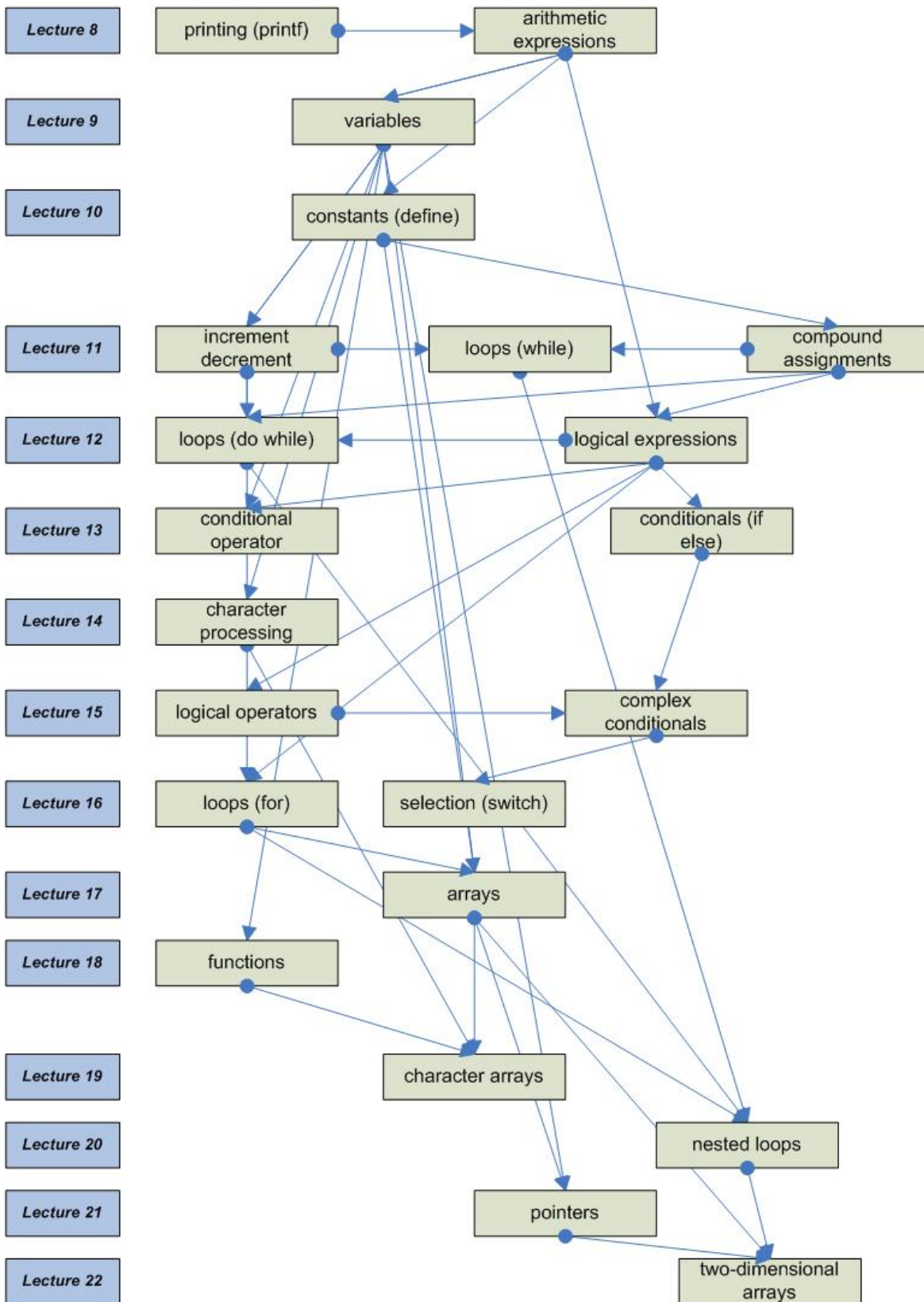


Figure 29: Topic-based structure of the C-Programming course: links represent *isPrerequisiteFor* relations

3.4.2 Topic-based student modeling and adaptation in *QuizGuide*

The main goal of our project was to demonstrate that a pure topic-based course organization could provide a basis for meaningful student modeling and adaptation. For modeling user knowledge, traditional weighted overlay student model was used (Carr and Goldstein 1977), where student knowledge is modeled as an overlay of the topic structure (i.e., for each topic, the system maintains an estimation of student knowledge of this topic). *QuizGuide* uses a numeric version of the overlay model (representing knowledge of each topic as a real number between 0 and 1) with no propagation between the topics; however, more sophisticated fuzzy or Bayesian models could be used, as well. The knowledge model is updated by tracking a student's activity with educational content. The update process is as simple and straightforward as the topic-based organization itself: each student's success or failure with a piece of educational content (a page is read, a question is answered, an animation is watched, etc.) causes a change in student knowledge level for the topic to which this content belongs.

It is important to stress that the topic-based student modeling is universal. While the implementation of this approach presented in this paper focuses on a specific implementation for the C programming, it can be used with little or no modification in many other domains and for other types of learning content. The point of this approach is to track user knowledge and goals as an overlay of domain topics and to use the topic-based content organization to map user actions to user knowledge. Figure 30 visualizes the topic-based user modeling and adaptation approach as a general framework. Within this framework, a specific adaptation approach was implemented that uses adaptive link annotation as the key adaptation technology.

The idea of topic-based adaptation in *QuizGuide* is guiding students to most appropriate topics by presenting the content of individual student's user model in the form of navigational

cues. Every topic link in *QuizGuide* is annotated with an icon representing the current state of a student's knowledge of learning material associated with the topic and the goal state of this topic. As a result, a student is constantly aware of his/her performance and is able to focus on those parts of the course most important to her/him. This adaptation approach can be considered as an interesting combination of open learner modeling (Bull, Dimitrova et al. 2007) and adaptive navigation support (Brusilovsky 2007).

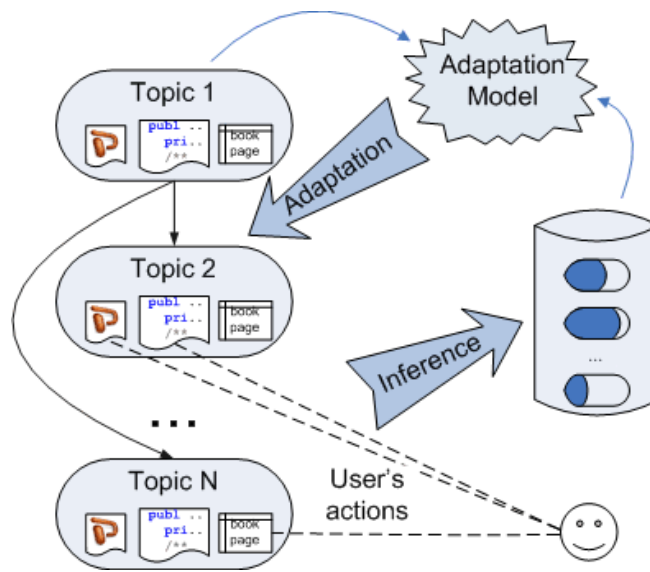


Figure 30: A topic-based adaptive system

Figure 31 presents a snapshot of the *QuizGuide* student interface. The left frame provides students with the annotated list of links to all topics available in the course. A click on a topic link reveals/hides links to one or more quizzes available for this topic. It allows students to organize their working space by opening only the quizzes of interest to them. The right frame of the *QuizGuide* interface contains the question that a student is working on. All quizzes and questions used in this study were provided by the *QuizPACK* system.

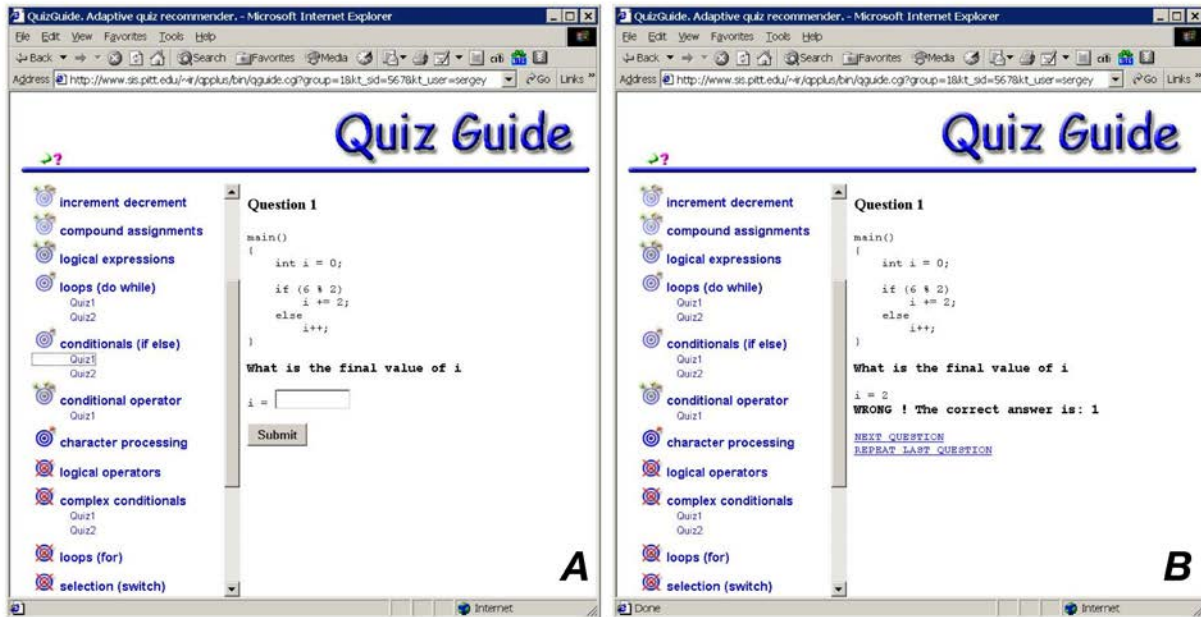


Figure 31: Interface of *QuizGuide*

For topic link annotation, *QuizGuide* uses large “target-arrow” icons (Figure 32 summarizes them). The icons deliver two kinds of information to a student:

- individual performance of the student with the topic’s content, and
- relevance of the topic to the current learning goal within the course.

The *number of arrows* (from 0 to 3) in the target reflects the amount of knowledge demonstrated for the topic. It is a simplified view of the student knowledge model (a mapping of student knowledge into a simpler 0-3 integer scale). If no, or very little progress has been made on the topic, the target icon for this topic will be empty, which invites the student to concentrate on this topic. Once the student starts giving correct answers to questions in any of the topic’s quizzes, the target-icon of the topic “receives” arrows, indicating the increase in student knowledge. After a sufficient number of correct answers, the system collects enough evidence of student knowledge to annotate the topic with the “3-arrows target”, the highest level of mastery.



Figure 32: Topic-based annotations in *QuizGuide*

The *color* of the topic icon designates the relevance of the topic to the current learning goal. As new topics are introduced by the teacher of the course, *QuizGuide* annotates them with bright-blue icons representing the current learning goal of the students. Topics that have not yet been introduced in the course are annotated with crossed-out target icons hinting that the student might not be ready for them yet. Topics that have been introduced earlier in the course are no longer the focus of learning. *QuizGuide* indicates so by annotating them with two different pale colors. Topics that are prerequisites for any of the current learning goals are marked with pale-blue target icons. If a student has insufficient knowledge of these, he/she most probably will have problems with the current topics as well. Pale blue annotations help to identify this situation. Past topics, which are not prerequisites to the current topics, are annotated by grey targets. Note that the availability of topic prerequisite allows distinguishing four goal states of a topic instead of a regular three.

It is easy to observe that topic annotation in *QuizGuide* combines two kinds of adaptation: individual progress-based adaptation and group goal-based adaptation. *QuizGuide* annotations inform the students about the individual and group-level state of the topics behind the annotated links, trying to direct students to the best learning content at any particular moment of time. At the same time, *QuizGuide* does not restrict access to the learning content in any way. The students can access any topics, even those that have not been introduced yet.

QuizGuide employs icon-based (vs. text-based) link annotation, which helps students to immediately recognize the topics' state, but which also requires some time to allow the students to remember the meaning of all annotations. To help the students learn and understand the annotations, *QuizGuide* dynamically generates mouse-over hints for the icons, explaining the meaning of the current state in natural language. A detailed help explaining all interface elements is available as well.

3.4.3 Evaluation of topic-based personalization

Topic-based approach is driven by the idea to reach a compromise between the complexity of content organization and user modeling from one side and teacher's ability to structure the learning content and understand the adaptation mechanism. The result of it, however, is a considerable simplification of commonly used concept-based adaptation approach. Both the domain model and the content model have been more coarse-grained and suggested relatively simple user modeling and personalization algorithms. Hence the important question is: *has not this approach gone too far, ending with a methodology that is more transparent for teachers, but fails to support meaningful user modeling and personalization?*

3.4.3.1 Evaluation context and data collection

The evaluation of *QuizGuide* has been performed in the context of an undergraduate course “*Introduction to Programming*” offered at the School of Information Sciences at the University of Pittsburgh. Main part of this course was dedicated to introductory C programming covering the basics of variables, data types, control structures, character processing, arrays, pointers, and functions. Overall, 22 topics (Figure 29) were taught in the course of 15 lectures. Each lecture

was defined as a learning goal in *QuizGuide*. 155 *QuizPACK* questions were developed and combined into 44 quizzes to cover the course material. The topics incorporated between one and three quizzes. The quizzes contained from three to five questions.

The evaluation consisted of several studies of *QuizPACK* and *QuizGuide* over the period of four consecutive semesters. The syllabus of the course, the set of topics, the course books, the in-class quizzes, the homework assignments, and other characteristics of the course did not change during this time. In the course, *QuizPACK*/*QuizGuide* systems were used as a supplementary learning tool. The systems were introduced to the students at the beginning of the course. Through the course, the students were encouraged to use them, however the use of the systems was fully voluntary and did not influence the course grades. In the first two semesters, students were using only non-adaptive quizzes served directly by the *QuizPACK* system through a simple course portal (Figure 33). In the middle of the third semester, *QuizGuide* was introduced. After that, the students had the option to use questions with adaptive navigation support (through *QuizGuide*) or without adaptive navigation support (through the course portal). The set of quizzes and questions, as well as the structure of topics and lectures (learning goals), were the same for adaptive and non-adaptive modes and did not change over the time of the experiment.

While most of the students taking the course never formally studied programming before, some students were capable of writing and launching simple programs prior to the course. To take into account students' starting level of domain knowledge, in the beginning of the course, they were asked to rank their own programming experience on the scale (“novice” – “medium” – “knowledgeable”).

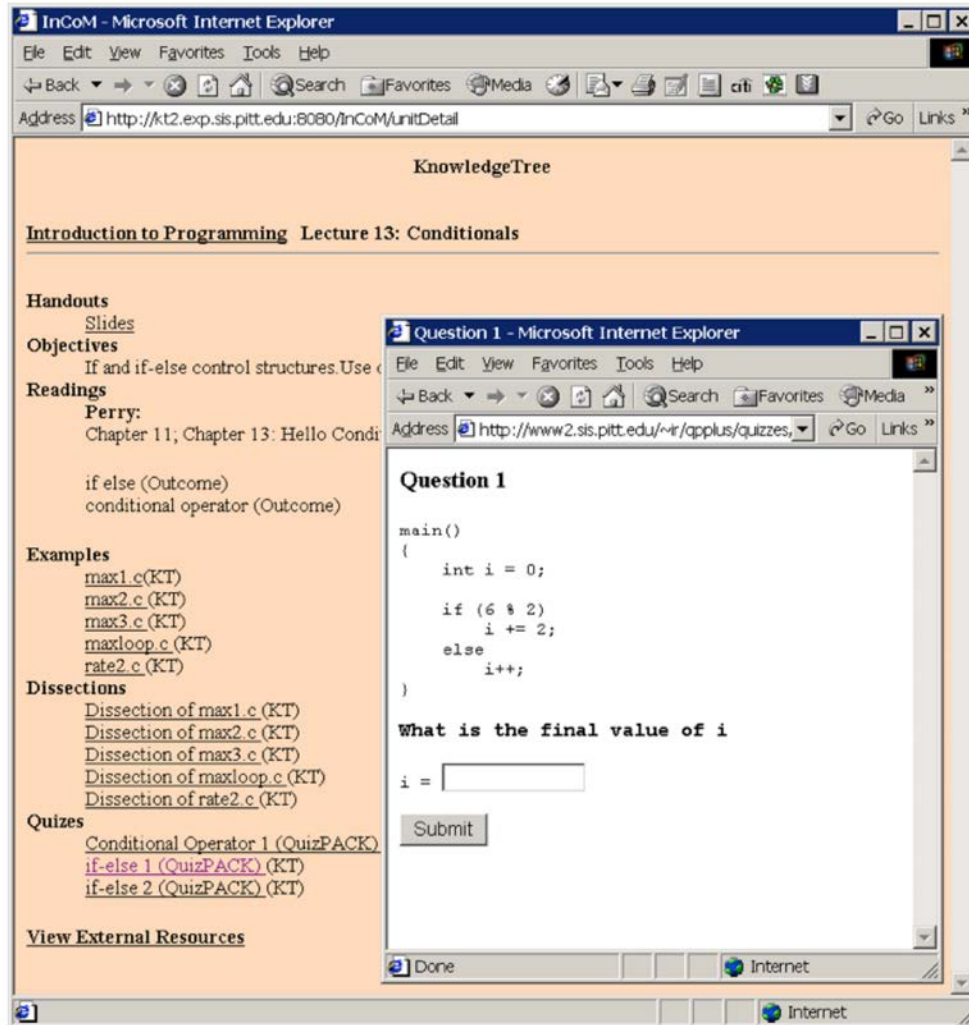


Figure 33: Non-adaptive access to quizzes

To estimate students' involvement in the course their attendance throughout the semester was collected. Table 2 summarizes the main descriptive parameters of the student groups. To ensure that the effects observed during the evaluation cannot be attributed to the differences in student population across semesters, but indicate the added value of adaptive annotation, the variance in distribution of these parameters was analyzed. The results of the Chi-Square test show that there was no significant difference between *QuizPACK* and *QuizGuide* groups in gender distribution (*Pearson Chi-Square* = 2.014; *p-value* = 0.156) or self-assessed

programming experience distribution ($Chi\text{-}Square = 4.429$; $p\text{-}value = 0.109$). The Mann-Whitney-Wilcoxon test shows that the class attendances of *QuizPACK* students ($M=0.7994\pm0.025$) and *QuizGuide* students ($M=0.7687\pm0.031$) do not significantly differ from each other ($Mann\text{-}Whitney U = 1270.5$; $p\text{-}value = 0.181$).

Table 2: Students' gender, experience and attendance distribution

Semesters		Number of students	% Female	Self-assessed programming experience (on a scale 1-3)	Attendance
Non-Adaptive semesters	Semester-1	45	1.68	1.58	77.96%
	Semester-2	28	2.29	0.57	83.13%
	Total (Non-Adaptive)	73	1.97	1.15	79.94%
Adaptive semesters	Semester-3	27	2.19	0.2	72.56%
	Semester-4	14	2.18	0.69	85.19%
	Total (Adaptive)	41	2.19	0.39	76.87%

3.4.3.2 Learning effect

For evaluating the impact of adaptive annotation on students' learning, their knowledge gains over the period of using the system were calculated based on the results of pre- and post-quizzes. These quizzes consisted of 10 code evaluation questions (similar to those used in the system) covering most of the course material. The pre-quiz and the post-quiz questions were identical except for numeric values, which ensured that the correct answers were different. The pre-quizzes were taken by the students in the very beginning of the semester to estimate their initial

level of expertise in the subject. The post-quizzes were administered after the last lecture to measure the increase in knowledge over the semester. The format and the content of the quizzes stayed generally the same over all five semesters.

The knowledge gain was calculated in two different ways. According to the canonical formula it is the difference between the corresponding results on a pre-quiz and a post-quiz:

Equation 1: Knowledge gain

$$**KG = PostScore - PreScore**$$

This formula sometimes is criticized for not adjusting to the differences in the students' initial knowledge levels. Therefore, another formula was introduced for calculating normalized knowledge gain (Hovland, Lumsdaine et al. 1949):

Equation 2: Normalized knowledge gain

$$**NKG = \frac{PostScore - PreScore}{1 - \frac{PreScore}{MaxScore}} = \frac{PostScore - PreScore}{MaxScore - PreScore}**$$

As the evaluation showed, students working with “adaptive” version of the system achieved higher knowledge gain. There was a significance difference in knowledge gain for adaptive ($M=6.44\pm0.40$) and non-adaptive groups ($M=5.09\pm0.27$); *Mann-Whitney U* = 693.0; *p-value* = 0.009. The effect also holds for the normalized knowledge gain: for the students working with QuizGuide ($M=0.67\pm0.04$) it is significantly higher than for the students working with QuizPACK ($M=0.56\pm0.03$); (*Mann-Whitney U* = 749.0; *p-value* = 0.032). This impact on learning can be easily explained when it is considered together with the motivational effect of the topic-based navigation support described in the next section.

3.4.3.3 Motivational effect

The most dramatic effect of topic-based adaptation was not on students' learning but on their motivation. The introduction of the adaptive services caused an impressive increase of students' interaction with the supported content as compared to their work with the non-adaptive interfaces in previous semesters. With *QuizGuide*, the students explored more questions, worked with questions more persistently, and accessed a larger diversity of questions. In some sense, adaptive annotations made the quiz work almost addictive. Once the students started a session, they stayed with the system much longer. The average session length and average number of questions attempted during a semester increased significantly.

The log data from two adaptive semesters (3&4) was compared with the data from two non-adaptive (1&2). The logs recorded every user click (i.e., submitting an answer to a quiz question). Data collection procedures did not differ across the discussed semesters and were not dependent on the method of student access to quizzes (whether via adaptive or non-adaptive systems). Student work with any of the discussed systems was included equally for user modeling. Log data gave clear evidence as to whether a student accessed quizzes through the adaptive service or not.

The following three variables were used to parameterize student performance:

1. activity: the number of attempts to answer a quiz question;
2. quantity: the number of quizzes taken, and
3. coverage: the number of lectures that the attempted quizzes were drawn from.

Each of these variables was aggregated on two levels:

1. overall performance level – the total number of attempts made, quizzes explored, and lectures covered by each user over the course of the semester; and

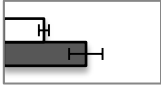
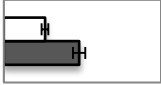
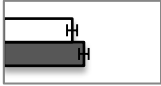
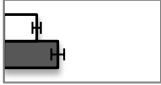
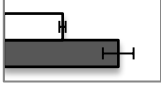
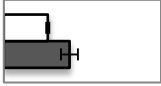
2. session performance level – the average number of attempts made, quizzes explored, and average number of lecture topics explored per session by a user.

The goal was to support our subjective observation that adaptive guidance does make a difference, i.e. to determine whether activity, quantity and coverage of topics was higher for students who were exposed to *QuizGuide* than for those who used the non-adaptive *QuizPACK* format. The results of our comparison demonstrate the value of the adaptive navigation support in motivating students to interact more with self-assessment quizzes.

During this analysis, approximately 20,250 *QuizPACK* and *QuizGuide* user actions (question attempts) were explored. The distributions of data across all our variables were severely skewed because there was a number of not very active students (in terms of attempts made), fewer moderately active students, and very few very active students. This, along with heterogeneity of variances, prevented us from applying parametric statistical tests to the comparison of usage data. Instead, Mann-Whitney tests were employed as *t-tests'* substitutes.

The results of these tests have shown that for nearly all variables and aggregation levels, users exposed to the adaptive features of *QuizGuide* achieved significantly higher results (Table 3). As the comparison of the levels of overall parameters demonstrates, users working with adaptive guidance were making *twice as many* question attempts per semester (an average of roughly 260 vs. 130). This increase of non-mandatory activity was statistically significant. The increase of course coverage was visible, but not statistically significant. Unlike other usage parameters, course coverage has a natural maximum boundary – the total number of lectures in the course. On the level of average user session statistics across all variables, the parameters of *QuizGuide + QuizPACK* combined were all roughly 1.5–2 times higher than those of *QuizPACK* alone. All of the observed increases were significant.

Table 3: Basic statistics of using the system (adaptive vs. non-adaptive)

		Non-adaptive (<i>semesters 1&2</i>)	Adaptive (<i>semesters 3&4</i>)	p-value	Graphical representation
Overall user statistics	Activity	127.68±15.97	261.21±53.15	0.023*	
	Quantity	13.11±1.06	23.97±1.96	<0.001	
	Coverage	8.70±0.64	10.18±0.61	0.188	
Average user session statistics	Activity	10.48±1.32	17.19±2.03	<0.001	
	Quantity	1.87±0.10	3.64±0.49	<0.001	
	Coverage	1.40±0.05	2.09±0.27	<0.001	

* boldface indicates p-value less than 0.05

3.4.3.4 Evaluation of topic-based user modeling: topic as an assessment unit

The chosen modeling approach, as well as the implemented adaptation strategy, could be reasonable; however, the resulting adaptive behavior might not be adequate to the student's actions and expectations if the assessment units are wrong. For evaluation of large topics as assessment units, learning curve analysis was applied (Newell and Rosenbloom 1981). Multiple experiments provide strong evidence that the learning process generally follows the power law (see for example (Anderson, Corbett et al. 1995; Mitrovic, Mayo et al. 2001)). In other words, the error rate of a learning skill decreases as the power function of the number of learning steps involving this skill.

To compute topic-based error rates, all question attempts were categorized per student, per topic. Hence, if a student gave 20 answers to question on *Arithmetic Expressions*, 20 entries have been created in the order the answers were given; the actual questions did not matter (it could be 20 answers to the same question or to 15 different question on the topic of *Arithmetic Expressions*). The maximum number of learning steps per student per topic was 214; the average was 11.86. Equation 3 was used to compute *Average error rates* for every *Learning step*:

Equation 3: Average error rate

$$\text{AverageErrorRate} = \frac{\text{NumberOfCorrectAttempts}}{\text{TotalNumberOfAttempts}}$$

Figure 34 demonstrates the dependency between the *Average topic-based error rate* and the number of *Learning steps*.

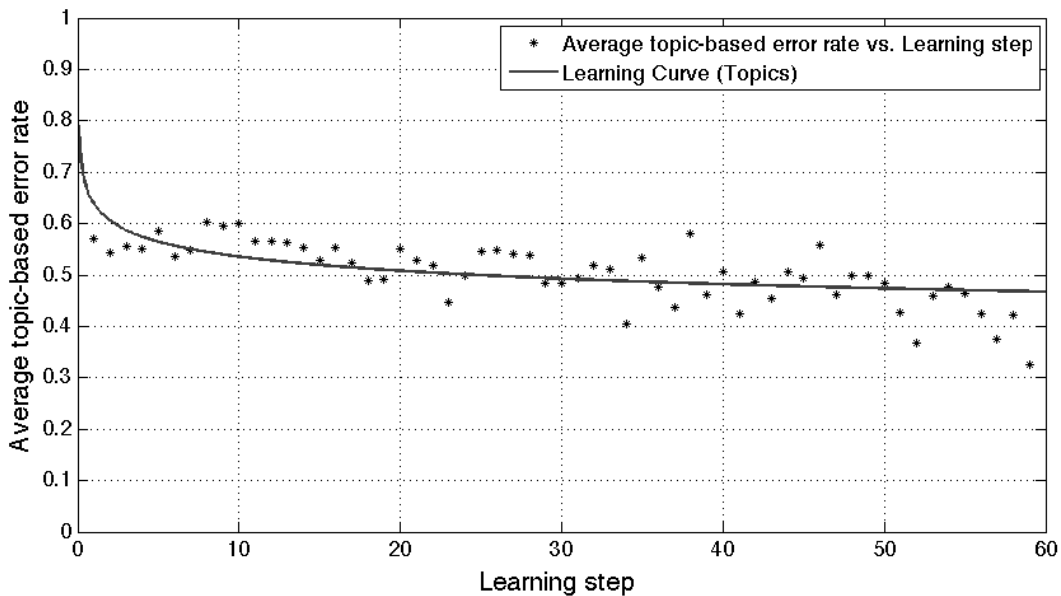


Figure 34: Topic-based learning curve

Equation 4 defines the power-law learning curve approximating this dependency. Though the downward trend shows some learning effect, the curve is not smooth and R^2 -statistics indicates that only about 39% of the variability in the *Average topic-based error rate* could be explained as the power law on the number of *Learning steps*.

Equation 4: Learning curve for topic-based average error rate

$$\mathbf{ErrorRate = 0.64 * LearningStep^{-0.07}; (R^2 = 0.39)}$$

The main feature of topics that differentiate them from the well-established concepts is that topics are large; they aggregate too much learning material and cover too much knowledge. To see, how it influenced the quality of topics as assessment units, the same learning-curve analysis was performed on the smaller chunks of learning material, namely quizzes and questions (every *QuizPACK|QuizGuide* topic combine on average 2 quizzes and approximately 7.5 questions). Equation 5 and Equation 6 demonstrate that by narrowing the scope of assessment units much better results can be achieved. The dependency between the *Error rate* and the number of *Learning steps* for an average quiz (Equation 5) and an average question (Equation 6) demonstrate better learning. Approximation of these plots with power law learning curves results in better fits (70% of the variability in the question *Error rate* is explained by the power dependence on the number of *Learning steps*). These data indicates that topics cover too much learning material to adequately reflect the changes in student knowledge.

Equation 5: Learning curve for quiz-based average error rate

$$\mathbf{ErrorRate = 0.66 * LearningStep^{-0.12}; (R^2 = 0.48)}$$

Equation 6: Learning curve for question-based average error rate

$$\mathbf{ErrorRate = 0.64 * LearningStep^{-0.21}; (R^2 = 0.70)}$$

Since topics involve too much knowledge, the precision of the assessment performed by the system is reduced. When taking a quiz on a particular topic, a student practices a number of interrelated concepts used by the questions in the quiz; a large topic is not capable of reflecting subtle changes in student's knowledge resulting from this practice. Hence, one of the potential solutions for topic-based modeling is to keep them reasonably small.

Another approach for improving topic-based knowledge modeling is to increase the predictability of students' results by harmonizing the assessment material. One of important aspects of learning content representation that our simplistic topic-based model missed on is the difference among questions in difficulty and structural complexity. A topic covering a dozen of questions treats all of them identically, even though students are much more likely to answer easy and simple questions correctly. By reducing of the variability in question difficulty one could improve the predictability of students' answers and hence improve the assessment characteristics of a topic. From another point of view when the question is too hard or too simple, the effectiveness of both learning and knowledge assessment is reduced.

To investigate this problem, the average question difficulty was estimated using the traditional measure – the mean value of error rate. The difficulty value ranged from 0.0645 to 0.8947, with mean value = 0.4588 and median value = 0.4699. The data was examined on two intervals: 90% and 50%. Equation 7 describes the very same learning curve as Equation 4 where 5% of the most difficult and 5% of the least difficult questions are removed from the computation of the average error rate. Equation 8 describes n the same curve, when questions below the 25th and above the 75th difficulty have been filtered out. The error rate variability explained by the power dependence on the number of learning steps almost double compared to the non-filtered topic-based learning curve.

Equation 7: Learning curve for topic-based average error rate (filter= 90%)

$$\mathbf{ErrorRate = 0.65 * LearningStep^{-0.08}; R^2 = 0.60}$$

Equation 8: Learning curve for topic-based average error rate (filter= 50%)

$$\mathbf{ErrorRate = 0.69 * LearningStep^{-0.13}; R^2 = 0.67}$$

There are several possible outcomes of this analysis that can be applied in order to improve the topic's knowledge modeling quality. More careful design of assessment questions or more accurate manipulation of presented question difficulty, or even providing students with reliable information about the question's characteristics can help to decrease the number of attempts of too hard or too simple questions, which can improve the learning process guided with the help of topic-based adaptation.

3.4.3.5 Evaluation of topic-based user modeling: predictive validity

The ultimate task of a student model is to represent the state of a student as close to reality as possible, in order to help the adaptive educational system anticipate student's actions and optimize learning process. If the predictions of student model do not correlate with what students demonstrate in reality, then the value of such a model is rather questionable. When it comes to the evaluation of student model's predictive validity, two components should be taken into account: the modeling units, and the modeling formula. As seen from the previous section, a topic is not an optimal modeling unit. To compute the correlation between the modeled values and actual users' states, for all learning steps of every student two measures were computed:

- the prior knowledge level for the topic;
- the immediate posterior knowledge level estimation for the topic.

The prior knowledge level is computed by the *CUMULATE* topic-based modeling component from the history of this student's interactions preceding the current learning step. Estimation of the actual student's performance right after the learning step has been computed based on the average score of 5 next attempts of the student to answer the questions belonging to the same topic. The results of the correlation analysis show that the ability of the topic-based knowledge model used by QuizGuide to predict immediate student's performance is fairly low: Pearson correlation coefficient = 0.22.

3.4.3.6 Evaluation of topic-based adaptation

In order to see how the progress-based adaptation of *QuizGuide* affects students' choices of quizzes to work with, one needs to take a closer look on what constitute these choices. At every moment of time, a student deciding, which question to try next, has 22 course topics to select from. The user model keeps track of student's history and can compute her current progress for all the topics. A student accessing questions through QuizGuide is constantly informed of her progress for all the topics by being presented with a combination of "0-3 arrows"-annotations. The "optimal" choice is to always prefer topics with a lower level of annotation¹⁷.

The easiest way to compute a metric showing how well a student follows the navigation would be by simply counting how many times topics with a certain number of arrows on the icon have been accessed. Unfortunately, this metric does not account for the fact that at every moment the numbers of topics with different levels of annotations are very different. In the beginning, all

¹⁷ This is a simplification, as students also take into account the goal-based annotation when choosing topics. However, in order to assess the roles of topic-based and goal-based adaptation separately, we have to simplify this analysis.

topics are annotated with zero arrows, as a student has accumulated no progress yet; while at the end, she really has a choice between “good” (few arrows) and “bad” (many arrows) topics.

Hence, instead of concentrating on the fact that a topic with a particular navigation state has been chosen, this analysis concentrates on the decisions that students have made, taking into account not only the level of the chosen topics but also, the levels of all other topics in the course. The student modeling process was reran on the logs of non-adaptive and adaptive courses, and at every step for every user the snapshot of her entire topic-based progress model was recorded. As a result on every step the quality of a navigational decision a student has made was estimated; two different metrics were computed in order to quantify the average percentage of “good” navigational decisions made by every student.

The *metric-A* is an aggressive metric; it assumes that students should always make the best decisions possible. Only if a student chooses a topic with the lowest level of annotations, the decision is labeled as a good one. If there are topics with lower levels, it is a “bad” decision. If there are no alternatives (all topics have the same annotation level), then it is not a decision at all – no labels are assigned.

The *metric-B* is a lenient metric; it assumes that as long as the decision made by a student is not the worst, she should be credited for it. As long, as there exist a topic with a higher-level annotation, the chosen topic constitutes a good navigational decision. Only if the topic is annotated with the maximum-level annotation, and there are topics with lower numbers of icons, the decision is labeled as a “bad” one.

Both metrics have been computed for all students from the non-adaptive groups (semesters 1&2) and adaptive group (semesters 3&4). Table 4 presents the results of a *t*-test: according to the both metrics the adaptive group has made on average significantly more good

navigational decisions than the non-adaptive group. This means, in the presence of progress-based adaptive navigation, students tend to choose topics with lower-level annotations, thus following the adaptive guidance as intended.

Table 4: Quality of progress-based navigation: comparing the percentage of “good” navigational decisions

Metric	Non-adaptive (Semesters 1&2)	Adaptive Semesters (3&4)	t-statistics	p-value
<i>A. Aggressive</i>	0.23±0.03	0.39±0.05	2.987	0.004
<i>B. Lenient</i>	0.55±0.03	0.69±0.03	3.281	0.002

3.4.3.7 User behavior: a global picture

While comparing student work with adaptive and non-adaptive versions, it was noticed that typical *QuizGuide* sessions are both *longer* and *more diverse* than *QuizPACK* sessions. Students attempted more questions through *QuizGuide* and more frequently accessed material corresponding to different lectures within the same session.

To take a closer look at the nature of these results, further analysis of student activity was performed, taking into account lecture coverage included in all students’ actions. Every selection of a question was attributed to the lecture (or the learning goal) it belongs to. For example, Figure 35 visualizes over 5,500 attempts performed by students using *QuizGuide* or *QuizPACK* in the *Fall 2004* semester. Fifteen lectures form the vertical axis. The time of the action is marked on the horizontal axis. One can detect three zones of activity. The zone “A” contains all of the “current” activity that students perform along the lecture stream of the course. It is fairly broad, since homework assignments and in-class quizzes introduce 1-2 weeks delay in shifting

the students' focus from the previous topics. Zone "B" contains a period of preparation for the final exam. The pattern of work with the system is completely different during this stream of time. Finally, zone "C" contains all actions that students performed during the regular part of the semester, for topics laying far from the "current" lectures. This zone is particularly interesting. All actions here are not directly motivated by the "current" course situation, but rather initiated by the students themselves, possibly in an attempt to bridge the gap in their knowledge that should have been acquired earlier.

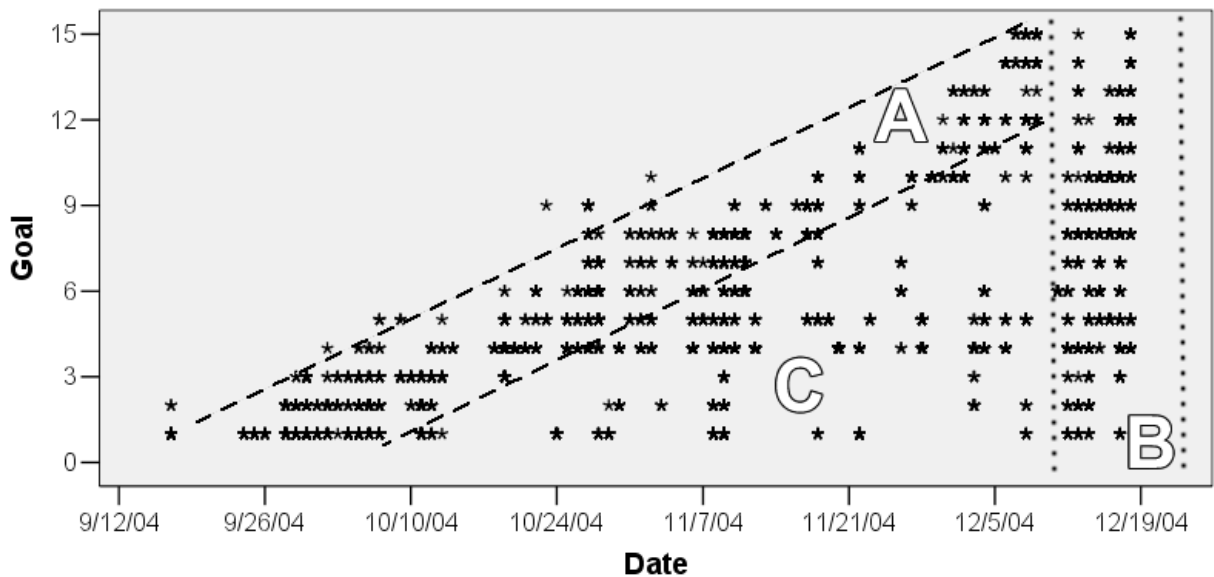


Figure 35: Time distribution of students' activity with QuizGuide and QuizPACK in the 4th Semester¹⁸

The intensity of students' *self-motivated* activity was assessed based on two measures: the number of actions in zone "C" divided by the total number of actions and the average distance between the learning goal, which is current at the time of a question attempt, and the

¹⁸ Zone "A" – lecture stream, zone "B" – final exam cut, and zone "C" – self-motivated work with the material of earlier lectures

learning goal, to which the attempted question belongs. For the calculation of the second measure, both zones “A” and “C” were used.

An average of “C”-ratio and goal distance were computed for all students. They were again divided into two groups: *non-adaptive* (semesters 1&2) and *adaptive* (semesters 3&4). The results of Mann-Whitney’s test are presented in Table 5. Both measures are significantly higher for the adaptive group, which means that students in the semesters when *QuizGuide* was available more willingly accessed non-current activities.

Table 5: Parameters characterizing the self-motivated activity of students with and without adaptive annotations¹⁹

	Non-adaptive	Adaptive	p-value
“C” ratio	0.20±0.03	0.28±0.04	0.025
Goal distance	5.89±0.84	9.56±1.61	0.026

The observed effect can be explained by the fact that progress-based and prerequisite-based adaptive annotations generated by *QuizGuide* directed students’ attention to the material related to those earlier lectures which were not understood. The main stream of activity stays in the zone of recommended work (zone “A” on **Error! Reference source not found.**), which is moving along *class progress* – topics are annotated as ready and current in the week they are presented in a lecture. However, a clear indication is provided for topics that are prerequisites for current topics. Students struggling with a current topic can easily focus on prerequisite topics

¹⁹ The “C”-ratio estimates the percentage of students’ activity performed outside the current course focus, while the Goal distance assesses how broadly roams (in terms of learning goals) the voluntary interest of a student who is working with the system

that were not well understood, according to the arrow progress measure. The progress-based annotation can also motivate students to review the previous topics and earn annotations of higher levels.

3.5 MEDIATION OF EXTERNAL MODELS

Over the last ten years, a number of adaptive educational systems have migrated from research labs to classrooms and are now employed by thousands of students. In some domains, the “density” of AES is reaching the point where several systems are available. Yet, in most cases, these systems do not compete, but rather complement each other, while offering unique functionality or content. This makes the problem of using several AES in parallel, an important practical task. It has been explored by several research teams and from several perspectives: architectures for integrating adaptive systems (Brusilovsky 2004), cross-system personalization (Niederée, Stewart et al. 2004; Carmagnola and Dimitrova 2008), user model ontologies (Heckmann, Schwartz et al. 2005; Dolog and Nejd1 2007), and user modeling servers (Kay, Kummerfeld et al. 2002; Kobsa and Fink 2006; Yudelso, Brusilovsky et al. 2007).

From the point of open-corpus content modeling and personalization, the task of interfacing two AESs would require finding correspondence between the models of the two systems in such a way that students’ activity with the content of one system could be traced by the other. This task becomes a variant of external content modeling when external content comes supplied with its own model. Presence of an external content model potentially helps to achieve better modeling of open-corpus content, as it will reuse the existing links between the elements of the external domain model and the external content that have been created by an expert at the

design time. However, the difference between the domain models can be so drastic, that their alignment becomes unfeasible.

In order to verify the mere effectiveness of such content-based integration when the domain models diverge very strongly of each other, two experiments have been organized on manual integration of existing systems. First, in the domain of Java programming, the *QuizJET* system has been integrated with the *Problets* mini-tutors. Then, in the domain of SQL, the adaptive navigation service *SQL-Guide* has been integrated with the constraint-based ITS *SQL-Tutor*. The domain models employed by *QuizJET* and *SQL-Tutors* are regular ontologies, while the domain model of *Problets*, and, especially, on of *SQL-Tutor* have very different nature.

The last experiment described in this Section present an example of the automatic integration of two homogeneous domain models – two RDFS ontologies. It applies an ontology mapping technique to align parts of ontologies for C and Java programming and then uses this alignment for translating overlay student models between the two domains.

3.5.1 Manual mapping of a domain ontology and a pedagogical model

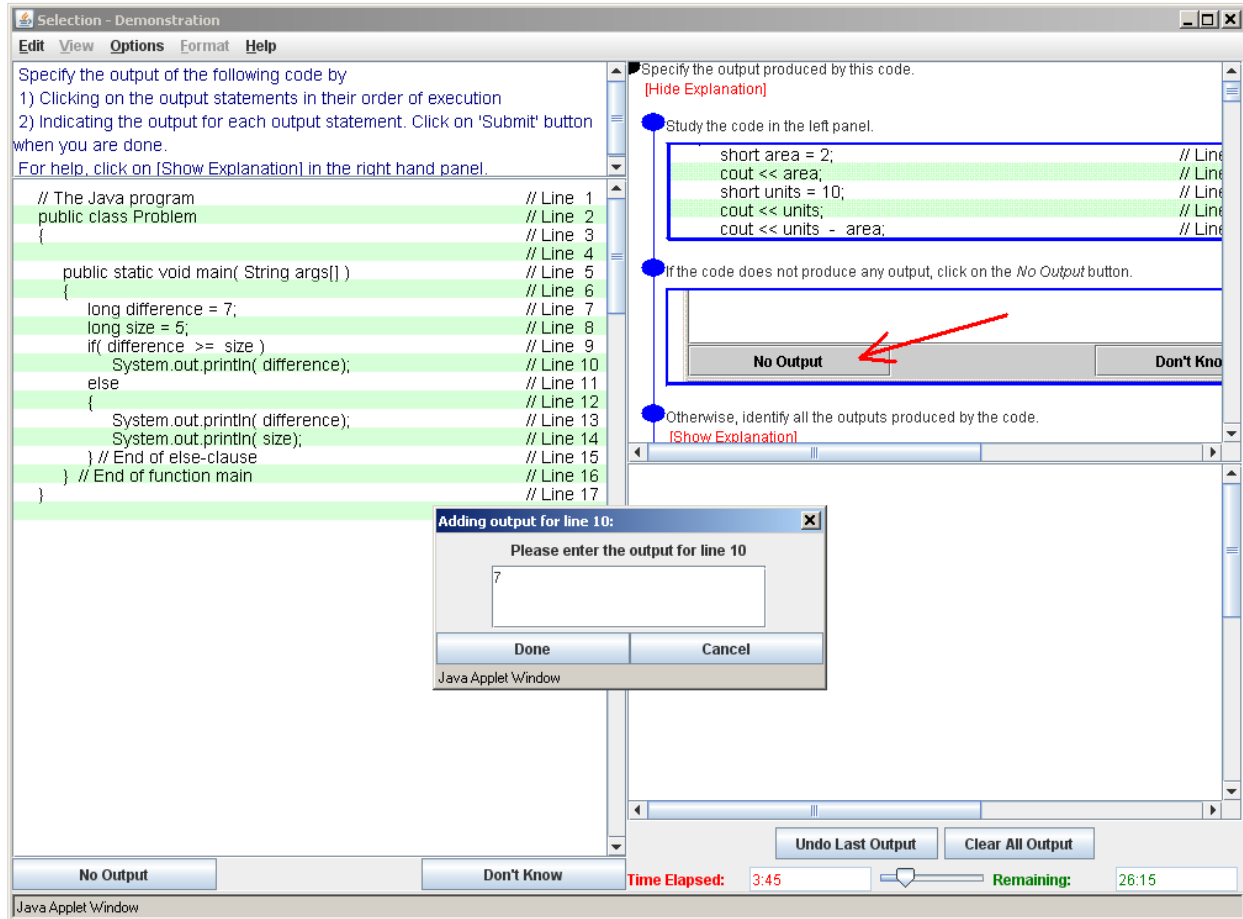
3.5.1.1 *Problets* and their pedagogical models

*Problets*²⁰ are problem-solving tutors on introductory programming concepts in *C/C++/C#/Java*. They present programming problems, grade the student's answer, and provide corrective feedback. *Problets* sequence problems adaptively, and generate feedback messages that include a step-by-step explanation of the correct solution (Kumar 2006). Students can use *Problets* for knowledge assessment and self-assessment, as well as for improving their problem-solving

²⁰ <http://www.problets.org>

skills. Figure 36 presents student interface of a *Probleto* on *if/else* statements in Java. The bottom-left panel contains a simple *Java* program. Students need to evaluate the program and answer a question presented 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 with green and red colors correspondingly. The detailed help on how to use the system, submit answers and receive feedback can be accessed in the right-top panel of the *Probleto* interface.

*Probleto*s domain models are built as concept maps enhanced with pedagogical elements called learning objectives. Each learning objective is associated with the proficiency level calculated based on the student's answers, thus forming a kind of an overlay student model. This student model provides the basis for adaptive decisions made by the tutor, through associating a proficiency model with each learning objective. The system propagates the proficiency values to the top levels of the concept hierarchy. At any point in the tutoring session, a student can observe the current state of her/his user model. Figure 37 demonstrates an example of the user model snapshot for the *if/else* Statements in *Java*.



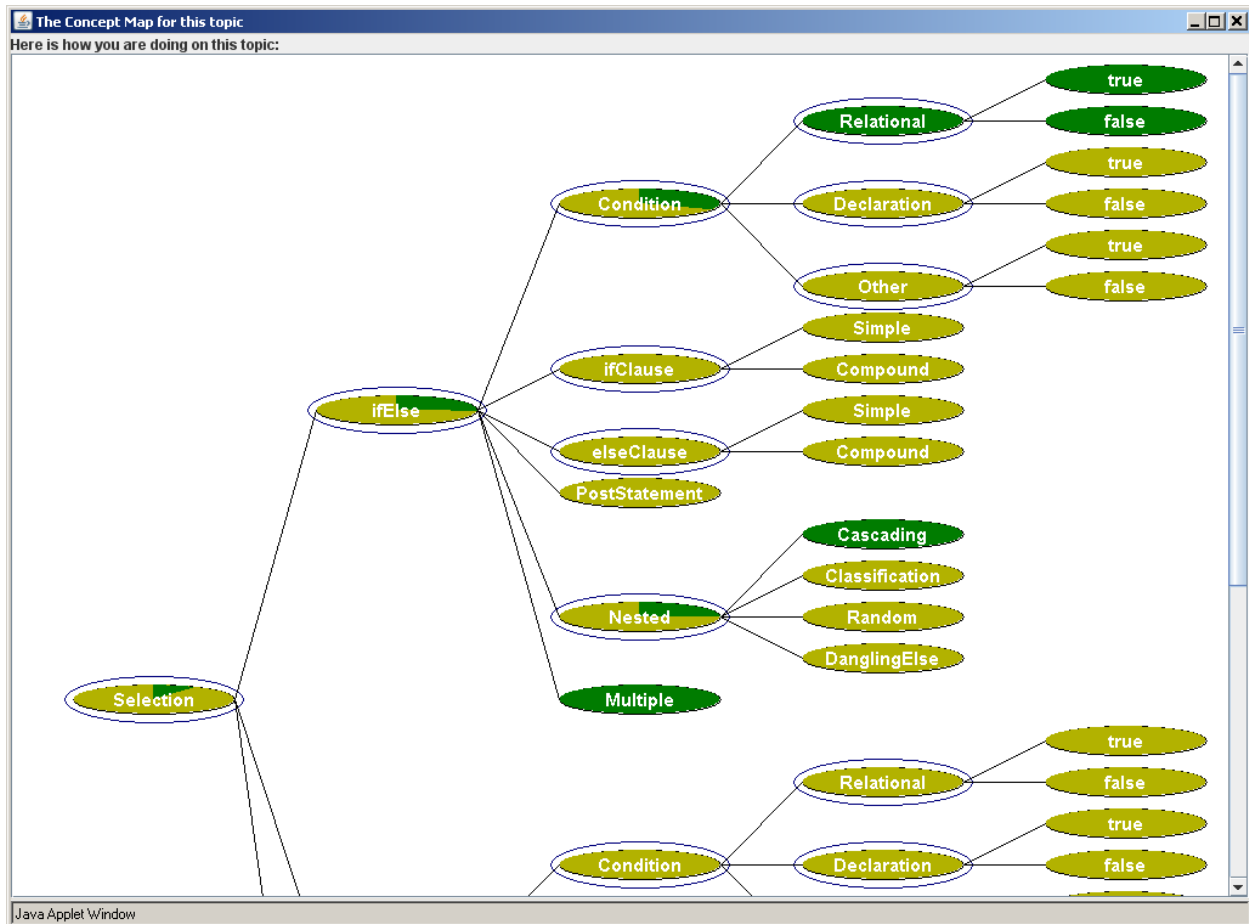


Figure 37: A part of the domain hierarchy on *if/else* statements in Java; learning objectives are associated with each concept in the hierarchy

3.5.1.2 QuizJET and the Java ontology

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 implemented as a simple Java program. Students need to evaluate the program code and answer a follow-up question. After the answer is submitted, *QuizJET* provides brief feedback specifying the correctness of the answer and the right answer if a mistake has been made. Figure 38 demonstrates the interface of QuizJET. The *Java* programs constituting *QuizJET* questions can consist of one or several classes. To switch 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.

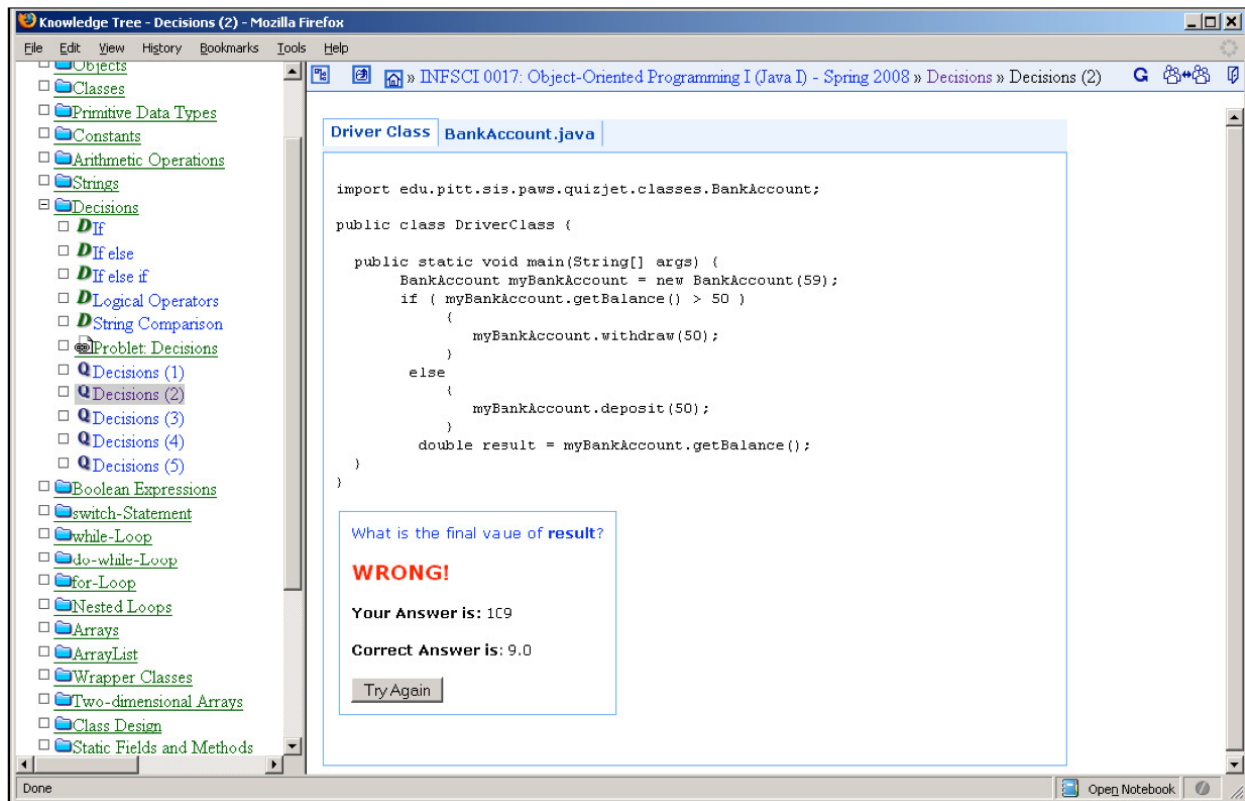


Figure 38: An example of *QuizJET* question on *Decisions* in Java accessed through the *Knowledge Tree* Learning Portal

Every *QuizJET* question is indexed by a number of concepts from the Java ontology. A concept in a question can play one of the two roles: prerequisite (if it is introduced earlier in the course), or outcome (if the concept is first introduced by this question). Figure 39 presents an extract from the Java ontology.

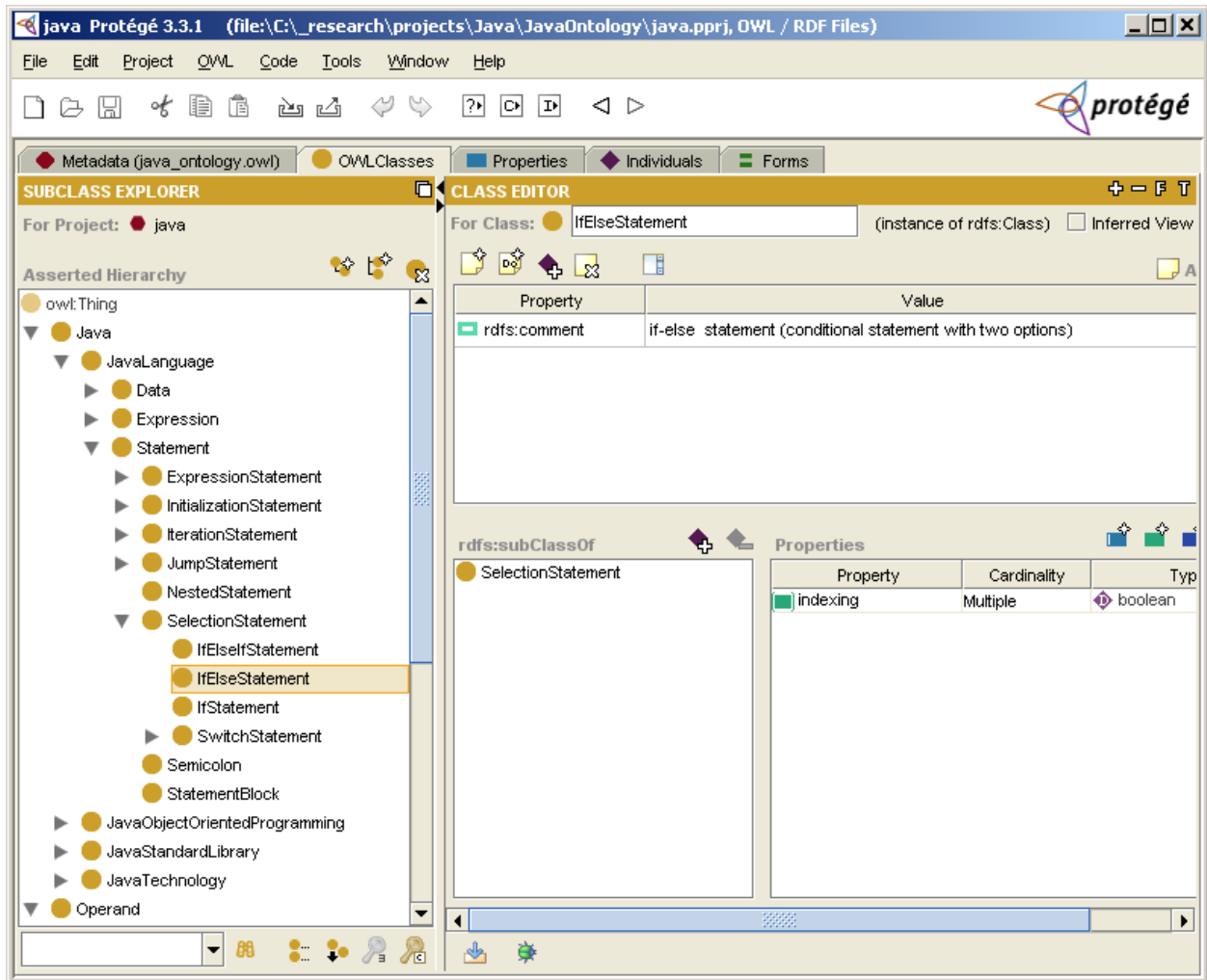


Figure 39: An extract from the Java ontology

3.5.1.3 Integration of *Problets*' pedagogical models and the Java ontology

Both *Problets* and *QuizJET*'s questions rely on modular content models that provide detailed representation of underlying domain knowledge. However, unlike *QuizJET* questions that are indexed with the concepts from the same ontology, each *Problet* relies on a separate model of learning objectives. These models cover six large topics of Java programming language: (1) *Arithmetic Expressions*, (2) *Relational Expressions*, (3) *Logical Expressions*, (4) *if/if-else*

Statements, (5) *while Loops*, and (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 considerably higher. The total number of concepts in the *Java* ontology related to these six topics is about 50. The cumulative number of nodes in the *Problets*' models is more than 250. The most important problem here is the difference in the modeling approaches (or different focus of modeling) used in *Java* ontology and *Problets*' domain models. Every learning objective models the 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. In order to properly map the context of learning objectives, they had to be connected to several concepts from the *Java* ontology. To prevent aggressive evidence propagation to the concepts modeling context of learning objectives, they were assigned with weights (from 0 to 1) that define how much knowledge of a particular concept defines the proficiency of the learning objective.

An example of mapping a learning objective to concepts is given by Figure 40. This terminal-level 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, one needs 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 contribute slightly to the knowledge of concepts *RelationalOperator* and *True*. Once this mapping is done for all *Problets*' learning objectives, any evidence of students' progress reported by any *Problets* in terms of learning objectives can be

interpreted in terms of the ontology-based student model used by *QuizJET*. Thus, *QuizJET* system receives access to the content from open corpus.

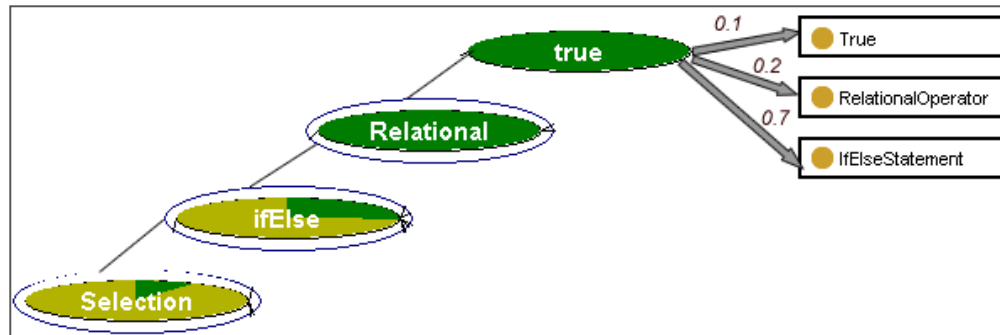


Figure 40. Mapping learning objectives to concepts

3.5.1.4 Summary

The described approach has been implemented and evaluated in the framework of an undergraduate course Java course in the School of Information Sciences (university of Pittsburgh). Both QuizJET questions and Problets were available to students during the semester as supplementary learning tools. Students' activity with the tools was stored on the user modeling server CUMULATE and was used by the learning portal KnowledgeTree for providing adaptive navigation support in form of progress-based annotation of links to Problets and QuizJET questions. Unfortunately, the amount of data collected was not enough to conduct statistical test an empirically validate the validity of collected student models and the effectiveness of generated adaptation.

3.5.2 Manual mapping of a domain ontology and a constraint-based model

3.5.2.1 *SQL-Tutor* and constraint-based modeling

SQL-Tutor is an intelligent tutoring system that helps university-level students learn SQL (Mitrovic, Martin et al. 2007). The system contains definitions of several databases, and a set of problems and the ideal solutions to them. In order to check the student's solution, *SQL-Tutor* compares it to the correct solution, using domain knowledge represented in the form of constraints. 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 system analyzes the solution, identifies mistakes (if there are any), and provides adaptive feedback. 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 student model. Figure 41 demonstrates the student interface of *SQL-Tutor*.

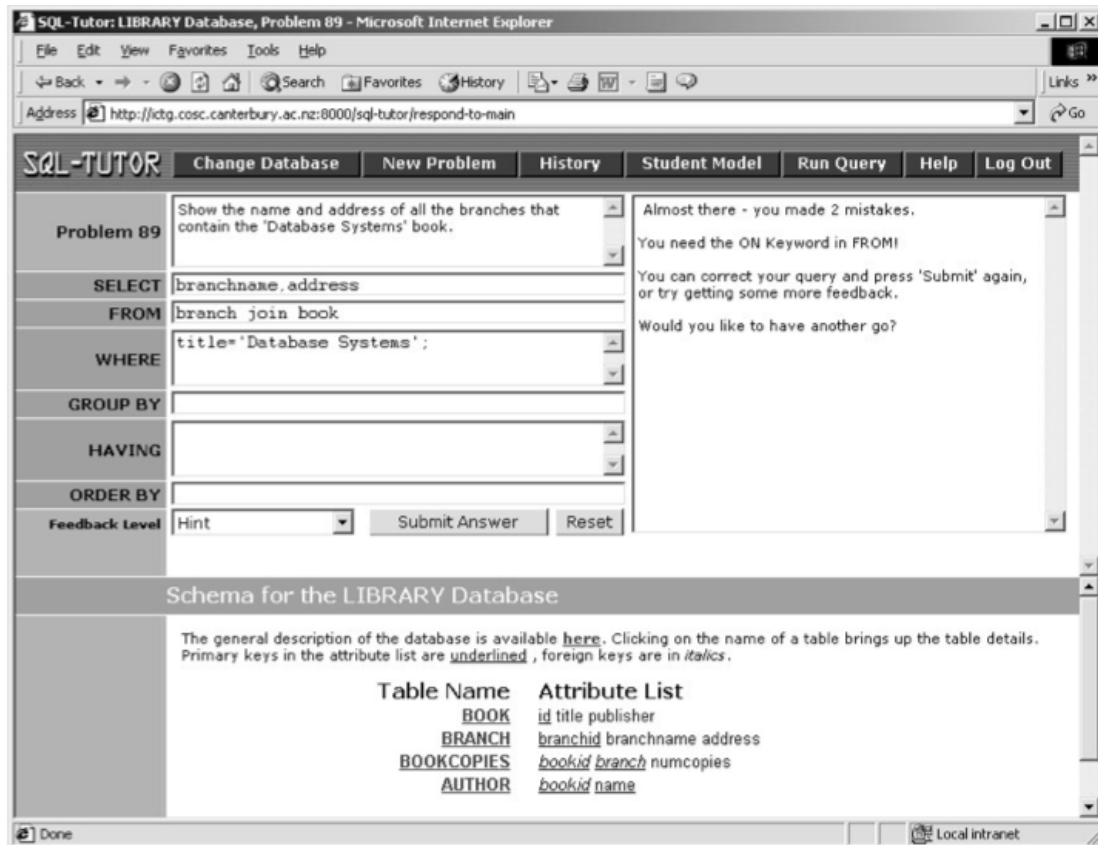


Figure 41: The interface of SQL-Tutor

SQL-Tutor represents domain knowledge as constraints. Constraints are domain principles that must be satisfied in any correct solution. Each constraint contains two conditions: the relevance condition and the satisfaction condition. A constraint is relevant if the features within the student's solution match the same features described in the relevance condition. The satisfaction condition describes what must be true in order for the solution to be correct. If the student solution violates the satisfaction condition of any relevant constraint, the solution is incorrect. Feedback messages attached to each constraint allow the system to present detailed and specific feedback on violated constraints. The constraint set in *SQL-Tutor* contains about 700 constraints, which check for syntactic and semantic correctness of the solution. Figure 42 illustrates two constraints.

```

(16 "You have to specify the grouping in the GROUP BY clause before you can
specify how to restrict grouping in the HAVING clause!"
(not (null (having ss)))
(not (null (slot-value ss 'group-by)))
"GROUP BY")

(635 "Check the condition involving the nested SELECT!"
(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")
"WHRRR")

```

Figure 42: Two example constraints

3.5.2.2 SQL-Guide and the SQL ontology

SQL-Guide is an AES helping students to practice SQL skills. 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 (see Figure 43). The system evaluates student's answer and provides simple feedback. To assist students in choosing the appropriate problem to practice, *SQL-Guide* annotates them with adaptive icons reflecting the progress of the student with the learning material underlying this problem. The *CUMULATE* user modeling server keeps track of all answers the student has given to *SQL-Guide*'s problems and computes the long-term model of his/her knowledge for the related concepts. *SQL-Guide* requests the state of the model and dynamically annotates problems with the appropriate icons. A more complete description of the system can be found in (Sosnovsky, Brusilovsky et al. 2008).

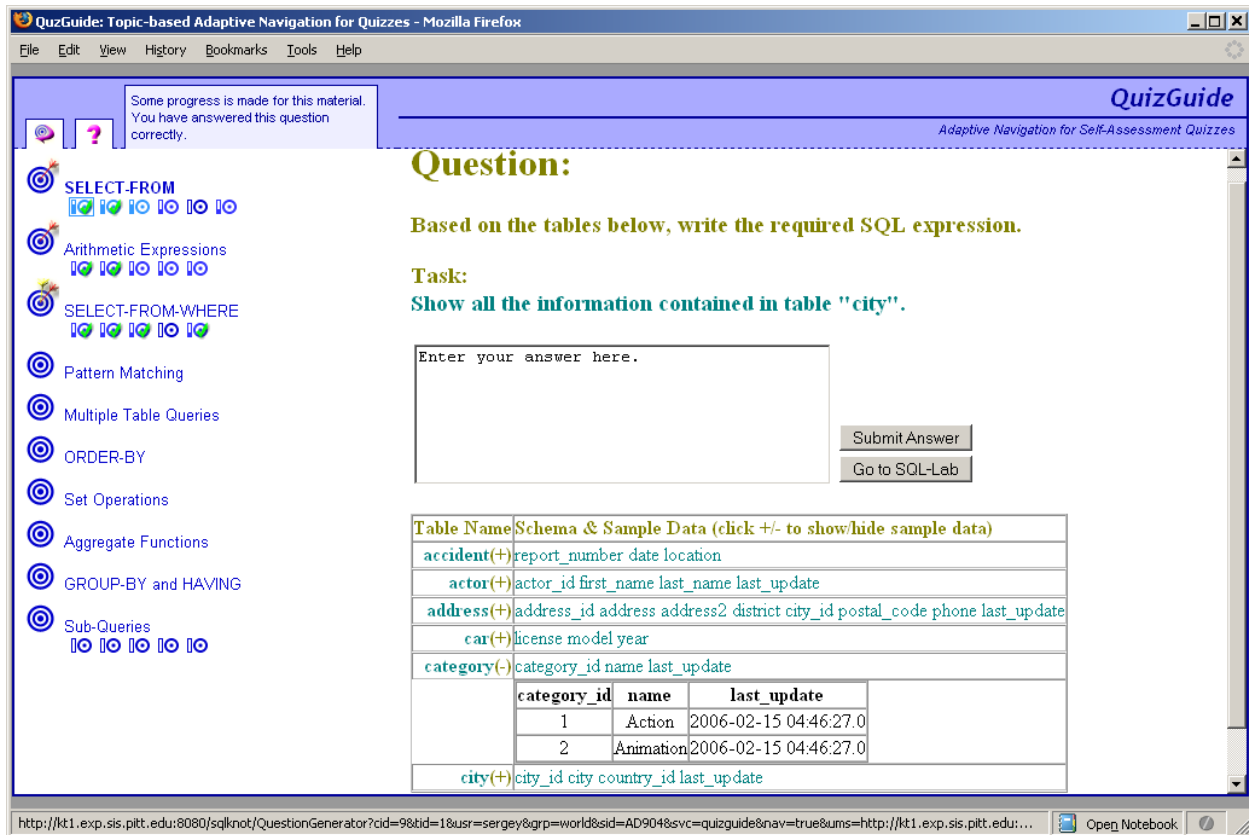


Figure 43: Interface of SQL-Guide

Every problem in *SQL-Guide* is indexed with several concepts from the SQL Ontology²¹, which was developed as a collaborative effort between the PAWS Lab of the University of Pittsburgh, and the ICT Group of the University of Canterbury. The main purpose of this ontology is to support the development of adaptive educational content for SQL and facilitate the integration of educational systems in this domain, while ensuring the objective modeling of SQL semantics. It is an OWL-Lite ontology, with more than 200 classes connected via three relations: standard *rfs:subClassOf* (hyponymy relation) and a transitive relation pair *sql:isUsedIn* – *sql:uses* (see Figure 44).

²¹ The ontology can be accessed at <http://www.sis.pitt.edu/~paws/ont/sql.owl>

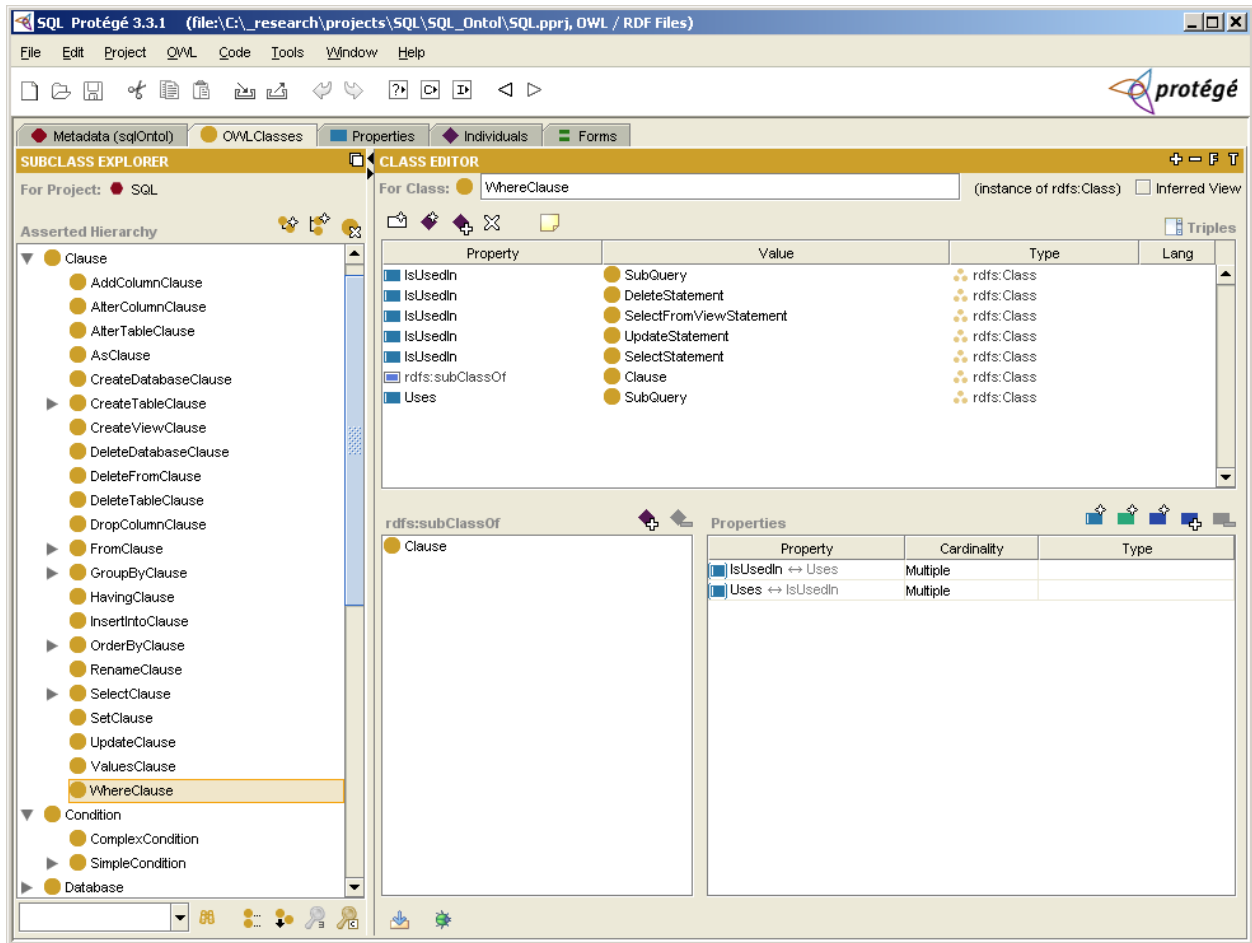


Figure 44: SQL Ontology

3.5.2.3 Semantic integration experiment

The fundamental differences in the domain models of the two systems make reliable automatic alignment of these models impractical. A well-established set of ontology mapping techniques cannot be applied to this task due to the unique nature of *SQL-Tutor*'s constraints. 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. The development of the algorithm even for partial resolution of the modeling discrepancies between ontologies and constraint-based models is not a trivial task.

To investigate the feasibility of integrating the two systems, a manual mapping experiment was organized. Six experts participated in the experiment: two instructors teaching relational database courses, two PhD students who served as teaching assistants in the database-related courses, and two PhD students regularly using SQL. Each expert was asked to find the most relevant concepts for 20 constraints, which were selected to cover both different parts of SQL and the variety of constraint types. Every expert had hierarchical layout of the ontology. For every provided constraint the task was to pick relevant concepts from the ontology and assign them with the weights (low/medium/large) designating the importance of the relation between the constraints and the concept. The time for this task was not limited. The experts also specified their levels of expertise in SQL, knowledge engineering and constraint-based modeling, summarized in Table 6.

Table 6: Levels of relevant expertise

Expertise in...	SQL	Knowledge Engineering	Constraints
Expert1	Medium	Low	Low
Expert2	High	Low	Low
Expert3	High	Low	Low
Expert4	Medium	High	Low
Expert5	Medium	High	Low
Expert6	High	High	High

Although manual mapping by experts is regarded as the golden standard comparing to the mapping acquired automatically, there is a number of problems with this approach. Two,

arguably, the most important of these problems are: errors and subjectivity. As a result, there is often a high level of disagreement between the experts. The data analysis confirmed this phenomenon. Overall, the experts used 61 ontology concepts for mapping. Although the average number of concepts a single expert used per constraint was only 3.1 (SD=1.1), the number of unique concepts used by all experts per constraint varied from 6 to 12. For ten constraints more than a half of the mappings were assigned by only one of the six experts. This means that for the half of the constraints no agreement has been reached on more than a half of the mappings. Moreover, for two constraints there were no mappings that were agreed upon by the majority of the experts (4 out of 6). To numerically express the agreement between the experts, the matching ratio for every pair was computed. The ratio is the percentage of mapping cases provided by the first expert, on which he/she has agreed with the second expert. For example, if the expert1 found 100 mappings, out of which 50 have been confirmed by the expert2, the rating of agreement of the first expert to the second is $50/100 = 0.5$. The average rating of agreement varied from only 40% to 66%.

More detailed analysis showed that the experts approached mapping in two different ways: some experts provided very short sets of mappings, others tended to over-specify the conceptual maps of constraints. The total number of mappings varied from 40 to 81. In order to align the manually created mappings, the taxonomic structure of the SQL ontology was employed along with three main heuristics to refine mappings:

- If an expert maps a constraint to all children of a single concept, such mappings are substituted by the mapping “constraint – parent concept”;

- If the minority of experts map a constraint to a child of the concept chosen for the same constraint by the majority of experts, the majority vote is taken: the mapping is set to the “constraint – parent concept”
- If the minority of experts map a constraint to a parent of the concept chosen by the majority for the same constraint, and there is no sibling concepts mapped to the same constraint, the majority vote is taken: the mapping is set to the “constraint – child concept”.

The ontology-enhanced mappings resulted an increased expert agreement. The pairwise t-test demonstrated a statistically significant difference between the original agreement ratings ($M = 0.52 \pm 0.046$) and the agreement ratings for ontology-aligned expertise ($M = 0.59 \pm 0.032$); $t(5) = 3.705$; $p = 0.014$.

3.5.3 Automatic translation between two domain ontologies

Unlike the two previous studies, the experiment presented in this subsection was aimed at automatic model mediation. Both models were based on domain ontologies; therefore it became possible to apply ontology mapping techniques for automatic discovery of corresponding elements across the models. Another difference from the previous experiments was that instead of two models in the same domain, in this study, two ontologies of relevant domains were integrated: Java and C programming (see Figure 45).

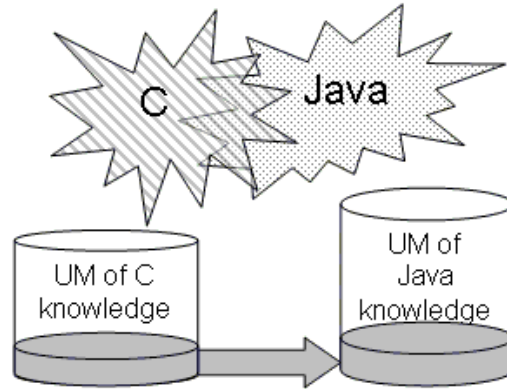


Figure 45. Relation between C and Java knowledge

3.5.3.1 Ontology mapping for student models translation

Two existing ontologies have been used in the study – one for Java and one for C. The Java ontology²² was developed for *AES Personal Reader for eLearning* (Henze, Dolog et al. 2004). The ontology is represented in RDFS; it defines 490 *rdfs:Class*'s connected to each other by the generic relation *rdfs:subClassOf*. The ontology is used to index the online Java Tutorial (Oracle 2011), which was the source for developing the ontology. The description of the tutorial structure and the index of its pages in terms of the ontology concepts is represented as another RDF document (Henze, Dolog et al. 2004).

The C programming ontology²³ is also represented in RDFS. It has been developed at the University of Pittsburgh. The ontology contains 575 *rdfs:Class*'s and defines 5 *rdf:Property*'s. Table 7 summarizes some basic characteristics of two ontologies. C Programming ontology provides much finer-grained representation of the conceptual structure of the domain. During the manual mapping, Java concept often mapped to several C concepts; therefore the manual mapping pull contains more C concepts than Java concepts.

²² http://personal-reader.de/rdf/java_ontology.rdf

²³ http://www.sis.pitt.edu/~paws/ont/c_programming.rdfs

Table 7: Basic metrics of the mapped ontologies

	Java Ontology	C Programming Ontology
Number of rdfs:Class's (concepts) defined	490	575
Number of concepts mapped	108 (22%)	257 (45%)
Number of rdf:Property's defined	0 (uses rdfs:subClassOf)	5 (isA, partOf, directPartOf, implement, utilize)
Number of RDF-triples	1013	1538

The focus of modeling for C Programming Ontology was also different. While the Java ontology was initially created to describe the content of the particular Java tutorial, the C programming ontology was deliberately developed as a general-purpose ontology, as an attempt to model the domain of C programming in the maximally precise and unbiased way. These differences in the modeling perspectives added additional difficulties for mapping. The content for indexing by the C Programming Ontology was taken from two online C tutorials: University of Leicester C Tutorial²⁴ and Rob Miles's Tutorial²⁵.

To perform ontology mapping a implemented the modified version of the *GLUE* algorithm has been implemented (Doan, Madhavan et al. 2002). This algorithm computes the matrix of similarities between all concepts of two ontologies. The main idea of *GLUE* is based on the cross-classification of concepts' instances (tutorial pages) and calculation of joint probabilities for each pair of concepts from two ontologies. The quality of this algorithm is determined by the precision of classifiers, which highly depends on the quality and the size of

²⁴ <http://www.le.ac.uk/cc/tutorials/c/>

²⁵ http://www.eumus.edu.uy/eme/c/c-introduction_miles/contents.html

the training data sets. To reduce this dependency, an extra mapping step based on the concept names was added. On this step concept names are parsed into separate words and compared based on their lexical similarity calculated using Levenshtein's edit distance (Levenshtein 1966) and the metric proposed in (Maedche and Staab 2002). The output of the name-based mapping has been used as an input for the *GLUE* component. Another characteristic of the input data, which has a negative impact on the mapping precision, is the large percentage of instances shared by several concepts (i.e. tutorial pages are indexed with a number of concepts). This is a typical situation for adaptive content authoring, but not for the ontological engineering. To deal with it during the classification instances were weighted according to the ratio of influence that the concept has on this tutorial page. Hence a page indexed with a set of n concepts including the concept A contributes to the training of the classifier for this concept n times less than a page whose index contains only the concept A .

The original *GLUE* algorithm was developed to find for all concepts of the first ontology best possible "1-to-1" mappings from the second ontology. This requirement was relaxed since the main goal was not the ontology mapping itself, but model translation. In some cases, the mapping algorithm was unable to distinguish between closely related concepts because they had similar names and definitions and shared many instances. This is less important in the context of knowledge modeling, because students tend to have similar knowledge for related concepts, e.g.: *WhileLoop* and *DoWhileLoop*. Hence, instead of finding best "1-to-1" mapping the modified algorithm finds several best candidates for every concept, where mapping is possible. These candidates are weighted according to the similarity values found during the mapping process. When SMs are translated these weights determine the percentage of contribution by candidate concepts from the first ontology to the resulting knowledge level of the target concept from the

second ontology. This feature helps dealing with “1-to-many” mappings, especially when the candidate concepts are from different parts of ontology. For example, good candidates for a concept *return* from the Java ontology will be concepts *FunctionReturnValue* and *StatementReturn*, which belong to different branches of the C Programming ontology.

3.5.3.2 Evaluation of translation quality

The experiment was conducted in the introductory course on object-oriented programming taught to undergraduate students of the School of Information Sciences, University of Pittsburgh. In the beginning of the course, a questionnaire assessing their initial experience in C and Java programming was collected. Out of 36 students seven demonstrated prior knowledge of Java and therefore were not eligible for the main experiment. Eight students knew neither C nor Java; they formed the control group for evaluation of C-model quality. Finally 21 students had C programming experience but had never used Java before. They became our main experimental group. During the analysis of outliers two cases were filtered. The final numbers of students in the groups are 7, 7 and 20 correspondingly.

The 10-question pre-quiz has been used for initialization of SM for C-programming. As a result the prior knowledge of 37 concepts from the C ontology were estimated. Java knowledge were evaluated using weekly quizzes. Each of these quizzes included 2-3 extra-credit questions checking student knowledge of several Java concepts that have some analogy in C language. At the time of the quiz, these concepts were not yet mastered in the class so the recorded knowledge of these concepts was a result of transfer from C, not the newly acquired knowledge. This provided the overlay models of knowledge of 21 concepts from the Java ontology. The problems (both in C pre-quiz and in Java extra-credit questions) assessed students’ knowledge on the level of application – they had to analyze code fragments and answer related questions.

To create the “*ground truth*” for our automatic ontology mapping experiment, these sets were manually mapped; 17 mapping cases were identified from four large categories: *Functions/Methods*, *Operators*, *Simple Data Types* and *Control Structures*. All 17 cases were “1-to-1” mappings. They were the best mappings one could achieve between the target parts of the two ontologies. The automatic ontology mapping reduced this subset to nine concepts. It became the target set of concepts. For these nine concepts it was possible to compare the results of the user model translation, based on the automatic ontology mapping, with the translation being based on the manual (“*perfect*”) mapping.

On the next step, the quality of the user model translation based on automatic ontology mapping was compared with the “*ground truth*,” as provided by the results of manual user model translation. User models were represented as vectors, where every concept is a coordinate, in a way the student knowledge of nine related C and Java concepts are characterized by two 9-dimensional vectors (as an example, Figure 46 shows the user model of knowledge of the C concepts *StatementFor* and *StatementIfElse* and the user models of knowledge of the corresponding Java concepts *for* and *else*, represented as 2-dimensional vectors). Since the manual mapping was the best possible one-to-one mapping, one can say that the dimensions of the C and Java vectors are collinear. Hence, it is possible to superimpose these coordinate spaces and calculate the Euclidian distance between two vectors, which reflects the divergence between the initial C and Java knowledge of the student, and, consequently, the divergence between the Manually-translated Java Model and the Original Java Model.

Based on the manual mapping, as well as the results of the C pre-quiz and Java extra-credit questions, the divergence distances for all students from the target group of students (those, who new C and did not know Java ion the beginning of the experiment) were calculated.

These distances characterize both the shortcomings of the student modeling, due to the simplification of modeling assumptions and defects in elicitation process, as well as possible differences in the students' understanding of the relevant concepts of C and Java, unexpected mistakes (or guesses) on pre-quizzes, etc. Overall, these distances characterize the intrinsic imperfection of the obtained dataset. It is not possible to compute a set of smaller distances, i.e., to perform a better translation of the user models based on the given data.

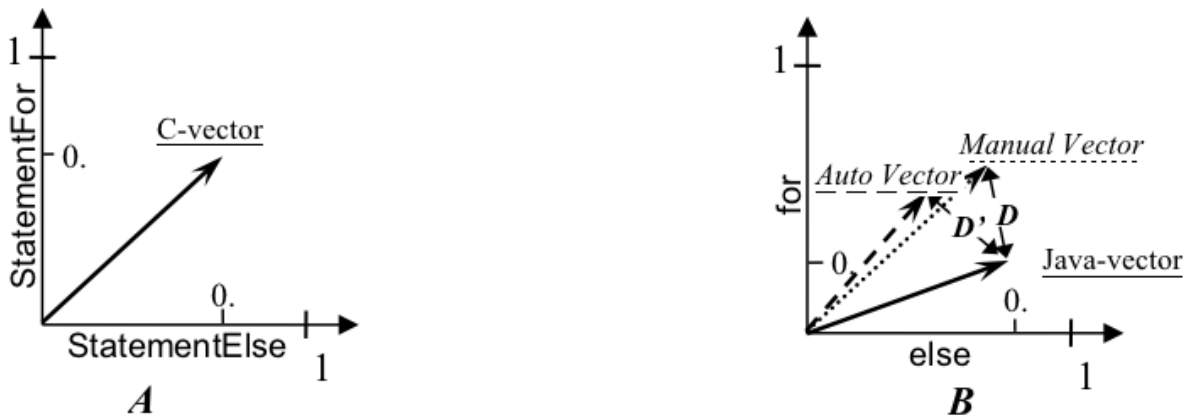


Figure 46. The user models of knowledge for two concepts represented as 2-dimensional vectors²⁶

Automatic mapping provided another set of distances. These distances characterize the differences between the original Java model vectors obtained from the Java pre-quiz and the Java model vectors computed by automatic translation of the user models of C programming into the user models of Java programming. Every coordinate (every concept's level of knowledge) of

²⁶ **A:** The C-model (knowledge level of *StatementIfElse* is 0.7; knowledge level of *StatementFor* is 0.6). **B** The Java-model (knowledge level of *else* is 0.8; the knowledge level of *for* is 0.2) with the two other vectors superimposed: (i) Manually-translated Java vector (Corresponding concepts of the C and Java models are mapped to each other as 1-to-1; therefore the dimensions of the two coordinate spaces are collinear); (ii) Automatically-translated Java vector. Euclidian distance D shows the divergence between the Manually-translated Java Model and the Original Java Model, Euclidian distance D' shows the divergence between the Automatically-translated Java Model and the Original Java Model.

later vectors is calculated based on the results of the automatic ontology mapping and the knowledge of C concepts being mapped. To verify that the distances obtained manually are close enough to the estimates based on the automatic mapping, the correlation analysis was applied. The results (*Kendall's Tau b* = 0.844; $p \ll 0.05$) demonstrate an exceptionally strong relationship between the two variables.

This effectively shows that the user models of Java knowledge obtained with the help of automatic translation from the user models of C knowledge are very close to those computed manually, i.e., automatic mapping provides close to “perfect” translation of knowledge levels from the C domain to Java domain.

3.6 SUMMARY: BUILDING UP THE MAIN DISSERTATION APPROACH

This chapter has given an overview of the author’s contributions to the topics of semantic modeling of open-corpus content and open-corpus personalization. It plays a role of a bridge between the related work in the field (described in Section 2.2) and the main dissertation approach (Chapter 4.0). Figure 47 tries to represent the entire spectrum of open-corpus content modeling technologies and their relations to this dissertation. All technologies are classified into the three main categories described above. They define three principles ways of how a system can help a human author model new content: the model can be extracted automatically, its manual creation can be significantly simplified, or it can be integrated with another model. Some of the technologies in the figure have not been investigated by the author; their description can be found in Section 2.2 (e.g. *Semantic Annotation*, *Term-based Modeling* or *Semantic Tagging*). Another set of technologies was evaluated by the author during his PhD study and has been

described in the subsections 3.3-3.5 (e.g. *Model Extraction from Formal Content*, or *Manual Mapping of Domain Models*). The figure also highlights the technologies that served as motivation for the approach presented in this thesis (*Extraction of Content Structure*, *Coarse-grained Modeling*, and *Automatic Model Translation based on Ontology Mapping*). Next three subsections summarize the pros and cons of these technologies from the point of open-corpus personalization for e-Learning.

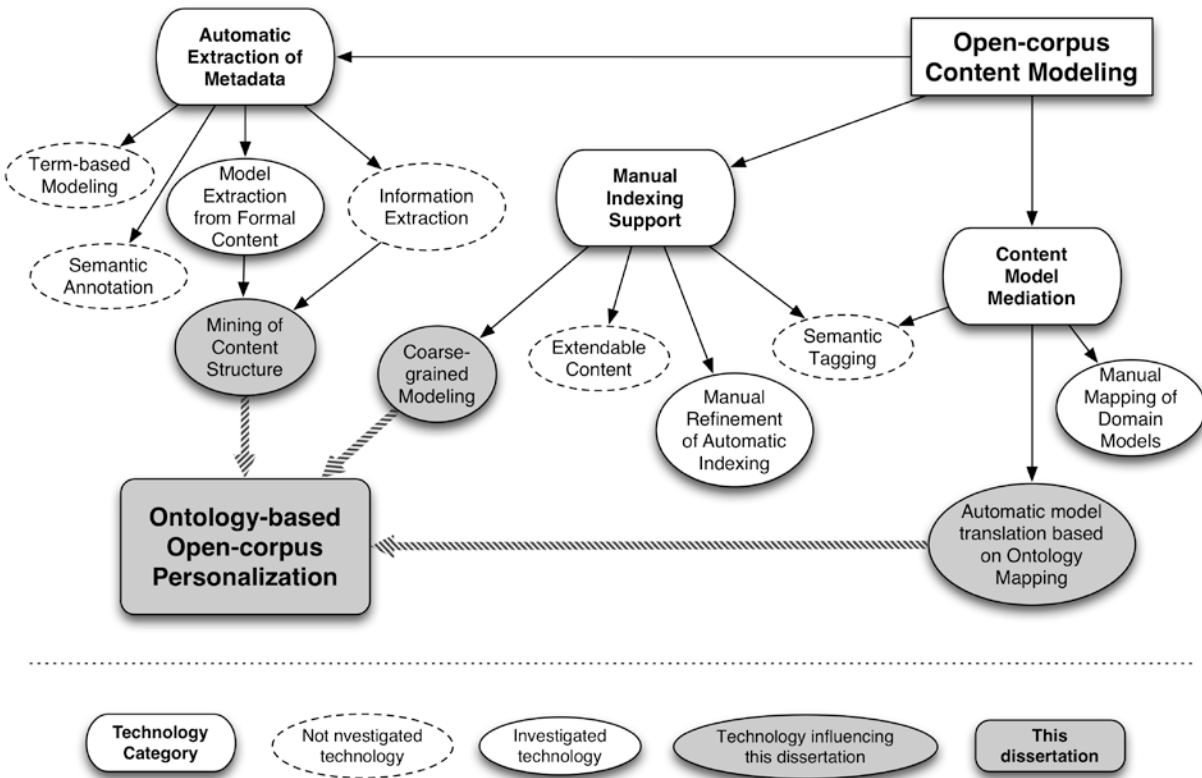


Figure 47. Classification of open-corpus content modeling technologies, including investigated and not-investigated technologies and the relation between them and this dissertation.

3.6.1 Automatic extraction of content models from structured and semi-structured content

As with any knowledge engineering task, involving human expert in manual creation of content models or refinement of automatically extracted ones should result in better-quality modeling. However, in the case of *open-corpus* content modeling, any solution but fully automatic is not scalable enough, as the ultimate scale of the task is entire WWW, or at least a certain kind of Web-content.

Programming code is almost an ideal type of content for automatic model extraction. It follows well-defined, formal rules; the tools for parsing it according to these rules have been created and standardized; and there is a lot of programming code available online. These considerations motivated the research project described in Section 3.3. There are limitations, of course: not all instructionally-important concepts can be automatically recognized, not all learning resources include programming code, and only a few domains rely on parsable content. However, the main idea behind this approach – to automatically extract semantic model from the content based on its structural regularities – can be applied in a broader context.

If one looks at the spectrum of technologies that allow automatic modeling of open-corpus content for e-Learning, there is a certain tradeoff between how semantically-deep the resulting model is and how broad is its applicability from the point of subject domain and supported personalization approaches (Figure 48).

Classic term-based modeling allows for broad range of domains and implementation tasks, but the resulting models are rather statistical than semantic. Using a lexical ontology, such as *WordNet*, can help the obtained model to deal with such phenomena as homonymy (multiple meanings of a word) and synonymy (multiple words expressing the same meaning), and can

even enhance it with structural relations between terms. But this solution would only work in very general domains where *WordNet* is well-developed, it will fail to cover specific subjects, that are important for e-Learning applications. Semantic tagging based on mapping folksonomies and ontologies would allow for domain-independent creation of semantically-enriched models, requires student-provided tags that limits usage of this technology to a very narrow set of instructional scenarios. Semantic annotation and information extraction are often used interchangeably. They allow for extraction of very rich semantic models in rare circumstances.

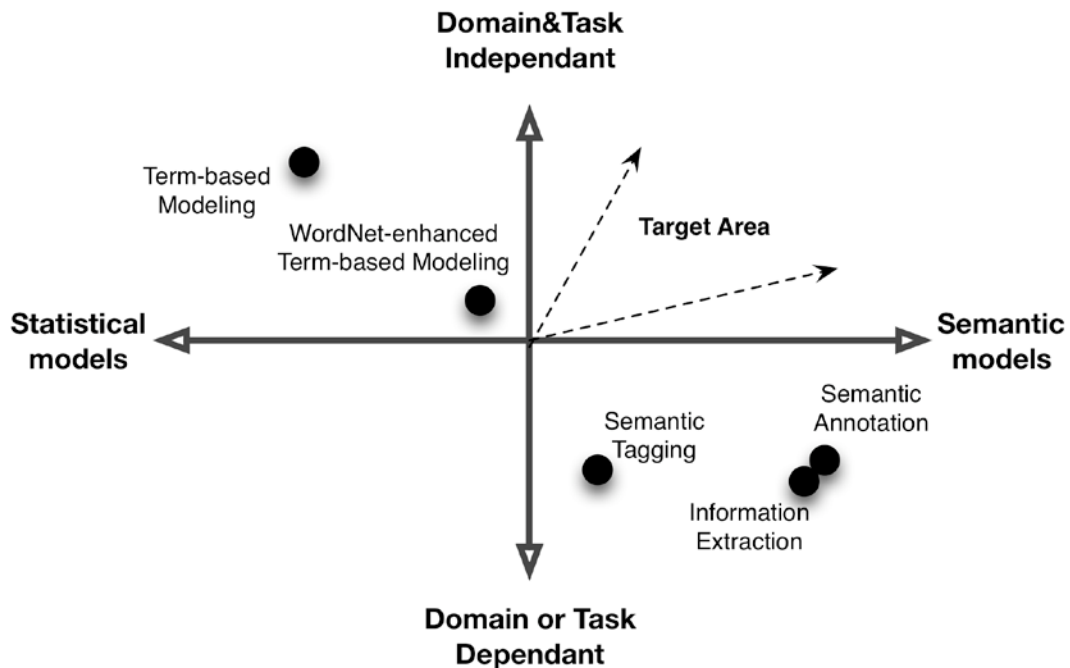


Figure 48. Technologies for automatic open-corpus content modeling

The focus of this dissertation is the first quadrant of Figure 48. A modeling mechanism placed there enables extraction of semantic models based on domain ontologies while not restricting created models only to a handful of domains and/or learning scenarios. Some

sacrifices will have to be made, though. By refusing to sacrifice automation and domain independence, this work chooses to somewhat relax the requirements to the properties of the extract model.

3.6.2 Topic as a modeling unit

Similar to finer-grained concepts, the main purpose of the coarse-grained topics described in Section 3.4 is to represent elements of knowledge in the domain of discourse, to serve as a basis for identification of the student's understanding of corresponding learning material, and to support proper adaptation. However, the topic-based approach to knowledge modeling is different in several ways:

- Topics provide means for learning material aggregation, instead of traditional indexing. As a result, the relationships between topics and pieces of learning content are “1-to-many” (e.g. one topic corresponds to many quizzes/questions) and topic-based “indexes” do not exceed manually manageable numbers.
- Topics provide a natural approach for a classroom teacher (or a textbook author) to organize the course into logically separate units and assign appropriate pieces of content to them. Authoring of a topic-based domain model can be easily done by a classroom teacher while s/he is developing the course structure.
- In adaptive systems, topics can play two roles: as knowledge components for the student modeling and content-based adaptation, and as interface elements for content structuring, labeling and navigation.
- Topics are coarse-grained; therefore, when relying on topic-based knowledge modeling, the model precision is sacrificed. Yet, the example of QuizGuide demonstrates that even

such a minimalist approach to domain and student modeling can result in efficient adaptive interfaces that have a positive impact on several aspects of learning process.

- Topics are subjective. If comparing topic structures of the same course developed by different teachers, one can expect them to be quite different. The presence or absence of a single topic, the naming labels, the size of particular topics, the inter-topics relations, and the scope of the entire set can vary from one structure to another based on the personal decision made by a teacher.

In summary, topics are unique knowledge components, which in the framework of adaptive e-Learning, have a number of positive and negative features compared to smaller concepts. However, the main advantage of a topic is that, while relaxing the requirements to content modeling, it is capable to ensure sufficient adaptation quality. The stream of research described in Section 3.4 has demonstrated how topics can help to manually create effective adaptive interfaces. Yet another implication is the possibility to automatically extract topic-based models and build adaptation around them. Today's Web contains a lot of information resources that have been created with clear topic-based structures in mind. At the same time, topics are not suitable for student modeling. A solution should be found that helps to mediate between the extracted topic-based content model and a traditional concept-based student model.

3.6.3 Ontology mapping as a basis for automatic model mediation

The experiments described in Section 0 investigate cross-model knowledge mediation from several perspectives: how different the mediated models can be, how reliable manual model translation is, can ontology mapping be used for automatic translation of knowledge models.

For this dissertation, the results of the last experiment (Section 3.5.3) are especially important. It has shown that automatic ontology mapping can be employed for translating student knowledge from one model to another. For the problem of open-corpus personalization, it means the possibility of automatic translation between an external content model and the internal domain/student model. Based on this assumption, and taking into account the results of other studies summarized in the subsections 3.5.1 and 3.5.2, the main approach of this dissertation has been proposed. Next chapter describes it in details.

4.0 AUTOMATIC MODELING AND ADAPTATION OF OPEN-CORPUS CONTENT COLLECTIONS WITH THE HELP OF ONTOLOGY MAPPING

4.1 INTRODUCTION

This chapter provides a detailed description of the main proposed solution to open-corpus personalization and content modeling in the context of e-Learning. To ensure broad applicability in e-Learning, the proposed approach has been designed to follow several important principles:

- minimal to no involvement of human author;
- reliance on semantic modeling of domains, content and students;
- independence of the domain-driven techniques and heuristics;
- support of a wide range of adaptation technologies;
- support of a wide range of open-corpus resource types.

The approach requires the development of a high-quality central ontology of a domain and relies on the assumption that there exist open-corpus collections of semi-structured material suitable for learning. The approach roots in the previous work done by the author on automatic content indexing, topic-based personalization and ontology-based student modeling. It uses ontology mapping as a tool for mediation between the fine-grained domain ontology used for representing student knowledge and coarse-grained content model extracted from online content collections.

The rest of the chapter is organized as follows. It starts with the discussion of the main assumption leading to the approach, i.e. the abundance of well-structured content on WWW available for automatic modeling in terms of coarse-grained topics (Section 4.2). It also outlines a few previous attempts to leverage the structure of Web-resources for creating knowledge models. Section 4.3 of the chapter provides a general overview of the proposed knowledge workflow and explains in greater details the main steps of the approach from topic extraction to topic adaptation. Section 4.4 discusses several positive features of the proposed solution. Section 4.5 concludes the chapter.

4.2 STRUCTURED CONTENT ON THE WORLD WIDE WEB

Information on the Web is not without structure. Authors of many online information resources create them as a reflection of their own internal organization of related knowledge. They encode this organization by formatting the text with lists and headings, breaking documents into sections and pages, linking pages together, creating tables of contents, etc. Many of such well-formatted online resources and collections of resources can be successfully used as sources of instructional material. Tutorials and electronic textbooks, dictionaries and encyclopedias, manuals and digital libraries – they all contain high-quality, domain-specific information written by domain experts with the purpose of explaining or documenting important knowledge. When formatting such resources their authors try to organize them according to the structure of the domain. If extracted and represented formally, such a structure becomes a model of the resource and the domain as seen by the author. As a domain model it will be subjective, coarse-grained, and incomplete. However, for this target resource it is an automatically extracted semantic model.

Several existing projects tried to approach a similar problem in a similar manner. However, they either focused on a single content collection formatted in a very particular way, or strongly relied on domain-specific heuristics and tools.

The project *MECUREO* mentioned in Section 2.4.2 is a good example of the first kind (Apted and Kay 2004). It was developed for automatic construction of the Computer Science ontology from the Free On-Line Dictionary of Computing (*FOLDOC*). *FOLDOC* pages follow strict and consistent rules of page formatting and linking between them. Every page has been created to formally represent a single notion of computer science – a single concept. The resulting model learnt by *MECUREO* is a lightweight ontology where concepts are connected by four types of relations: parent, child, synonym, and antonym. The system and the ontology can be used to provide personal access to the content of *FOLDOC*, but not other online collections.

An example of a domain-dependent technique for automatic knowledge extraction can be taken from Section 3.3 of this dissertation. It describes the author's work on automatic indexing of code-based exercises in the domain of C programming. Most of the programming code fragments online are properly formatted with one of the commonly accepted HTML-tags: `<code>` `<pre>` or `<tt>`. As a result, the programming code can be extracted and inputted to a parsing component that will automatically disassemble it into single concepts. However, every programming language will require a dedicated parser, while non-programming domains and resources without embedded programming code will not benefit of this solution at all.

The approach presented in this chapter does not rely on domain-specific techniques and it is rather forgiving with regards to content structuring and formatting. If the content is structured in such a way that every page represents a single cohesive notion in the domain, it will make the resulting model better, but it is not necessary. If the content formatting allows applying a set of

heuristics for mining more detailed knowledge elements from within pages, it will make the model better but it is not necessary. If the link structure of the collection is unified and well-developed, if extra resources like indices or thesauri are available and, thus, exact semantics of relations could be established, this would make the model better, but it is not necessary. The only assumption is that there is some structuring done by the author of the collection, and that the author knew what he or she was doing. In other words, the minimum acceptable level of content organization is the presence of page (section) headings and some relations between these headings, which are identifiable from the page-to-page (section-to-section) linkage and/or through a hierarchy of headings (e.g. based on the table of contents or the HTML tags, such as <H1>, <H2>, etc.). Such low level preconditions enable the applicability of the proposed solution in many domains and for a very wide range of content collection types.

4.3 PROPOSED CONTENT MODELING AND PERSONALIZATION WORKFLOW

This section describes the complete workflow of the proposed solution to open-corpus content modeling and personalization. To follow the scenarios described in Section 1.1.3, the procedure starts with a teacher discovering a good collection of online instructional material that s/he needs to connect to an existing adaptive system or a course. A teacher supplies the URL of the collection's index page and the system should do the rest. The transformation consists of the following three steps:

1. Topic-based modeling of open-corpus content in terms of its structure;
2. Mapping of the extracted model into the central ontology;

3. Adapting the open-corpus material based on the knowledge mediation between the central ontology-based student model and the topic-based content model of the open-corpus collection.

Steps 1 and 2 are content modeling steps and happen offline. Step 3 maintains real-time interactive and adaptive access to open-corpus learning material. The next three subsections provide the details of every step.

4.3.1 Step 1: topic-based modeling of open-corpus content in terms of its structure

This step combines several important subtasks of open-corpus personalization, namely:

- identification of individual learning objects in the open-corpus content,
- modeling of open-corpus content,
- connection of open-corpus learning objects to each other.

An author creating an instructional resource is trying to make it more readable and understandable by structuring it into chapters, sections, subsections, etc. Every subsection is intended to represent a single topic, it is provided with a title that conveys the essence of the topic and contains text that explains the meaning of it. Such topics do not necessarily correspond to a single concept in the domain. However, they do tend to combine the learning material in a logical manner, and, to some extent, represent a consistent piece of domain knowledge. Hence, though, their main purpose is structuring content, they inescapably structure knowledge, as well.

The extraction of this hidden semantic layer can be effective more or less. However, even the least ambitious solution, relying only on top-level structural relationships between the pages will help automatically extract a model of a collection. The positive side of this approach is that it can be applied in many domains and for a broad range of learning content types.

The weaker side is that the extracted model is not very detailed: a topic per page (or section). Yet, as the example of *QuizGuide* system has shown, such a content model can support effective adaptation of instructional material (see Section 3.4).

Figure 49 demonstrates an example of extracting such a model from one of the Java Tutorials²⁷. The page describes *if-else* statements in Java. It is formatted in such a way that helps users understand not only the content, but also the structure of this particular topic in the domain of Java Programming Language. “*The if-then and if-then-else Statements*” is the first-level heading, there are two second-level headings, as well: “*The if-then Statement*” and “*The if-then-else Statement*”. The table of contents on the left is supplied for the ease of navigation. The user can see from it that “*The if-then and if-then-else Statements*” is one of the subtopics of “*Control-flow statements*”, which is a subtopic of “*Language Basics*”. The breadcrumbs on the top-right show that “*Language Basics*”, in its turn, is a subtopic of “*Learning the Java Language*”. All this formatting and linking is encoded by the author(s) of the tutorial in order to facilitate both the understanding of the material and the navigation through the tutorial. As a result, based on trivial HTML parsing a topic-based structure of the tutorial is extracted (left part of Figure 49) and represented formally as an RDF model (Figure 50). The model contains the topic labels, the structural relationships between topics and the textual contents associated with them. The model also maintains the order of topics within the tutorial (this feature becomes important on the next step of the approach).

²⁷ <http://download.oracle.com/javase/tutorial/>

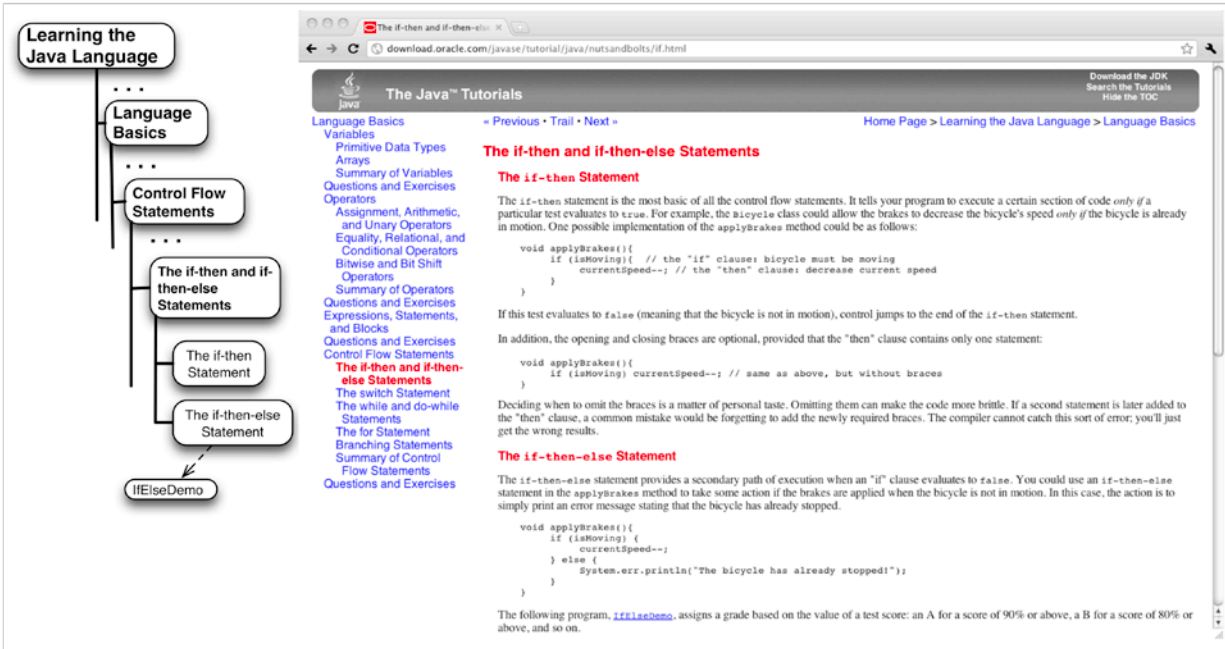


Figure 49: Example of a topic-based model extraction based on the resource formatting

```

...
<owl:Class rdf:about="#The_if-then_and_if-then-else_Statements">
  <rdfs:subClassOf rdf:resource="#Control-flow_statements"/>
</owl:Class>
<owl:Class rdf:about="#Learning_the_Java_Language"/>
<owl:Class rdf:about="#The_if-then-else_Statement">
  <rdfs:subClassOf rdf:resource="#The_if-then_and_if-then-
    else_Statements"/>
  <rdfs:comment>The if-then-else statement provides a secondary path
of execution when an ...

```

Figure 50: RDF-serialized extracted topic-based model

4.3.2 Step 2: mapping extracted model into the central domain ontology

On this step, modeling of the open-corpus content continues, but, most importantly, it takes care of connecting the open-corpus resources of the target collection to the rest of the learning content available in the system (or in the course).

The model extracted on the previous step is a subject to the usual drawbacks of the topic-based models (for a more detailed discussion see Sections 3.4 and 3.6.2):

- it is subjective (other Java tutorials and books will produce different models);
- it is coarse-grained (even the leaf-level topics can aggregate content indexable by a set of concepts);
- the semantics of topics and relations between them can be ambiguous (the structure of a tutorial is first and foremost intended to explain the domain, not to model it);
- it is often incomplete (rarely, tutorials provide the complete overview of the domain).

Yet, this model supplies means to access the material of the tutorial in terms of its topics, inference over the material in terms of topics and adapt the material in terms of topics. As a result, for a single tutorial, open-corpus content modeling (and - on its basis - personalization) will become possible. The experience of *QuizGuide* shows that topic-based personalization can be very effective. However, two major problems remain. First, as the detailed evaluation of *QuizGuide* shows, topics are not suitable units for modeling students' knowledge. Second, and even more important, if the topic-based model is used for student modeling it would allow tracing of students' progress and computing of students' knowledge only within one tutorial and only in terms of this tutorial structure. If the system needs to support multiple open-corpus collections, it will not be able to obtain an aggregated picture of students' activity with resources from these collections, as every collection produces its own model isolated from others'.

The solution to the both problems is based on a central high-quality ontology that is used for modeling students' knowledge and mediating between multiple topic-based tutorial structures. Figure 51 visualizes this procedure. Topic-based models extracted from several tutorials are mapped into the single domain ontology. The ontology is used as a domain model of

an existing AES. The resulting architecture allows this AES to trace students' actions with the central content, as well as with any of the open-corpus resources associated with the topics of external models.

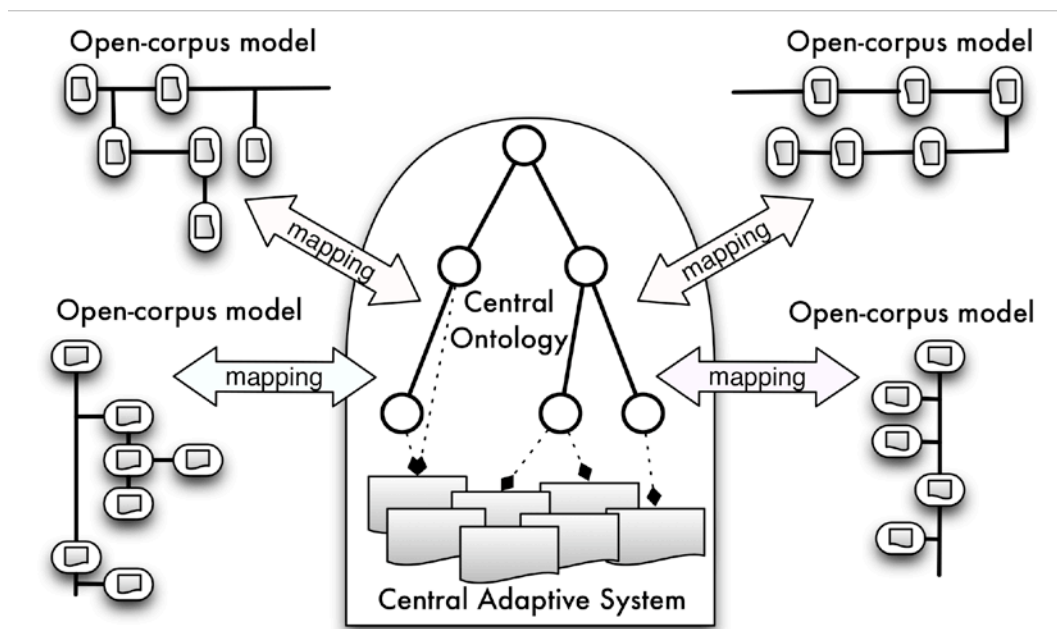


Figure 51: Mapping extracted models into central domain ontology

The mapping algorithm treats both the central ontology and the topic-based model as ontological models and follows the traditions of ontology-mapping techniques. It relies on the following four sources of information to inform the computation of topic-to-concept similarities:

- Naming labels (both: ontology concepts and open-corpus topics have textual names);
- Structural relations (an ontology would usually have several types of relations, the topic-based models would have at least one – subsumption relation²⁸);

²⁸ Identification of the exact semantics of relations in topic-based models is not a trivial task. It is not a subject of this dissertation.

- Instance corpora (concepts of a high-quality ontology have textual comments and definitions, they also index the learning resources of the central AES, every topic of the open-corpus tutorial is associated with a page or a section);
- Order of presentation (a central AES is used to teach the same subject as the open-corpus collection, their information resources are likely to follow a similar sequence of learning goals).

The output of the mapping algorithm is the matrix with rows and columns corresponding to ontology concepts and tutorial topics. The cells of the matrix contain weights (from 0.0 to 1.0) specifying how well a concept is mapped to a topic.

4.3.3 Step 3: mediated personalization of open-corpus learning material

Once the mapping between the two models is established, the systems can reason across these models. The mapping bridge between the central ontology and the tutorial model enables two principle procedures:

- trace student's actions with the tutorial topics, represent these actions in terms of the ontology concepts and update the ontology-based student model;
- request the current state of student model expressed in terms of ontology concepts, translate it into the tutorial topics, adapt the content, structure or presentation of the tutorial according to the implemented adaptation technique.

Figure 52 summarizes the principle relation between the components of the central AES and the open-corpus material, as well as the information flow across these relations.

The important aspect of the proposed solutions is that it does not simply allow connecting this particular tutorial with the central adaptive system; it stays open for more content to come,

thus providing a real open-corpus personalization platform. As a result, it is possible for students to receive unprecedented support in terms of adaptive access to open-corpus instructional material.

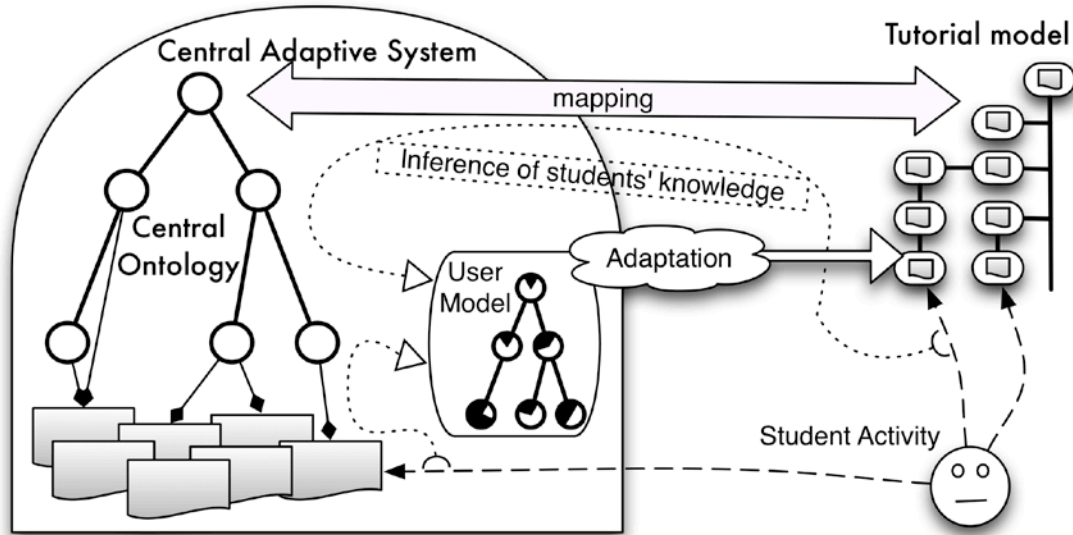


Figure 52: Meditated personalization of open-corpus learning material

4.4 ADVANTAGES OF THE PROPOSED APPROACH

The approach described in this chapter has a number of positive characteristics that distinguish it from previous solutions.

4.4.1 Independence of the human author

Probably, the most critical feature of any solution towards open-corpus personalization is the degree of automation. Open architecture and ability of the system to expand beyond the borders of the predefined set of documents is important but not enough. What should make the solution scalable and sustainable is a fully automated procedure for modeling external content. It is

especially important in the field of e-Learning, as AESs traditionally rely on semantic content models. Such models are hard to create manually, which emphasizes the importance for the solution to be independent of the human author. Such models are hard to create automatically, which highlights the importance of this dissertation's contribution. The approach proposed here expects from an author only the entry point to the open-corpus collection. Identification of individual learning objects, connecting them together and modeling with the elements of the domain semantics is done automatically.

4.4.2 Independence of the domain

Another crucial aspect of a successful open-corpus implementation is the support of the broad range of subject domains. In order for the dissertation approach to work, a core ontology should be developed. Once available, it will serve as a reference point for integrating any content collection in the target domain. At the same time, the approach places little restrictions on the domain itself and employs no domain-specific tools or heuristics. It can be extended to benefit of such tools for extracting more detailed or complete representations of content collections, but it does not require them. As a result, if there exist a domain ontology and a collection(s) of open-corpus content, the proposed approach should be able to provide open-corpus personalization of this collection.

4.4.3 Independence of the adaptation technology

It is important to mention that the proposed infrastructure can support a very diverse range of personalization technologies. The open-corpus topics can be adaptively sequenced according to

the order of learning goals defined in the central system. The most relevant topic(s) can be adaptively recommended to a student based on the current level of knowledge or the current task at hand. On the other hand, the irrelevant topics can be adaptively filtered to prevent students from accessing topics that they are not yet ready for, or that they have already mastered. The topics can be provided with adaptive annotations navigating students through the content of the tutorial. The next chapter will present the proof-of-concept implementation of the proposed approach; the implementation combines two different personalization techniques to demonstrate this point.

4.4.4 Independence of the content collection

The way a content collection is formatted and structured will influence the quality of the extracted model, the quality of its mapping to the central ontology and, at the end, the quality of personalization. The more detailed is the structure of the collection, the more thorough was the author when dividing the collection into topics and subtopics, choosing descriptive names for them, and augmenting the collection with additional sources of information and structure, such as table of contents, index, keywords, types of relations, etc., the better will be the final result. However, these are desired but not necessary conditions for the approach to function. The only necessary requirement is the presence of some coarse-grained structure that allows to associate a piece of content with a topic, and a topic with a naming label. A hierarchy or a network of topics can be also instrumental. Such lenient restrictions on the structural and formatting quality of the source content allow the approach to be applicable for a broad range of collection types:

- learning object repositories, such as Connexions²⁹ and Curriki³⁰;
- encyclopedias, such as Wikipedia³¹ and Britannica³²
- dictionaries such as Medical Subject Headings³³ and FOLDOC³⁴
- multiple online tutorials, textbooks, manuals, etc.

4.4.5 Independence of the application field

This dissertation focuses on the field of e-Learning, and the solution described here is primarily targeted to support open-corpus personalization in the context of e-Learning. However, the ability to provide adaptive access to open-corpus information resources based on semantic modeling of the domain, the content and the user is a desirable functionality for many information systems. If the central user modeling component instead of keeping track of users' knowledge will estimate their interests or information needs, the resulting infrastructure can be used to implement a more general-purpose application supporting personalized access to open-corpus information in a more general context. For example, an adaptive help system can recommend to a user pages of an online manual. An adaptive travelling assistant can be used to navigate through a traveling-related forum or a blog. An adaptive search component can be employed to retrieve Web-pages that are relevant not only to a user query but also to her interest profile.

²⁹ <http://cnx.org/>

³⁰ <http://www.curriki.org/>

³¹ <http://www.wikipedia.org/>

³² <http://www.britannica.com/>

³³ <http://www.nlm.nih.gov/mesh/>

³⁴ <http://foldoc.org/>

4.5 CONCLUSION

This chapter is the central part of this dissertation. It has discussed the main assumptions the proposed approach towards open-corpus personalization for e-Learning, presented the conceptual description of its three main steps and outlined its major advantages.

The previous chapter ended with analysing the previous work done by the author in the field of open-corpus personalization and content modeling, as well as its relation to the state-of-the-art (see Section 3.6). One of the concluding thoughts was the lack of domain and task-independent approaches for automatic open-corpus content-based personalization. The approach presented in this chapter is filling this niche (Figure 53).

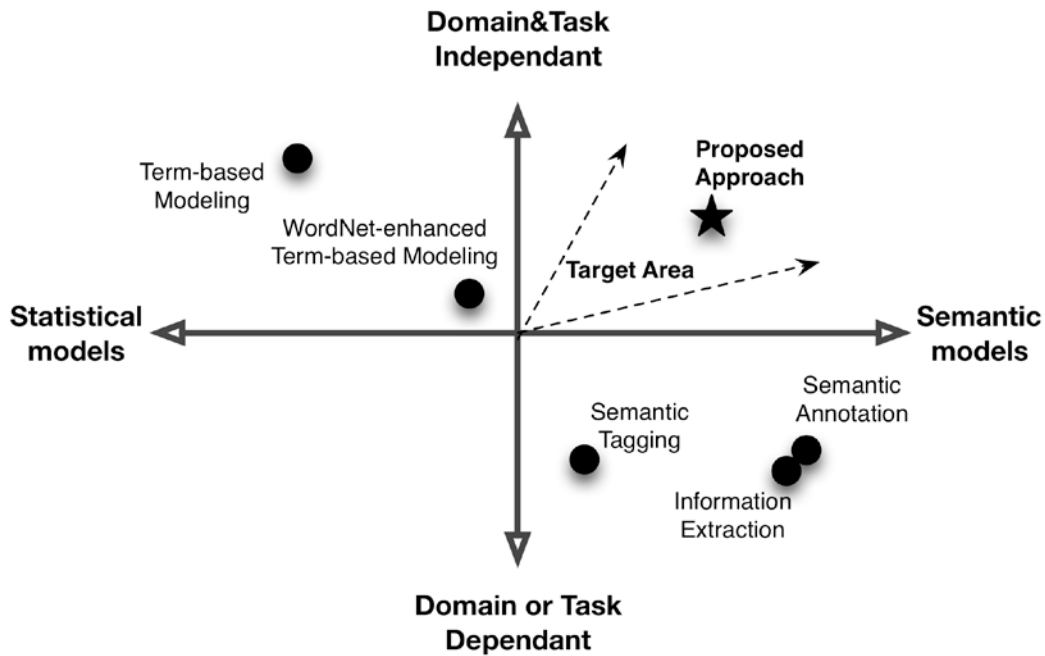


Figure 53: The proposed approach in the spectrum of open-corpus content-based personalization technologies

5.0 ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION SERVICE

5.1 INTRODUCTION

This chapter describes the details of proof-of-concept implementation of the approach presented in Chapter 4.0. Ontology-based Open-corpus Personalization Service (*OOPS*) has been designed to adaptively present supplementary reading material to students solving self-assessment exercises. A prototype of the service has been developed in the domain of introductory Java programming. The role of this chapter is to demonstrate the feasibility of the proposed approach and, together with Chapter 7.0 describing the results of the *OOPS* evaluation, verify the practical value of the main contributions of this dissertation.

The rest of the chapter is structured as follows. It starts with the brief analysis of the subject domain and the instructional scenario, used for the proof-of-concept implementation (Section 5.2), including the developed Java ontology and the extracted model of the open-corpus Java textbook. Next, it presents the student interface (Section 5.3) and the architecture (Section 5.4) of the service. Section 5.5 discusses the main aspects of the adaptation technologies implemented in *OOPS*. Finally, Section 5.6 concludes the chapter.

5.2 THE PROOF-OF-CONCEPT IMPLEMENTATION SETTINGS

As discussed in the previous chapter, the general approach is independent of the domain. The architecture and the design of the *OOPS* service implementing the approach also follows this principle. Both the approach and the service are also largely independent of a specific instructional scenario, and can be used in conjunction with different types of adaptive systems and kinds of instructional content. However, due to certain characteristics of *OOPS*, some scenarios of its usage might be more suitable.

5.2.1 The subject domain and the instructional scenario

The service has been developed as a value-added service that is used in parallel with another system (called the central system from now on) and augments it with adaptive access to supplementary reading material. A typical scenario would be: the central system serving learning problems or exercises for practice or self-assessment. This way, the relevant reading material provided by *OOPS* could benefit students struggling with a difficult exercise and support meaningful learning experience. The central system, in its turn would be responsible for objective assessment of students' knowledge in the domain. Its exercises should be connected to the central ontology and it should report students' activity to the central user modeling component in order to populate the ontology-based models of students' knowledge. The central system can be an AES responsible for student modeling and adaptation of exercises; or it can be a component in a distributed architecture that is responsible only for serving practice exercises and assessing students' knowledge.

As a proof-of-concept, for this dissertation, *OOPS* has been built into an existing infrastructure that helps students to learn Java programming language. The learning material of the central system is targeted at the entry-level university students who have never studied Java (or even programming) before.

5.2.2 The core domain ontology

For modeling Java programming domain, the content of the central system, as well as the knowledge of students a central Java ontology has been developed. Figure 54 presents a screenshot of the Protégé tool used to develop the ontology.

It defines 344 *owl:Classes* and uses 3 relations (*rdfs:subClassOf*, *java:realizedTo* and a pair of inverse relations *java:isPartOf* – *java:hasPart*). The goal of the ontology is to describe the entire domain of Java programming. On the top level it subdivides the domain into:

- *JavaLanguage* that contains all the classes describing basic elements of Java programming language, including data types, statements, expression, program structure etc.;
- *JavaOOP* that models important concepts and principles of object-oriented programming and design with Java;
- *JavaStandardLibrary* that supplies a collection of the most used packages, classes, interfaces, methods and objects from the Java Standard API Specification;
- *JavaTechnology* that is concerned with Java APIs and techniques helping programmers to deal with a set of important tasks, such as networking, database connectivity, multithreading, XML processing, etc.

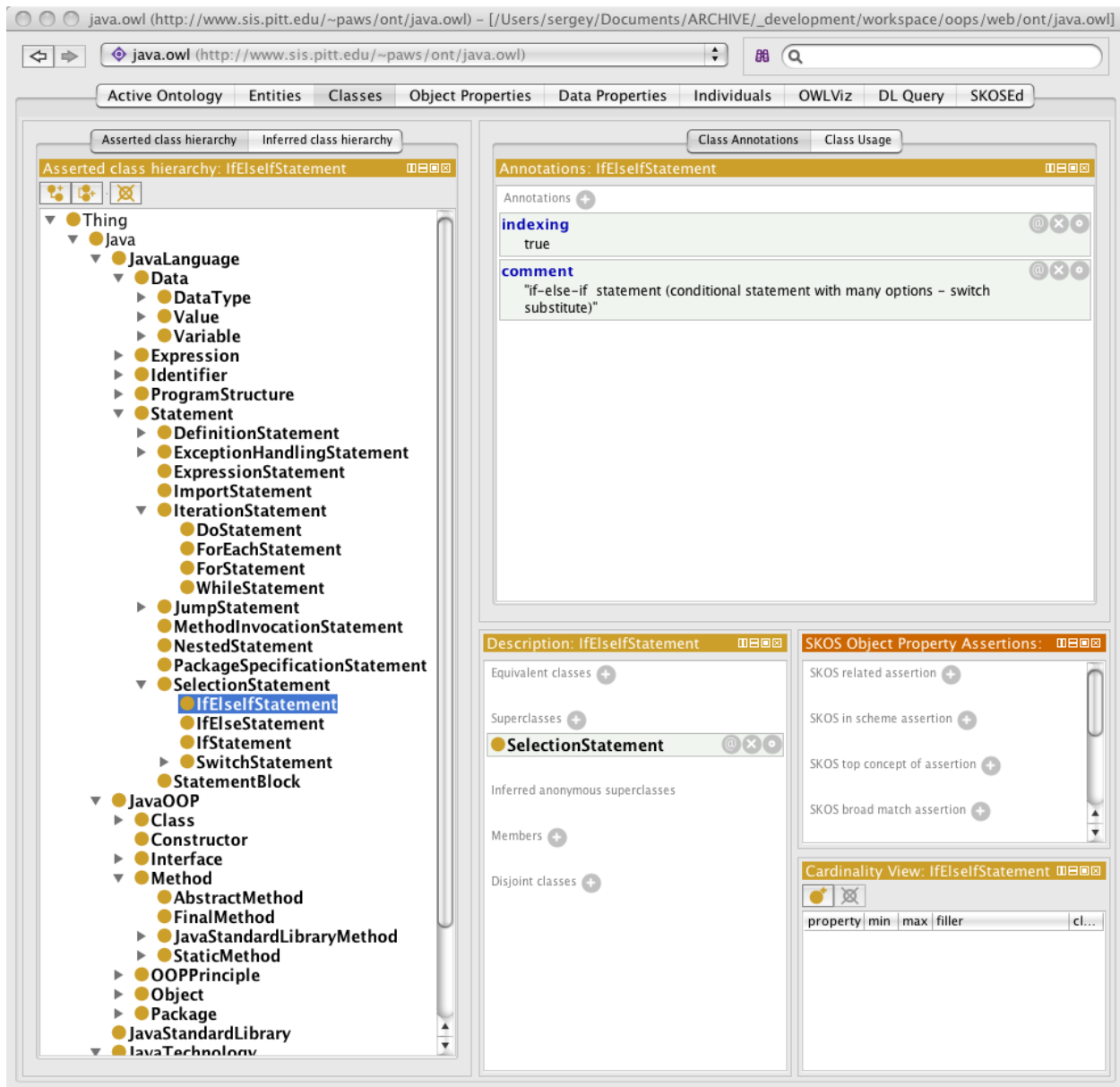


Figure 54: Central Java ontology³⁵

The relations used in the ontology specify one of the three situations:

- the standard *rdfs:subClassOf* is used as the main structuring relation; it specifies a subsumption relation, a super-class/sub-class relation (e.g. `<java:IfElseStatement>` `<rdfs:subClassOf>` `<java:SelectionStatement>`);

³⁵ <http://www.sis.pitt.edu/~paws/ont/java.owl>

- the pair *java:isPartOf* – *java:hasPart* defines the partonomy relation, a whole/part relation (e.g. *<java:AbstractMethod> <java:isPartOf> <java:AbstractClass>*); *java:isPartOf* and *java:hasPart* are inverse to each other (if A *java:isPartOf* B, then B *java:hasPart* A); they are also transitive relations (if A *java:isPartOf* B and B *java:isPartOf* C, then A *java:isPartOf* C);
- the *java:relatedTo* is used to denote a situation when two classes from different branches of the ontology model related concepts, but the exact type of this relation is not defined (e.g. *<java:ReturnStatement> <java:relatedTo> <java:ReturnedValue>*); *java:relatedTo* is a transitive relation, as well, and it also is a symmetric relation (if A *java:relatedTo* B, then B *java:relatedTo* A).

5.2.3 The open-corpus content collection

An electronic version of an introductory Java textbook has been used as a sample open-corpus collection of instructional material (Horstmann 2007). A topic-based structure has been extracted from the textbook and serialized as an RDFS model (Figure 55). The topic-based model consists of 288 *owl:Classes* (every class corresponding to a topic). Only one relation type is used – *rdfs:subClassOf*. There are only two levels of topic nesting: top book class -> chapter class -> exact topic class. Every topic has an id specifying its order within the chapter and the order of the chapter within the book. Every topic has an *rdfs:label* that contains its title as extracted from the book. Every topic has an *rdfs:seeAlso* that contains a URL of the topic's page.

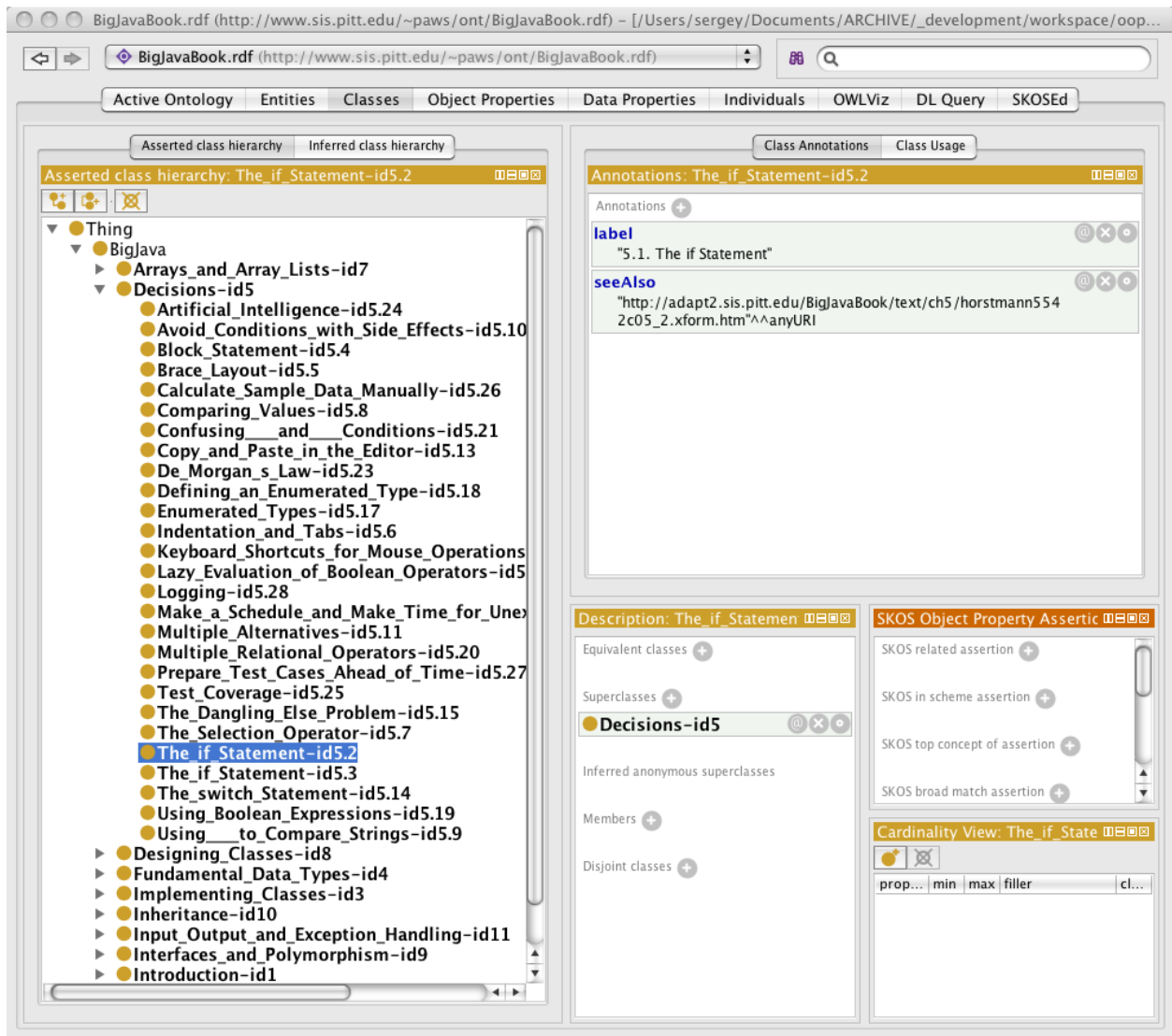


Figure 55: Extracted model of the sample Java textbook

5.3 THE INTERFACE OF THE SYSTEM

The goal of *OOPS*, as an e-Learning application, is to recommend students instructional material for supplementary reading. It does not support the main learning activity, and does not serve the main learning content; instead it augments another e-Learning system with an *added value* of relevant reading content. Therefore it has been developed as a value-added service.

The student interface of *OOPS* has been designed as a widget working with another system. It has two interaction phases: recommendation (when a student is presented with a list of recommended pages) and reading (when a student is studying recommended material).

5.3.1 Recommendation phase

Figure 56 presents a screenshot of the recommendation phase. Area “*B*” is the interface of the main system. The main system, running *OOPS* widget was *QuizJET* (Hsiao, Brusilovsky et al. 2008). *QuizJET* serves online self-assessment exercises in the domain of Java programming. A typical exercise requires students to analyze a simple Java program and answer a question about the final value of one of the variables or a console output produced by the program. *QuizJET* is described in more details in Appendix D.

Area “*A*” presents a list of recommendations produced by *OOPS* for the current exercise of *QuizJET*. The list can contain up to five items depending on the number of relevant recommendations *OOPS* could produce. A student is free to choose any of the items or ignore them altogether. *OOPS* does not interrupt student’s interaction with the main system and does not require a student to use its service, it merely recommends. Every item in the list is a topic label from the harvested open-corpus content collection. The order of an item in the list is determined by its relevance to the current activity of main system (in our case – current *QuizJET* exercise). The relevance metric is computed by the mapping algorithm and described in Section 5.5.4.

The recommended items are provided with adaptive annotation in form of little man-shaped colored icons. The coloring of an icon annotating a topic represents the amount of knowledge a student has demonstrated for the learning material behind this topic. The knowledge though is measured not in terms of a topic, but in terms of central ontology concepts mapped to

the topic and provided by the central student model. The annotation level is computed as a weighted aggregation of knowledge levels for all concepts mapped into the topic. The annotation algorithm is described in more details in Section 5.5.5.

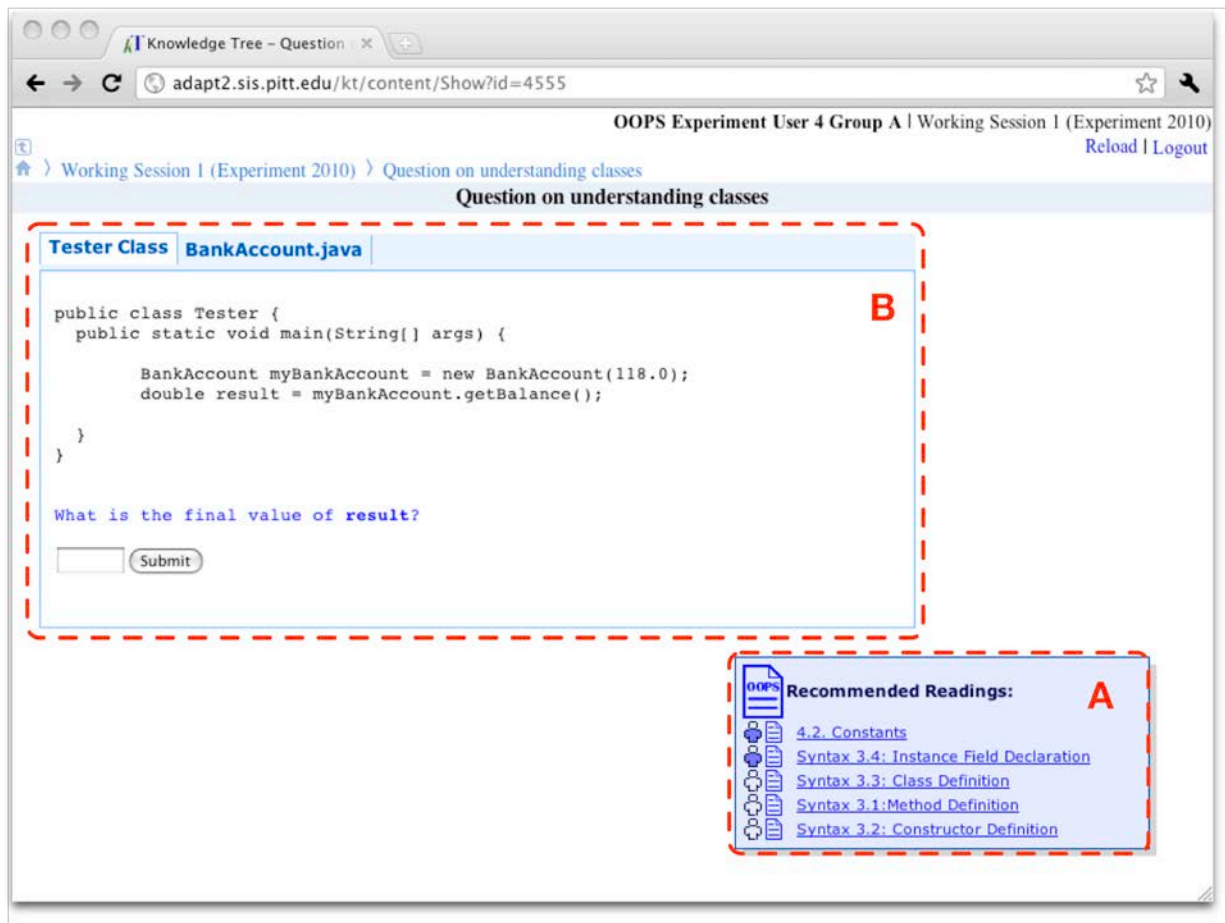


Figure 56: OOPS interface: list of recommendations

5.3.2 Reading phase

Once a student decides to accept a recommendation by clicking on a topic link, s/he goes into the reading phase of the OOPS interface (Figure 57). In this phase, *OOPS* provides a student with an opportunity to read the actual material behind the topic link. Naturally, the *OOPS* widget

becomes bigger and its interface changes. It contains three main areas. Area “A” is the content area, where the content of the selected recommendation is presented. Area “B” is the navigation area, where the links to the previous and the next topics are presented, should the student choose browsing the structure of the open-corpus collection. Area “C” contains two buttons that allow the student to exit the reading phase and to report whether s/he has found the recommendation useful for the current learning task or not. Once the student leaves the reading phase, *OOPS* interface switches to the recommendation phase again.

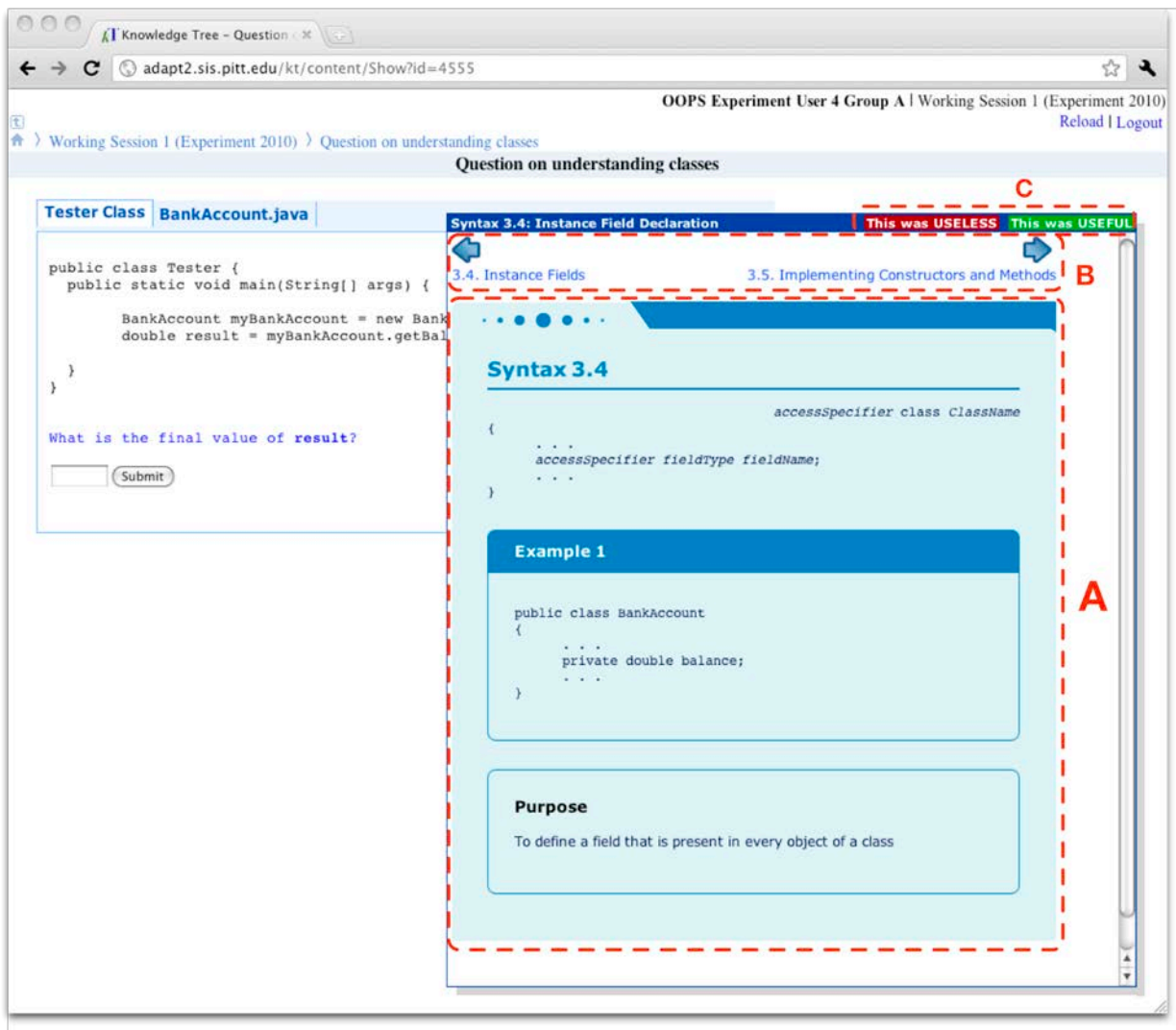


Figure 57: OOPS interface: reading a recommendation

5.4 THE ARCHITECTURE OF THE SYSTEM

From the architecture perspective, on the top level, there are two main modules of *OOPS*:

- the interaction module that runs the student interface and communicates with several different components online;
- the modeling module that harvests open-corpus content collections, extracts their topic-based models and maps them into the central ontology in the offline mode.

5.4.1 Interaction Module

As an e-Learning service, *OOPS* operates as a value-added service within the ecosystem of other components of *ADAPT*² architecture. *ADAPT*² is a distributed architecture for adaptive Web-based learning. It advocates for decoupling functionally of classic subsystems of adaptive educational systems among dedicated and interchangeable Web-based services. Important types of such services are:

- user modeling server that records students' activity and infers their characteristics (such as knowledge in the domain),
- learning portal, which is an entry point to the courses and aggregator for learning material;
- content providers, that serve different kinds of learning material;
- value-added services that enrich learning activity with extra feature, such as adaptation.

More details about *ADAPT*² are given in Appendix A. Appendix B and Appendix C provide descriptions of two important components of *ADAPT*² – user modeling server *CUMULATE* and learning portal *Knowledge Tree*.

Figure 58 visualizes how *OOPS* communicates with three other systems within *ADAPT*² in order to bring students recommended and annotated reading material:

1. The student logs into the learning portal *Knowledge Tree* that aggregates multiple learning resources including *OOPS* –empowered *QuizJET* questions.
2. When an individual question is launched, *OOPS* acts as a wrapper:
 - it receives the authentication information from *Knowledge Tree* (namely, student login, session id, and the URL of the student model server),
 - uses it for initializing itself and
 - passes this information to *QuizJET*.
3. In order to initialize, *OOPS*:
 - requests the concept-based index of the question being loaded from *QuizJET*
 - populates the list of topics for recommendation based on the topic relevance metric;
 - retrieves the current state of student’s knowledge from the user modeling server *CUMULATE* for the concepts mapped into the presented list of topics
 - computes the weighted average level of knowledge for every presented topics and annotates them with appropriate icons.
4. As a result a student is presented with an interface populated by three different systems:
 - *Knowledge Tree* fills the upper frame with functionality that helps the student to move around the course and the portal;
 - *QuizJET* controls the interaction of the student with the main learning activity at the moment - the self-assessment exercise;
 - *OOPS* produces the recommendation widget.

5. Any student's action within any of the three systems is reported to the user modeling server *CUMULATE*:
 - Answering an exercise within *QuizJET*;
 - Choosing another learning activity within *Knowledge Tree*;

Accepting a recommendation, navigating through the content, providing feedback on the usefulness of the recommendation within *OOPS*.

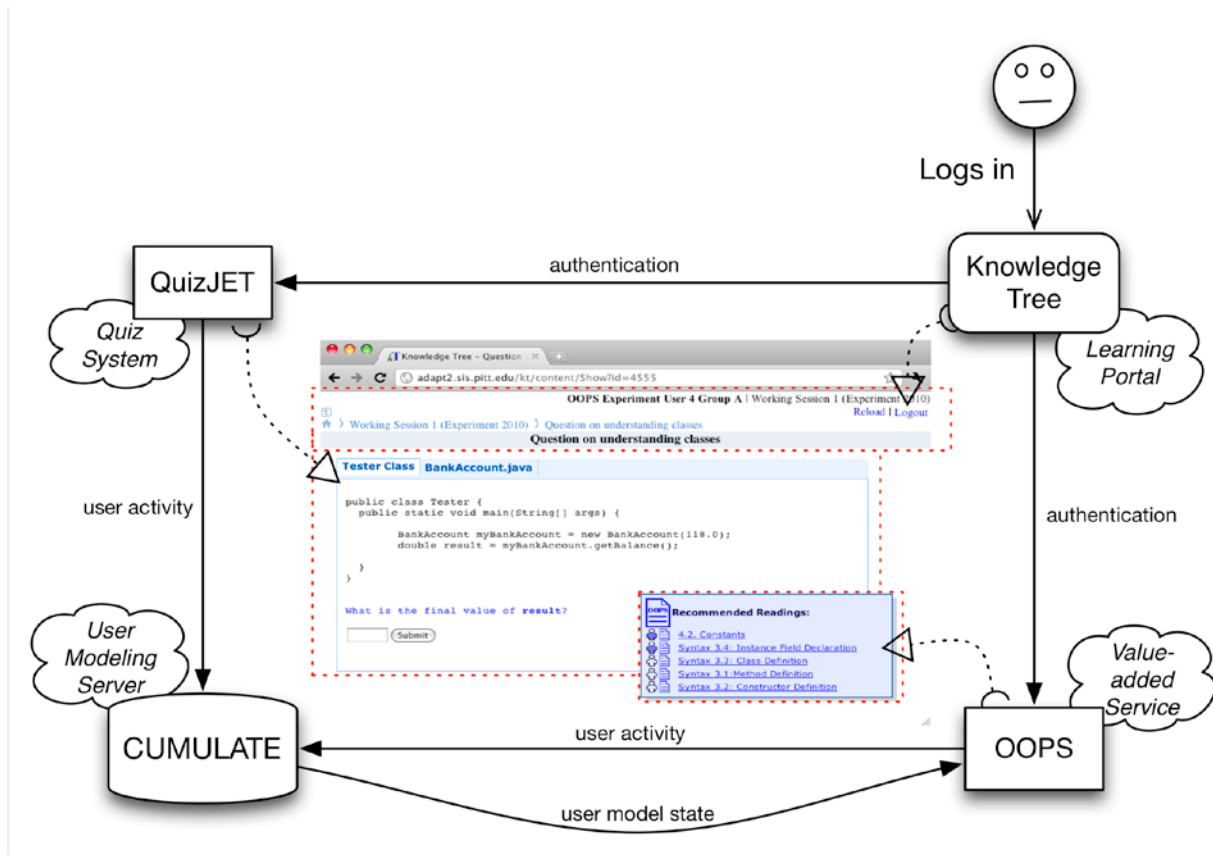


Figure 58: OOPS architecture: communication with other components of *ADAPT*²

5.4.2 Modeling module

This module of the system works separately from the interaction module. Unlike interaction module, the modeling module produces its results offline. Depending on the size and the

structure of the harvested open-corpus collection, as well as, the size of the central domain ontology, the modeling process can take from minutes to hours.

Figure 59 demonstrates a general workflow for the extracting topic-based models from open-corpus learning content collections. It starts with a teacher providing a URL of the index page that contains the table of contents of the collection. On the next step the crawler program goes through the table of contents and extract the initial topic structure of the collection, including the name labels and the hierarchical relations between topics. Then the crawler continues to the individual pages of the collection. They provide two main kinds of information: extra links between pages that contribute to the topic hierarchy refinement and the formatting of pages. The format of pages is analyzed in a straightforward way: headers enhance topic structure with new levels of hierarchy (HTML tags <h1>, <h2>, etc.), while texts corresponding to headers become individual topics' pages (they are extracted, provided with CSS, external links and buttons are removed, internal links between pages and links to graphics are localized).

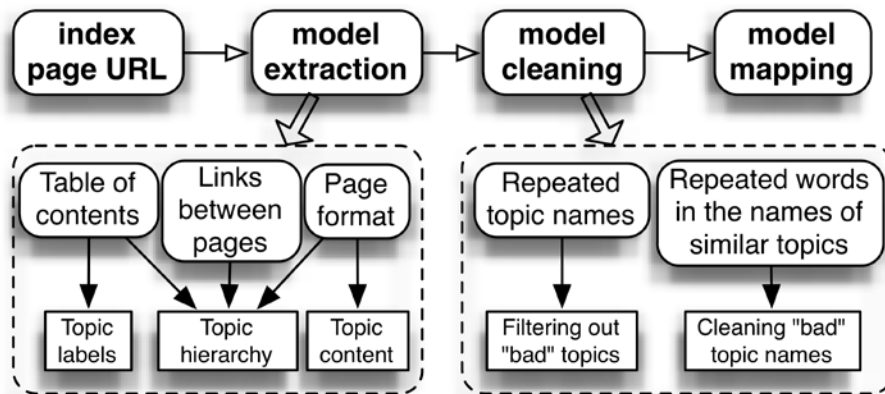


Figure 59: The general workflow for modeling an open-corpus content collection

On the next step the extracted model needs to be cleaned of the irrelevant, non-informative not-domain-describing topics, and the names labels of good topics need to be cleaned of irrelevant, non-informative and not-domain-describing words. Tables of contents of many tutorials and books contain such topics as “*Exercises*”, “*References*”, “*Questions*”. Topics like this contain little material that can be used as supplementary reading. They also provide little information for the consequent mapping algorithm. For reason like these *OOPS* modeling module filter these topics out. There are three kinds of information that can be used for that:

- A dictionary of common name labels;
- A procedure for detection of topics with the same names that reoccur in all chapters in the same place (beginning or ending of the chapter);
- The heuristics for detection of the content of such topics (they usually consist of long lists of items: exercises, questions, references, etc.)

For the Java textbook used in this dissertation these procedures allowed to filter out about 25% of topics, reducing the number of topics in the model from 370 to 288.

The second kind of cleaning that has been applied is the cleaning of the topic names. Some topics are good and informative, but their names carry non-informative words or phrases, such as “*Summary*”, “*Advanced Topic*”, “*Common Error*”, “*Productivity Hint*”, etc. These words and phrases are annotations of a kind, included by an author in order to inform a reader of the nature of this particular topic. These phrases can be important for the reader, but they damage the precision of the mapping algorithm. Therefore it has been decided to keep the full names for the presentation of topics in the *OOPS* interface, but remove these unnecessary phrases from the mapping algorithm input. The list of methods for detecting these words is similar to the previous list with one exception: the topics containing these phrases can occur in any place in the chapter.

The last step of the open-corpus content modeling workflow is mapping of the extracted and cleaned topic-based model into the central ontology. A dedicated algorithm has been developed for this purpose. It is described in the Chapter 6.0 .

5.5 PERSONALIZATION IN *OOPS*

In order to fully explore the feasibility of the proposed approach the *OOPS* service implements a complex of personalization technologies. The current section will provide a detailed explanation of these technologies including the unique features of the resulting personalization infrastructure.

One of such important features is the maintaining of cross-content personalization. *OOPS* is one of the few examples of an AES supporting students' work with the content resources of multiple types (self-assessment exercises and supplementary reading fragments). For open-corpus AES this is a crucial functionality. The ultimate goal of the open-corpus personalization is to adaptively support user's access to all relevant resources on WWW independently of their types. Yet the system needs to recognize the type of any resource and present them to a student in a coherent and non-contradicting manner. For example, *OOPS* presents textbook pages as reading resources relevant to the current exercises that a student might have difficulties with. Yet, the interface of *OOPS* does not require students to use the help of *OOPS* if they do not need it. The evaluation of the service presented in Chapter 7.0 proofs that students benefit of such arrangement.

5.5.1 *QuizJET* exercise indexing

Even though *QuizJET* itself is not an adaptive system, its exercises are used to assess an model students' knowledge in terms of domain knowledge. Therefore every exercise of *QuizJET* has been indexed with the concepts from the central Java ontology. Figure 60 graphically represents modeling of *QuizJET* material. The indexing follows a similar procedure described in Section 3.3.1.2. Exercises are ordered based on the sequence of learning goals – every learning goal combines several exercises. A concept can play one of two roles in an exercise's index:

- it can be a prerequisite – a concept that students need to learn in one of the previous learning goals, or
- it can be an outcome – a concept that students learn while working with the exercise.

Every relation between a concept and an exercise receives a weight equal to the number of the concept occurrences in the exercise content.

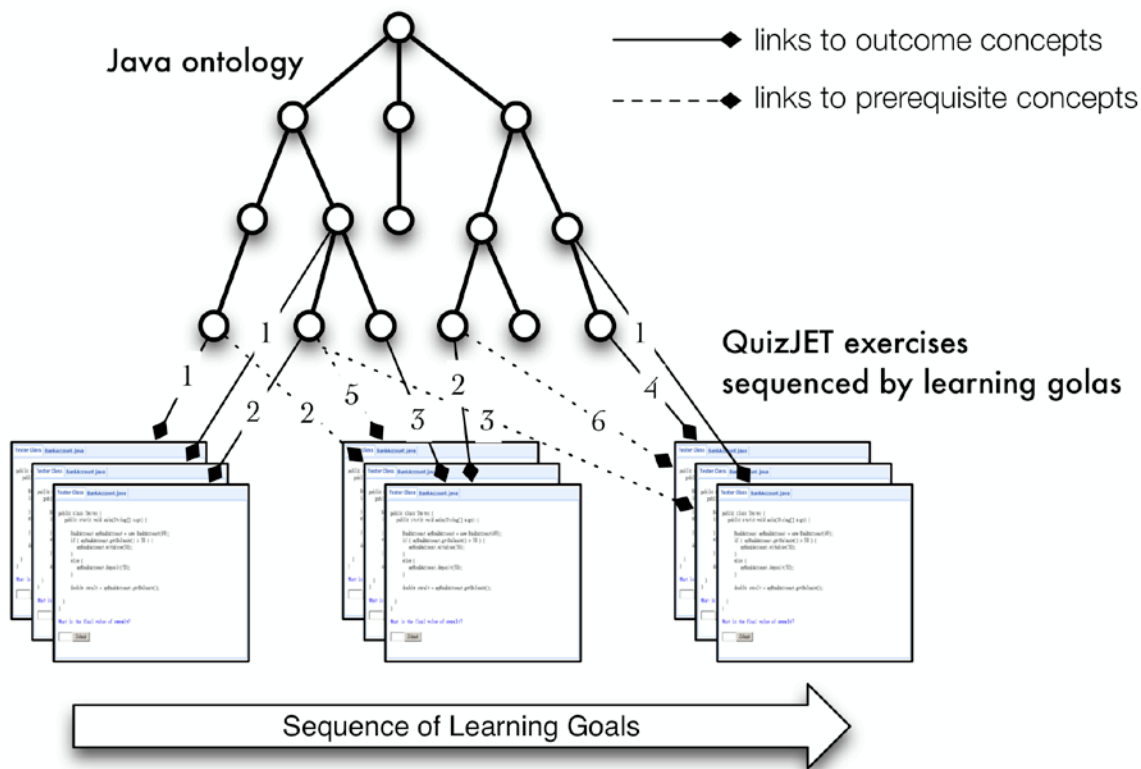


Figure 60: QuizJET exercise indexing

5.5.2 Modeling students' knowledge of Java programming

Student modeling server *CUMULATE* has been used for estimating students' knowledge of Java and reporting them to *OOPS*. If a student submits a correct answer to a *QuizJET* exercise, *CUMULATE* increases his/her knowledge levels of all prerequisite and outcome concepts from the index of this exercise. If a student submits an incorrect answer, *CUMULATE* ignores it, i.e. negative knowledge updates are not possible – *CUMULATE* models no forgetting. A knowledge level of a concept starts with 0 (no exercise involving this concept has been answered correctly yet) and asymptotically grows towards 1 (it can never reach 1, though).

When *OOPS* requests from *CUMULATE* current levels of the student's knowledge it receives an XML-serialized report for all concepts in the domain. Figure 61 presents an example of the *CUMULATE*'s report on student's knowledge. For every concept the knowledge level is estimated on several levels of cognitive activity according to Bloom's taxonomy of educational objectives (Bloom 1956). *QuizJET*'s updates are aggregated on the "application" level.

```
<?xml version='1.0' encoding='UTF-8' ?> <report>
  <user>myudelson</user>
  <user-app-hash>2009-06-11 20:20:05.545</user-app-hash>
  <user-hash>2009-06-11 20:20:05.545</user-hash>
  <concepts>
    ...
    <concept>
      <name>add</name>
      <cog_levels>
        <cog_level>
          <name>application</name>
          <value>0.6266574470377776</value>
        </cog_level>
      </cog_levels>
    </concept>
    <concept>
      <name>sub</name>
      <cog_levels>
        <cog_level>
          <name>application</name>
          <value>0.36388277385950873</value>
        </cog_level>
      </cog_levels>
    </concept>
    ...
  </concepts>
</report>
```

Figure 61: *CUMULATE*'s report on student's knowledge³⁶

³⁶ From http://adapt2.sis.pitt.edu/wiki/CUMULATE_protocol

5.5.3 Two models of the textbook

An introductory textbook on Java was chosen as a sample open-corpus collection of reading material (Horstmann 2007). Two content models for this textbook were created. A topic-based model was generated automatically as described in Section 5.4.2. This model was used to support open-corpus personalization of *OOPS*. The second content model was created manually, based on careful indexing of every textbook page with the individual concepts of the central Java ontology. This model supported the closed-corpus version of the *OOPS* service, which was used as one of the control conditions during the *OOPS* evaluation experiment (see Chapter 7.0).

On average, every page in the closed-corpus model has been indexed with 3 ontology concepts. The same page in the open-corpus model corresponded to a single topic. In addition to manual indexing, irrelevant pages were manually filtered from the closed-corpus model. In the case of open-corpus model, filtering was done automatically based on a set of heuristics described in Section 5.4.2. As a result, the cleaned open-corpus model included 277 pages, when the closed-corpus model – only 157.

5.5.4 Recommendation of reading material

One of the two adaptation technologies used in *OOPS* is recommendation: *OOPS* recommends links to the pages of reading material from open-corpus collections. As discussed in Section 4.4.3 the principle approach implemented in *OOPS* can support a wide range of adaptation technologies. Recommendation has been chosen due to the several reasons:

- it is a very popular technology (arguably, the most dominant adaptation technology currently on the WWW);

- it allows for unobtrusive personalization (*OOPS* is a value-added service for supplementary reading – it should be available when it is needed, but should not obstruct students’ work with the main instructional content – self-assessment exercises);
- it allows to implicitly several other adaptation technologies (when the list of topics is chosen for recommendation, adaptive filtering and adaptive ranking are also used).

On the architecture level, the recommendation algorithm takes into account the current exercise a student is working on, but not the level of student’s knowledge. In other words, the recommendation algorithm provides context-based adaptation: the list of recommended reading resources is populated based on the current learning context defined by the content model of exercise at hand. The link from the exercise’s concept-based model to the recommended topic and is established based on the results of the mapping between the ontology and the model of the textbook. Figure 62 shows how the individual concept-to-topic similarity computed by the mapping algorithm participate in producing the list of recommended topics.

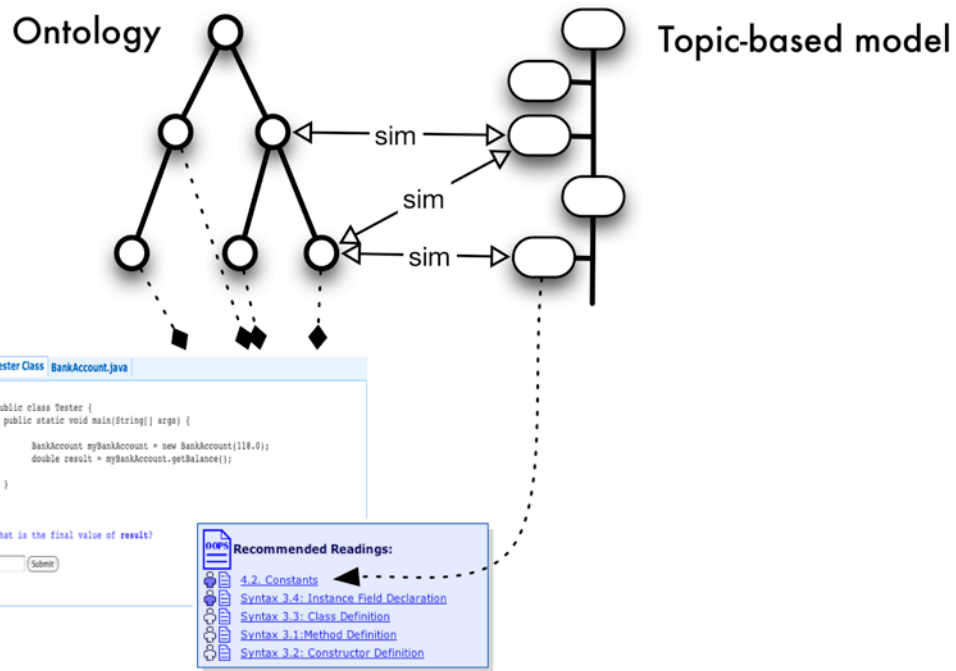


Figure 62: OOPS uses model mapping to order and recommend open-corpus reading resources

Equation 9 formally defines how the topic recommendation rank is computed. All topics are ordered based on the sum of their similarities to all outcome concepts of the current exercise. The top five topics constitute the list of most relevant reading to the learning objectives of the exercise and presented to a student.

Equation 9: Determining the recommendation rank for a topic

$$\mathit{rank}(\mathit{topic}_j) \leftarrow \sum_{i=1}^N \mathit{sim}_{ij}, \text{ where } N \text{ is the number of outcome concepts indexing the exercise.}$$

5.5.5 Adaptive link annotation

In addition to the context-based adaptive recommendation, *OOPS* also implements progress-based adaptive link annotation. Every topic is annotated with an icon reflecting the current level

of student's progress with the material relevant to the topic. The goal of the icons is to draw students' attention to the most important topics – those where not enough progress have been made. Adaptive annotation, as another example of an adaptation technology supported by the proposed approach, has been chosen for several reasons:

- progress-based adaptive link annotation is one of the most popular adaptation technology used in Web-based AESs, it has been proven to have several important positive effects on students' learning;
- similar to adaptive recommendation, adaptive annotation is an unobtrusive technology;
- from the interface perspective, it has been easy to combine link annotation and link recommendation in a single widget;
- annotation of links with progress-based icons supports an intuitive interface built on a clear metaphor (students easily understand the meaning of adaptive icons, which is especially important within the time constraints of a controlled evaluation study).





Figure 63 shows four icons that represent four possible levels of annotation.



Figure 63: Levels of adaptive annotation in OOPS

To produce the annotation of the specific topic for the specific student and the specific moment of time, *OOPS* requests the concept-based student model from *CUMULATE*. For every topic a list of mapped concepts is populated. Concepts with similarity values lower than a threshold = 0.1 are filtered out. The rest of concepts together with their knowledge levels participate in computing the exact annotation based on a simple algorithm (see Figure 64).

```

topic.knowledge.level = 0
For i = 1 To number.of.concepts.mapped.to.a.topic
    topic.knowledge.level += knowledge.of.concept[i]
Next i
topic.knowledge.level /= number.of.concepts.mapped.to.a.topic
If (topic.knowledge.level > 0.75)
    annotation = 
Else If (topic.knowledge.level > 0.50)
    annotation = 
Else If (topic.knowledge.level > 0.25)
    annotation = 
Else
    annotation = 
End If

```

Figure 64: Adaptive link annotation algorithm in OOPS

5.6 CONCLUSION

A practical implementation of the main dissertation approach has been presented. *OOPS* provides adaptive access to open-corpus textual content as a supplementary reading for students solving self-assessment exercises. It has been implemented as a value-added service within the framework of *ADAPT*² architecture, and thus communicates with its several components: student modeling server *CUMULATE*, learning portal *Knowledge Tree*, and as online exercise system *QuizJET*.

Java programming has been chosen as the target domain. A Java ontology has been developed to maintain overlay student modeling and serve as a reference point for mapping open-corpus models. An electronic version of an introductory Java textbook is used as the sample collection of open-corpus reading material. Its topic-based representation has been automatically extracted based on the table of contents and section headings.

Two main personalization technologies are implemented. Context-based adaptive recommendation of topics is combined with progress-based adaptive link annotation.

Overall, *OOPS* provides a full-scale proof-of-concept for the open-corpus personalization approach proposed in the previous chapter. It has been designed to maximally demonstrate the most important characteristics of the approach: automation, domain independence, and wide support of different types of content and adaptation technologies.

6.0 ALGORITHM FOR TOPIC-TO-ONTOLOGY MAPPING

6.1 INTRODUCTION

One of the innovations of the main dissertation approach is the employment of ontology mapping in open-corpus content modeling. It is used to establish the links between the topics of automatically extracted coarse-grained model of an online collection and the concepts of the central domain ontology. Later, *OOPS* needs these links for two purposes:

- it uses them to decide, which topics to recommend as supplementary reading for the current self-assessment exercise (topics linked to the outcome concepts of the exercise will be recommended);
- it uses them to translate the knowledge of a student retrieved from the student model server and to compute the levels of adaptive annotation for the recommended topics (the knowledge levels are received in terms of the ontology concepts; the links are used to compute the topic-based knowledge levels).

This chapter provides a detailed description of the developed Algorithm for Topics to Ontology Mapping (*ATOM*). Figure 65 demonstrates the workflow of *ATOM*. The main input data for the algorithm is the ontology and the topic-based model of the tutorial serialized in RDF. The output of the algorithm is the matrix of similarities where every row corresponds to an ontology concept, every column to a topic from the collection model and every cell contains a value

between 0 and 1 that represents how similar the corresponding concept is to the corresponding topic (0 – means no similarity, 1 means a complete match). The algorithm combines four individual mappers that utilize different sources of information about the mapped models. The name-based and instance-based mappers use naming labels and instance corpora of concepts and topics to produce two individual similarity matrices. The structure-based component uses concepts' and topics' neighborhoods to merge the two previous matrixes into a single one. Finally, the order-based mapper uses sequences of topics and concepts represented as learning goals to refine the results of the prior components and produce the final similarity matrix.

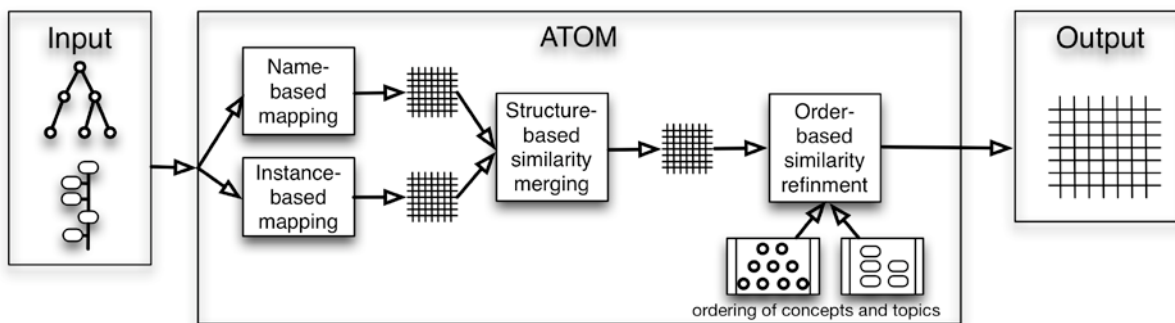


Figure 65: The workflow of ATOM

The rest of this chapter is structured as follows. Section 6.2 discusses the differences between ontologies and topic-based models from the mapping point of view. The next four sections (from 6.3 to 6.6) describe individual mapping components: the name-based, the instance-based, the structured-based, and the order-based. The last section concludes the chapter with a short discussion and the summary of important features of *ATOM*.

6.2 MAPPING CONCEPTS AND TOPICS

Section 3.6.2 summarized important characteristics of a topic as modeling and adaptation unit. From the mappings perspective, topic-based structures have another set of features influencing the choice of techniques a mapping algorithm should implement:

- Topic names are long. Although, concepts names can consist of several words as well, an average length of a topic name is much longer than the one of a concept. Topics names are created to be read by a user who is not an expert in the domain. They are chosen to explain the main idea of a chapter (or a page), and to help users navigate through the collection material. Concept names do not have to be so descriptive, and an author of an ontology does not have a goal of explaining the domain to a non-expert.
- Topic names are subjective. An author of a content collection does not have to create a “shared conceptualization of the domain” (Gruber 1995).
- Topic names are irregular. An ontology designer usually chooses a particular naming convention for the concepts (e.g. *IfStatement*, *ForStatement*, *AddOperator*, *SubtractOperator*, etc.). An author of a tutorial does not need to follow such restrictions.
- Topics differ in volume of aggregated knowledge. Some topics can be very specific, while others combine multiple notions. Ontology concepts represent discrete and cohesive elements of domain semantics.
- Topics do not always provide a balanced coverage of the domain. Some parts of domain knowledge can be more challenging from the learning perspective and therefore draw more attention from the author; in such a case he would write several sections/topics to

explain all the details. Other parts can be less important, or simply out of scope of the tutorial. Ontology designers, usually aim at providing a complete picture of the domain.

- The exact semantics of relations between topics is hard to define. The structure of a section in a textbook can be a mixture of taxonomy and paronymy. Sub-topics can be included in a section only for instructional purposes and have no relation to the super-topic, at all.

ATOM applies several techniques in order to address these discrepancies in naming, structuring and modeling practices for topics and concepts.

6.3 NAME-BASED MAPPING

Concept-naming labels usually provide the most informative source of information for a mapping algorithm. Two ontologies modeling the same area of knowledge are likely to use similar subset of concepts for it and these concepts are likely to have similar names. In the case of *ATOM*, though one of the mapped models differs from conventional Semantic Web ontology. The result of the *ATOM*'s name-based is a similarity matrix, where every cell estimates the likelihood that a given a concept is similar to a given topic. Figure 66 presents the algorithm used to compute this matrix.

- 1) For every concept $c \in C$ from the Ontology O :
 - a) Extract words from the name of c
 - b) For every extracted word $w_i^c \in name^c$:
 - filter out stop-words
 - convert words to terms by applying Porter stemming algorithm ($w_i^c \rightarrow term_i^c$)
 - compute $idf_i = \log_e \frac{|C|}{1 + |\{c : term_i^c \in c\}|}$
- 2) For every topic $t \in T$ from the topic model M : repeat the steps 1a) and 1b)
- 3) For every $c \in C$ from the Ontology O and every $t \in T$ from the topic model M compute similarity of $name^c$ and $name^t$:
 - a) For every $term_i^c \in name^c$ and every $term_j^t \in name^t$:
 - compute Levenshtein edit distance led_{ij}
 - compute the syntactic similarity measure of two strings: $ssm_{ij} = \max\left(0, \frac{\min(|term_i^c|, |term_j^t|) - led_{ij}}{\min(|term_i^c|, |term_j^t|)}\right)$
 - b) Compute the best assignment of $term_i^c \in name^c$ to $term_j^t \in name^t$ using Hungarian algorithm:
 - find the maximum name length in terms: $K = \max(|name^c|, |name^t|)$
 - extend the matrix $|name^c| \times |name^t|$ with extra rows(columns) of zeros to obtain a $K \times K$ matrix
 - apply the Hungarian algorithm to the term matrix, as a result every $term_i^c \in name^c$ is assigned to some $term_j^t \in name^t$: $term_k^c \leftrightarrow term_k^t$, where $k=1..K$
 - c) Compute the name similarity as a weighted average of similarities of assigned terms (weights are sums of their idf -values) normalized to the number of terms matched and number of terms not matched:

$$sim_{CT} = \frac{\sum_{i=1, j=1}^K (ssm_{ij} * (idf_i + idf_j))}{\sum_{i=1, j=1}^K (idf_i + idf_j) * N_{matches} * (1 + \log_e(1 + N_{no-matches}))}$$
- 4) The output of the algorithm is the matrix of similarities sim_{CT}

Figure 66: Name-based mapping in ATOM

First two stages of the algorithm rely on a set of standard IR techniques to find the relative importance of the terms across the entire corpus of terms used to name concepts in the ontology and topics in the topic model. Concept and topic names are broken into individual words. Stop words are filtered out³⁷, and then the Porter stemmer is applied to produce meaningful terms by removing “morphological and inflexional endings from words” (Porter 1980). Finally, *idf* metric³⁸ helps to weight the importance of terms like “*Break*” and “*Statement*” in the concept name “*BreakStatement*” based on the fact that the term “*Statement*” occurs in many other concepts’ names. The *idf*-weights are used later on the step 3c to compute the final similarity measure.

For example, Table 8 summarizes the result of these two stages for a concept “*BreakStatement*” and a topic “*The_break_and_continue_Statements*”. The name labels are split into words, stop-word “*The*” is filtered out, the words are stemmed and lower-cased, and *idf* scores are computed.

Table 8: Example of terms extracted from naming labels with their *idf* scores

<i>BreakStatement</i>		<i>The_break_and_continue_Statements</i>	
Term	IDF	Term	IDF
<i>break</i>	5.15	<i>break</i>	4.97
<i>statement</i>	2.08	<i>and</i>	1.81
		<i>continu</i>	4.97
		<i>statement</i>	3.36

³⁷ The standard stop words list has been modified to avoid filtering words important for the Java programming domain: *and, do, else, for, if, or, while*.

³⁸ *idf* has been chosen over *tf*idf*. *tf*-component is not relevant in this algorithm, as it is rare for a term to occur more than once in a concept name. If a name indeed contains multiple occurrences of a term, this does not provide extra evidence that the concept name is about this term.

Stage 3 incorporates the core steps of two names similarity computation. First (Step 3a), a simple syntactic similarity measure is estimated for every term of a concept name and every term of a topic name. It is based on Levenshtein’s edit distance (Levenshtein 1966) normalized using the metric proposed in (Maedche and Staab 2002). The result of Step 3a is a matrix of similarities between all terms of the concept name and all terms of the topic name. Table 9 presents the syntactic similarity table for terms extracted on the previous step.

Table 9: Syntactic similarity between terms

	<i>break</i>	<i>and</i>	<i>continu</i>	<i>statement</i>
<i>break</i>	1	0	0	0
<i>statement</i>	0	0	0	1

Finding the best combination of matches between the terms (Step 3b) can be solved as an assignment problem³⁹ using Hungarian algorithm (Kuhn 1955). This algorithm has been originally developed to find the optimal assignment of tasks to agents based on the cost matrix. ATOM applies it using one list of terms as agents and another as tasks⁴⁰. The original syntactic similarity table needs to be modified:

- the input for Hungarian algorithm is a square matrix of costs, therefore, a number of rows (or columns) with zero values have to be added to the syntactic similarity table;
- Hungarian algorithm minimizes the cost; therefore, values of the syntactic similarity table have to be transformed, as follows: $new-value = 1 - original-value$.

³⁹ http://en.wikipedia.org/wiki/Assignment_problem

⁴⁰ As Hungarian algorithm minimizes the cost, values of the similarity matrix, first, have to be transformed, as follows: $new-value = 1 - original-value$

The output of the Hungarian algorithm is a vector, with values corresponding to the indices of agents and indices corresponding to the indices of tasks. In the given example, the output of the algorithm will be the following vector: $\{1, 3, 4, 2\}$. Essentially, only two values are meaningful in this case: the first (assigning $term_1^{BreakStatement}$ to $term_1^{The_break_and_continue_Statements}$) and the last (assigning $term_2^{BreakStatement}$ to $term_4^{The_break_and_continue_Statements}$). Two other values refer to zero-similarity cells. The unmatched cells in the syntactic similarity table are set to zero.

At the last step (3c), the actual name similarity of the target concept and a topic is computed. The matched terms are weighted by their *idf*-values to ensure that rare terms contribute more to the name similarity than the popular ones, as they are considered more indicative of the given name. The name similarity is also normalized to the number of unmatched terms, thus promoting the situations when all terms from one name have been matched to terms from another name. It is also divided by the log value of the number of matched terms to give a slight advantage to shorter names. This has been done to demote accidental partial mappings that happen more often between longer names. The final value for the chosen pair of names is: 0.324 .

The Stage 3 is repeated for all concepts from the ontology and all topics from the topic-based model resulting in a large similarity matrix.

Concept and topic names do not always provide enough information to detect the relevance between them. Besides, in some cases, ATOM's name-based mapper computes high similarity values for concepts and topics that are only mildly relevant, or not relevant at all. A simple example of such situation is mapping of a concept "*Statement*" and a topic like "*The_for_Statement-id6.9*": since "*statement*" is the only word of the first name, it will strongly influence the mapping. Obviously, concept "*Statement*" is only mildly related to a topic about *for*-loops. Another example is a high name similarity of the concept "*InterfaceImplementation*"

and the topic “*Implementing_a_Graphical_User_Interface_GUI_-id10.35*”. Terms “*interface*” and “*implement*” have very different meanings in these name labels; these concept and topic should not be mapped. To deal with such problems, ATOM employs three more mappers that compute concept-to-topic similarities based on other sources of information than naming labels.

6.4 INSTANCE-BASED MAPPING

The second most important source of information used by ATOM is the instance corpora associated with topics and concepts.

An instance of a topic is the piece of content under the topic. The exact way of creating a model of a topic instance depends on the type and domain of the content and can utilize domain-specific and media-specific algorithms (e.g. parsing programming code, or extracting text from video). In this dissertation, a straightforward approach towards text modeling was used. Every topic in the model extracted from the textbook has a chapter or a page associated with it (e.g. Figure 67 presents a part of the page that belongs to the topic “*The_break_and_continue_Statements*”). The RDFS representation of the model stored a URL of a topic’s pages using standard `<rdfs:seeAlso>` property. For an instance-based mapping algorithm, this page becomes the only instance of the target topic. In order to process the instance, the algorithm extract text from the HTML source of the page and represents it as term vectors⁴¹, where every term is associated with a $tf*idf$ value.

⁴¹ Section 6.3 briefly describes how a term vector is created.

Advanced Topic 6.4: The break and continue Statements

Advanced Topic 6.4

You already encountered the `break` statement in Advanced Topic 5.2, where it was used to exit a `switch` statement. In addition to breaking out of a `switch` statement, a `break` statement can also be used to exit a `while`, `for`, or `do` loop. For example, the `break` statement in the following loop terminates the loop when the end of input is reached.

```
while (true)
{
    String input = in.next();
    if (input.equalsIgnoreCase("Q"))
        break;
    double x = Double.parseDouble(input);
    data.add(x);
}
```

In general, a `break` is a very poor way of exiting a loop. In 1990, a misused `break` caused an AT&T 4ESS telephone switch to fail, and the failure propagated through the entire U.S. network, rendering it nearly unusable for about nine hours. A programmer had used a `break` to terminate an `if` statement. Unfortunately, `break` cannot be used with `if`, so the program execution broke out of the enclosing `switch` statement, skipping some variable initializations and running into chaos [2, p. 38]. Using `break` statements also makes it difficult to use *correctness proof* techniques (see Advanced Topic 6.5).

However, when faced with the bother of introducing a separate loop control variable, some programmers find that `break` statements are beneficial in the “loop and a half” case. This issue is often the topic of heated (and quite unproductive) debate. In this book, we won’t use the `break` statement, and we leave it to you to decide whether you like to use it in your own programs.

In Java, there is a second form of the `break` statement that is used to break out of a nested statement. The statement `break label;` immediately jumps to the *end* of the statement that is tagged with a label. Any statement (including `if` and block statements) can be tagged with a label—the syntax is

```
label: statement
```

The labeled `break` statement was invented to break out of a set of nested loops.

Figure 67: An extract from the page associated with the topic “*The_break_and_continue_Statements*”

Every concept in a well-designed ontology should have a human-readable textual definition or a description explaining the meaning of the concept. By default, property `<rdfs:comment>` is used to store such definitions. Every concept of the Java ontology described in Section 5.2.2 has been provided with an `<rdfs:comment>`. Texts for the comments have been taken from several online Java tutorials; most of them – from the Oracle Java Tutorial⁴². For example the `<rdfs:comment>` of the concept “*BreakStatement*” is:

⁴² <http://download.oracle.com/javase/tutorial/>

“The break statement is used to break from an enclosing do, while, for, or switch statement. It is a compile error to use break anywhere else. 'break' breaks the loop without executing the rest of the statements in the block. The break statement has two forms: labeled and unlabeled.”

Such comments have been also processed into term vectors. As a result, every concept in the ontology and every topic in the extracted textbook model were associated with an instance.

ATOM uses a modified version of the *GLUE* algorithm developed by Doan et. al. to produce instance-based mapping (Doan, Madhavan et al. 2002). The *GLUE* algorithm computes the matrix of similarities between all concepts of two ontologies based on cross-classification of concepts' instances and calculation of joint probabilities for each pair of concepts from two ontologies (see Figure 68).

- 1) For every concept $c \in \mathcal{C}$ from the Ontology \mathcal{O} and every topic $t \in \mathcal{T}$ from the topic model \mathcal{M} :
 - a. Divide the set of instances of \mathcal{O} into U_o^c and U_o^{-c} , where U_o^c = term vector of $\langle rdfs:comment \rangle$ of the concept c and U_o^{-c} = term vectors of $\langle rdfs:comment \rangle$ of all other concepts from \mathcal{O} .
 - b. Divide the set of instances of \mathcal{M} into U_M^t and U_M^{-t} , where U_M^t = term vector of the $\langle rdfs:seeAlso \rangle$ page of the topic t and U_M^{-t} = term vectors of $\langle rdfs:seeAlso \rangle$ pages of all other topics from \mathcal{M} .
 - c. Train classifier L_c using U_o^c as the positive instance set and U_o^{-c} as the negative instance set.
 - d. Apply classifier L_c to the instance set U_M^t .
This will partition U_M^t into two subsets U_M^{ct} and U_M^{-ct} .
 - e. Similarly apply classifier L_c to the instance set U_M^{-t} .
This will partition U_M^{-t} into two subsets U_M^{c-t} and U_M^{-c-t} .
 - f. Repeat steps 1c – 1e with the roles of \mathcal{O} and \mathcal{M} being reversed and obtain U_o^{ct} , U_o^{-ct} , U_o^{c-t} and U_o^{-c-t} .
- 2) Compute joint probabilities of concept c and topic t :
 - a.
$$P(c,t) = \frac{N(U_o^{ct} + U_M^{ct})}{|C| + |T|}$$
 - b.
$$P(c,-t) = \frac{N(U_o^{c-t} + U_M^{c-t})}{|C| + |T|}$$
 - c.
$$P(-c,t) = \frac{N(U_o^{-ct} + U_M^{-ct})}{|C| + |T|}$$
- 3) Compute Jaccard similarity coefficient:

$$Jaccard - sim(c,t) = \frac{P(c \cap t)}{P(c \cup t)} = \frac{P(c,t)}{P(c,t) + P(c,-t) + P(-c,t)}$$
- 4) The output of the algorithm is the matrix of similarities $sim_{\mathcal{C}\mathcal{T}}$

Figure 68: Instance-based mapping in ATOM

The first phase of the algorithm tries to estimate the number of documents (instances) where topics and concept are present together. The algorithm processes one pair of concept-topic at a time. At the end all possible combinations are accounted for. First, the algorithm breaks all instances of concepts in the ontology into two sets: the set of instances that belong to the target concept U_o^c and the set of all other instances U_o^{-c} . Naturally, the first set will consist of a single instance – the $\langle rdfs:comment \rangle$ of the target concept. A classifier is trained to determine whether

a document (expressed as term vector) “belongs” to the target concept (step *1c*). The concept’s *<rdfs:comment>* is used as positive training set, while all comments of other concepts in the ontology constitute the negative training set. *ATOM* uses Naive Bayesian approach to build a classifier.

Once the classifier is trained the set of instances of the target topic \mathcal{U}_M^t is classified (step *1d*). Thus, two sets emerge: \mathcal{U}_M^{ct} and $\mathcal{U}_M^{\neg ct}$. Apparently, the set \mathcal{U}_M^t consists of a single instance – the term vector of the *<rdfs:seeAlso>* page of the target topic, therefore one of the subsets \mathcal{U}_M^{ct} and $\mathcal{U}_M^{\neg ct}$ will contain one instance, and the other will be empty.

Using the same classifier, the set of instances that do not belong to the target topic $\mathcal{U}_M^{\neg t}$ is divided into the set of instances where the target concept is present and the set of instances where it is not: $\mathcal{U}_M^{c\neg t}$ and $\mathcal{U}_M^{\neg c\neg t}$, respectively (step *1e*).

Finally steps *1c-1e* are repeated with the roles of the ontology and the topic model reversed. Thus four more sets are obtained: \mathcal{U}_O^{ct} , $\mathcal{U}_O^{\neg ct}$, $\mathcal{U}_O^{c\neg t}$ and $\mathcal{U}_O^{\neg c\neg t}$.

On the second stage of the algorithm, the eight sets of instances are used to compute the joint probability distribution of the target pair concept-topic.

Then the probabilities $P(c, t)$, $P(\neg c, t)$ and $P(c, \neg t)$ are used to calculate the measure of similarity between the concept and the topic.

These steps are repeated for every combination of every concept in the ontology and every topic in the textbook model to obtain the second similarity matrix, the one based on instances and their attribution to the elements of the two mapped models.

For the pair “*BreakStatement*”–“*The_break_and_continue_Statements*”, the *ATOM*’s instance-based mapper computes similarity = 0.1916. This is the maximum value in the similarity table row corresponding to the concept “*BreakStatement*”; this means topic

“*The_break_and_continue_Statements*” is the most similar topic to the concept “*BreakStatement*”. At the same time, in the column corresponding to the topic “*The_break_and_continue_Statements*”, it is only third maximum value. *GLUE* decides that concept “*Statement*” and “*Constants*” are more similar to “*The_break_and_continue_Statements*”, based on their instances.

The next section describes how *ATOM* merges the name-based and the instance-based similarity tables together into a single similarity matrix that incorporates not only combined name-based and instance-based information, but also the structural relations within the ontology and the topic model.

6.5 STRUCTURE-BASED SIMILARITY MERGING

Structure-based mapping techniques often build on the results of name-based and instance-based approaches in order to refine or supplement the existing mapping. The exact goal of structure-based techniques can differ from finding the closest sub-graph to spreading similarity along the structure of the ontology. Since, the exact semantics of relations within a topic-based model is not defined, *ATOM* applies a heuristic approach when processing structure. The concepts (topics) connected to the target concept (topic) form its neighborhood. If, according to the similarity table, the neighborhoods of the target pair “concept-topic” are similar, *ATOM* uses this to strengthen the mapping relation between them.

Two other tasks of this algorithm are cleaning the source similarity tables from “bad” mappings and to merging of name-based and instance-based mappings into the single similarity table.

Figure 69 formally represents the algorithm for structure-based similarity merging in ATOM. First, similarity matrices computed by name-based and instance-based mappers have to be normalized, as the original algorithms producing these matrices use very different scales to represent similarities. If a concept and a topic have exactly the same name, their similarity produced by the name-based mapper will be equal to 1.0. On the other hand, it is practically impossible to achieve a 1.0 similarity as a result of the instance-based mapper.

On the second stage, the algorithm cleans the source matrices from “bad” mappings. Similarity values that are lower than a minimal similarity threshold (0.2) are considered noise. To prevent them from propagating their influence through the models’ structure the algorithm sets them to 0.0.

The third stage of the algorithm is the core part of the structure-based mapping. One important difference of the *ATOM*’s task that had to be considered at this stage is that the structures of the topic-based models are much less formal than the ones of ontologies. The semantics of relations in a topic-based model is not clearly defined. The RDFS representation of the BigJava textbook connect all sub- and super-topics with `<rdfs:subClassOf>`. However, from a formal perspective, these relations are not always taxonomic. For example, the top-level topic “*Iteration*” has subtopics like “*The_while_Statement*” and “*The_for_Statement*” that can be considered a *subClassOf* of “*Iteration*”. It also has subtopics “*Forgetting_a_Semicolon*” and “*Loop_Invariants*”, for which *partOf* relation is more suitable. It contains a subtopic “*The_break_and_continue_Statements*”, which is *related* to iterations, but definitely is not a *subClassOf* it. Finally, it has a number of topics that have nothing to do with iteration, but have been inserted in the corresponding chapter for instructional reasons, e.g. “*Using_a_Debugger*” or “*Random_Numbers_and_Simulations*”.

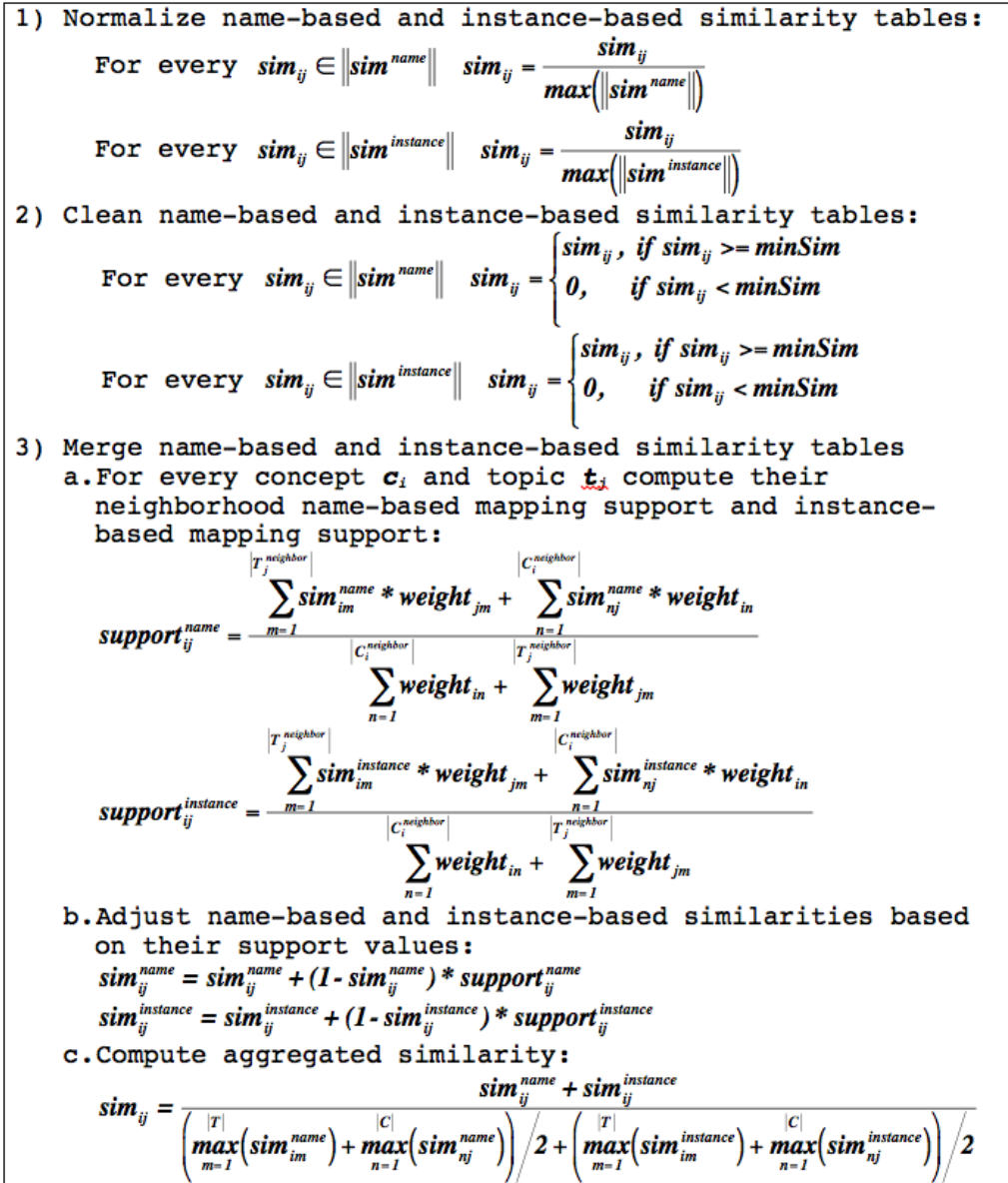


Figure 69: Structure-based similarity merging in ATOM

For these reasons, the current version of the *ATOM*'s structure-based mapper simplifies processing of different types of relations within the ontology, as well. All elements related to the target concept (and topic) with any kind of relation are considered its “neighbors”. Different categories of neighbors have different weights:

- $c_i \langle rdfs:subClassOf \rangle \mathbf{parent} \Rightarrow \mathbf{weight}^{\mathbf{parent}} = 0.5;$
- $\mathbf{child} \langle rdfs:subClassOf \rangle c_i \Rightarrow \mathbf{weight}^{\mathbf{child}} = 0.3;$
- $c_i \langle rdfs:subClassOf \rangle \mathbf{parent} \cup$
 $\mathbf{sibling} \langle rdfs:subClassOf \rangle \mathbf{parent} \cup$
 $\mathbf{sibling} \neq c_i \Rightarrow \mathbf{weight}^{\mathbf{sibling}} = 0.8;$
- $c_i \langle java:isPartOf \rangle \mathbf{whole} \Rightarrow \mathbf{weight}^{\mathbf{whole}} = 0.2;$
- $\mathbf{part} \langle java:isPartOf \rangle c_i \Rightarrow \mathbf{weight}^{\mathbf{part}} = 0.5;$
- $c_i \langle java:isRelatedTo \rangle \mathbf{relative} \Rightarrow \mathbf{weight}^{\mathbf{relative}} = 0.8.$

Once the neighborhoods of every concept and every topic is identified, the algorithm computes the *support* value as the weighted average of similarities between the target topic and all neighbors of the target concept and similarities between the current concepts and all neighbors of the target topics, using the weights specified above. The *support* is computed separately for the name-based and the instance-based similarity tables.

On the step 3b the algorithm uses the *support* values to strengthen similarities in both tables. The stronger the support is provided by the neighborhoods of a concept-topic pair, the more their similarity will grow.

On the last step (3c) the algorithm merges two similarity tables. The aggregated topic-concept similarity is computed as a sum of their name-based and instance-based similarities divided by maximum existing similarities for these topic and concept. If, in both tables, the target concept is the best possible mapping for the target topic, their aggregated similarity will be 1.0. If any of tables contains higher values either in the target concept's row or in the target topic's column, the resulting similarity will take a value between 0.0 and 1.0.

For the pair “*BreakStatement*”–“*The_break_and_continue_Statements*”, the *ATOM*’s structure-based mapper computes aggregated similarity = 0.6012. This is the maximum value for both the concept “*BreakStatement*” and the topic “*The_break_and_continue_Statements*”.

6.6 ORDER-BASED MAPPING REFINEMENT

The three algorithms described in Sections 6.3-6.5, use traditional sources of information for the area of ontology mapping. This section presents a rather new addition to the standard toolset of ontology mapping techniques. It utilizes the order of elements in the mapped models to refine the results of the previous mappers.

Usually, when two ontologies or schemas are being mapped, the concepts and attributes of these ontologies do not have a fixed order. In some cases, such an order can be mined (e.g. order of attributes in a relation database, order of fields in a web-based form), but it rarely represents any meaningful regularity. Due to the specific origin and purpose of the models mapped by *ATOM*, as well as the usage scenario of produced mapping, the elements of these models come in fixed sequences. More importantly, these sequences follow meaningful principles, and the semantics of these principles is likely to be the same across the models. These sequences are sequences of learning goals, and these principles are based on prerequisite-outcome relations between the goals.

An author of a tutorial or a textbook orders its learning material in such a way that the most basic topics are explained first. More advanced topics will require knowledge of the basic ones, and, therefore will have to follow their prerequisites. When extracting the topic-based model *OOPS* preserve order of the elementary topics within chapters and order of chapters

within tables of content, thus enabling sequencing of topics according to the order of their associated learning goals. Naturally, different tutorial can sequence some of their topics differently. E.g. iteration statements can be explained before or after conditional statement, inheritance can be explained before or after interfaces. However, for the most of the topics the mutual order has to be preserved to ensure meaningful learning of domain knowledge.

The usage scenario of *OOPS* assumes that there already exists a course or another e-Learning system that employs the central domain ontology for content and student modeling. Based on the sequence of the course material or contend elements of the main e-learning system, the concepts of the central ontology are also ordered according their learning goals. The main study of this dissertation uses QuizJET as such system. The complete set of QuizJET exercises covering a typical introductory Java course have been indexed in terms of the central Java ontology. The prerequisite-outcome identification algorithm described in Section 3.3.1.2 has been applied to automatically assign learning goals ids to every ontology concept used for exercise indexing. Section 5.5.1 and Figure 60 briefly explains how *QuizJET* exercise and the concepts indexing them were ordered into a sequence of learning goals.

Figure 70 presents *ATOM*'s algorithm for order-based mapping refinement. The algorithm takes as input the source models, two sequences of learning goals, association of learning goals and elements of the source models, and the similarity table produced by the structure-based mapper. The output of the algorithm is the final similarity table adjusted based on the cross-ordering of two models.

- 1) Define the sequence of learning goal for ontology concepts: \mathcal{G}^o as a function $f^o: \{1, 2, 3, \dots, N\}$, such that $g_n^o = f^o(n)$ and the extracted topics: \mathcal{G}^t as a function $f^t: \{1, 2, 3, \dots, M\}$, such that $g_m^t = f^o(m)$
- 2) Assign to every learning goals $g_n^o \in \mathcal{G}^o$ the corresponding set of concepts c_i ($i \in I_n$), where I_n is the set of concepts in the learning goal g_n^o ;
Assign to every learning goals $g_m^t \in \mathcal{G}^t$ the corresponding set of topics t_j ($j \in J_m$), where J_m is the number of concepts in the learning goal g_m^t ;
- 3) Build the learning goal correspondence matrix $cor[N \times M]$, where every $cor_{nm} = \sum_{i \in I_n; j \in J_m} sim_{ij}$
- 4) Normalize the learning goal correspondence matrix:
$$cor_{nm} = \frac{cor_{nm}}{\max_{\substack{n=1..N \\ m=1..M}}(cor_{nm})}$$
- 5) Apply the Hungarian algorithm to the learning goal correspondence matrix, as a result every $g_n^o \in \mathcal{G}^o$ is assigned to some $g_m^t \in \mathcal{G}^t$: $g_n^o \Leftrightarrow g_m^t$
- 6) Build the learning goal distance matrix $dist[N \times M]$, where every $dist_{nm} = \frac{abs(g_n^o - g_x^o) + abs(g_m^t - g_y^t)}{2}$, where $g_x^o \Leftrightarrow g_m^t$ and $g_n^o \Leftrightarrow g_y^t$ in the learning matrix correspondence table, as computed by the Hungarian algorithm
- 7) Refine the mapping similarity table using the learning goal distance matrix:
$$sim_{ij} = \frac{sim_{ij}}{1 + dist_{nm}}$$
, where concept c_i is assigned to the learning goal g_n^o and topic t_j is assigned to the learning goal g_m^t

Figure 70: Order-based mapping refinement

The first two stages of the algorithm separate concepts and topics into two sequences of sets, where every set contains concepts/topics corresponding to their learning goal.

Stages 3 and 4 of the algorithm build the learning goal correspondence matrix; rows of the matrix correspond to the learning goals of the central system (*QuizJET*), and columns – to the learning goals of the textbook. Every element of the matrix cor_{nm} is computed as a sum of mapping similarities between all the concepts assigned to the learning goal g_n^o and all the topics

assigned to the learning goal \mathcal{G}_m^x . If two goals combine many similar concepts and topics, the value of their corresponding element will be high; on the other hand, if the learning goals are far from each other in terms of mapping, their cell in the matrix will have a lower value. Stage 4 normalizes the learning goal correspondence matrix, so that all its elements are between 0 and 1.

On Stage 5 the Hungarian algorithm is applied again to find the best possible match between two sets of goals based on their corresponding matrix. The output of the Hungarian algorithm is used to build another matrix – the matrix of distances (Stage 6). Elements of this matrix are equal to 0, if their row and column correspond to the best-matched learning. If the two goals have not been matched by the Hungarian algorithm, their element will be computed as an average of two distances:

- the row distance, which is the difference between the row goal id and the id of the best match to the column goal, and
- the column distance, which is the difference between the column goal id and the id of the best match to the row goal.

Figure 71 illustrates the procedure of computing elements of the learning goal distance matrix. The left part of the figure shows an example of the Hungarian algorithm's output for two learning goal sequences, each consisting of four learning goals. The right part of the figure presents the distance matrix computed based on example Hungarian matching. Elements with the indexes [1,2], [2,4], [3,3] and [4,1] are equal to zero, as they correspond to the best-matched combinations of learning goals. The element [1,1] contains distance $\frac{3+1}{2} = 2$. This distance is computed as follows. The row distance of the match $\mathcal{G}_1^o - \mathcal{G}_1^x$ is the absolute difference between the id of $\mathcal{G}_1^o = 1$ and the id of \mathcal{G}_n^o that is the best match for \mathcal{G}_1^x . According to the Hungarian output, \mathcal{G}_4^o is the best match for \mathcal{G}_1^x , hence the row distance of the match $\mathcal{G}_1^o - \mathcal{G}_1^x =$

$abs(1-4)=3$. The column distance of the match $g_1^o - g_1^x$ is the absolute difference between the id of $g_1^x=1$ and the id of g_m^x that is the best match for g_1^o . Hungarian algorithm calculates that g_2^x provides the best match for g_1^o . Therefore the column distance = $abs(1-2)=1$. The aggregate distance is computed as the average of the row and the column distances: $\frac{3+1}{2} = 2$.

The rest of the matrix is populated analogously.

Hungarian matching		Distance Matrix			
Central goals	Tutorial goals	1	2	3	4
1	2	$\frac{3+1}{2}$	0	$\frac{2+1}{2}$	$\frac{1+3}{2}$
2	4	$\frac{2+3}{2}$	$\frac{1+2}{2}$	$\frac{1+1}{2}$	0
3	3	$\frac{1+2}{2}$	$\frac{2+1}{2}$	0	$\frac{1+1}{2}$
4	1	0	$\frac{3+1}{2}$	$\frac{1+2}{2}$	$\frac{2+3}{2}$

Figure 71: Example of learning goal distance matrix calculation

The last stage of the order-based algorithm adjusts the mapping similarity matrix to the distances between learning goals: every similarity value is divided by the distance between the concepts' and the topic's learning goals. The output of the order-based algorithm is the resulting similarity matrix. Elements of this matrix are used by the *OOPS* recommender as the degree of relevance between the concepts indexing the current *QuizJET* exercise and the recommended topic. To conclude the example, used through this chapter, the degree of relevance between the

concept the concept “*BreakStatement*” and the topic “*The_break_and_continue_Statements*” is 0.4251. This is the highest relevance for these concept and topic.

6.7 CONCLUSION

This chapter presented a detailed description of the *ATOM* approach to mapping ontologies and topic-based models. *ATOM* combines four different algorithms using four different dimensions of information from the mapped models. The name-based and the instance-based mappers constitute the core of *ATOM*. The name-based mapper computes similarity between concepts and topics by comparing their naming labels. The instance-based mapper reuses *GLUE* algorithm originally developed by (Doan, Madhavan et al. 2002). It takes *<rdfs:comment>*'s of concepts in the ontology and pieces of learning content associated with topics as their instances and estimates similarity between the elements of two models based on cross-similarity of term vectors of their instances.

The name-based and the instance-based similarity matrices are normalized, cleaned and merged with the help of the structure-based mapper. The concepts' and topics' structural neighborhood is used to adjust the mapping. Finally, the order-based algorithm refines the mapping using sequencing of concepts and topics according to their learning goals. The values of resulting similarity matrix is used by *OOPS* as a relevance metric between topics and concepts.

ATOM relies on several unconventional solutions in order to cope with the peculiarities of mapping a formal ontology to a topic-based model:

- the name-based of *ATOM* makes a heavy use of IR techniques to handle long, subjective and often-irregular topic names;

- the instance-based mapper of *ATOM* cannot rely on the traditional notion of an instance, which is an individual of a class; instead it uses `<rdfs:comment>`'s of concepts and content of pages titled by topics as their instances;
- the structure-based mapper of *ATOM* has to take into account undefined semantics of relations in the topic-based model;
- finally, the order-based mapper exploits a unique source of information about the elements of the two models.

It should be mentioned that the best possible mapping is not always ranked first or even second. *ATOM-OOPS* approach tries to solve this problem by leveraging the intelligence of the user. When topics are recommended, the student is presented with the list of recommendations that is very likely to include the best possible mapping, as well. The descriptive names of topics become very instrumental, at this moment, as they can provide the student with enough information to make the right choice. From another point, in most cases, even suboptimal mappings are the source of relevant learning material, as they come from the neighborhood of the optimal mapping and the user might benefit from reading them, as well.

No dedicated study has been organized for evaluating the accuracy of *ATOM*. Instead, the quality of the produced mapping can be estimated based on the results of the experiment described in the next chapter. It evaluates the developed system as a whole instead of measuring the performance of individual components. However, the results presented in section 0 that are related the most on the accuracy of *ATOM*-produced mapping, are very positive. According to several metrics, the quality of open-corpus recommendation (that directly depends on the quality of topic-to-concept mapping) is statistically similar to the quality of closed-corpus recommendation. This means, *ATOM*'s automatic mapping of automatically extricated topic-

based models of open-corpus collections into a domain ontology is capable to support high-quality open-corpus personalization.

7.0 EVALUATION OF THE ONTOLOGY-BASED OPEN-CORPUS PERSONALIZATION SERVICE

7.1 INTRODUCTION

This chapter presents the results and the procedure of the *OOPS* service evaluation. It was organized as a controlled balanced experiment comparing the developed system against two control conditions. Figure 72 summarizes the differences between the conditions and the main direction of the effects that the experiment aimed at discovering. The experimental system (version 2) provided students solving self-assessment quizzes with open-corpus recommendation of reading material. Another version of the system (version 1) had the identical interface and generated recommendations from the same pool of reading material, but used traditional closed-corpus adaptation approach based on the manual indexing of recommended pages. The last configuration of the system (version 0) did not recommend any reading material. Instead, students using this version had a hard copy of the textbook, which was the source of reading material for the first two versions.

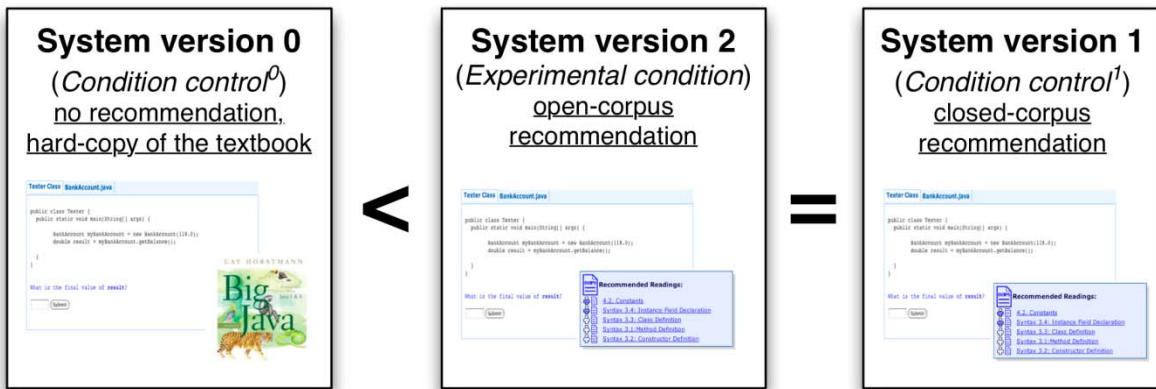


Figure 72: Three versions of the system and expected trend of effects

The goal of the experiment was to demonstrate that open-corpus personalization provided by *OOPS*:

1. is comparable in quality to the traditional closed-corpus recommendation and is capable of achieving similar effects on the main usage/learning parameters;
2. brings extra value compared to the regular text-nook and is capable to outperform the no-recommendation condition on the main usage/learning parameters.

The rest of the chapter is structured as follows. Section 6.2 describes the evaluation methodology, including the details of the experiment design, the study materials and the target user population. Section 6.3 summarizes the results of the analysis focused on the learning effect of the system. Section 6.4 reports the outcomes of the evaluation of the quality of recommendation generated by the *OOPS* service. Section 6.5 supplements two previous sections with the analysis of the subjective evaluation of the system based on the post-study questionnaire. Finally, Section 6.6 summarizes the results of the evaluation and its main conclusions.

7.2 EVALUATION METHODOLOGY

7.2.1 Three versions of the system

In order to estimate the effectiveness of the proposed approach and the added value of the developed recommender service, three different versions of the systems have been designed. The first steps of interaction with the system were the same for all versions: a student had to login, select the current session and then start the *QuizJET* by choosing one of the questions from the list (Figure 73). The questions were named in such a way that a student would recognize the main topic of the question⁴³. Once the student clicked on question link, one of the three versions of the system launched, depending on the condition associated with the current session of the student.

The *experiment*-version of the system has been described in details in Chapter 4. This version combines self-assessment questions served by *QuizJET* and adaptive recommendation of open-corpus reading material. The source for the recommended pages is the electronic version of the *BigJava* textbook ([Horstmann 2007](#)). Figure 74 demonstrates the basic interface of the *experiment*-version of the system. At every moment a student can submit an answer to a question, go back to the list of questions, or click on one of the recommended links and start reading. The details of this interface, including interaction with the recommended pages are described in Chapter 4.

⁴³ It has been arranged in order to not put the *control*⁰-condition at disadvantage. The students from the *control*⁰-condition would have had harder time finding relevant parts of the textbook, if the question topics were not specified. Yet, in real settings a student generally knows in advance the topic(s) she is going to practice.

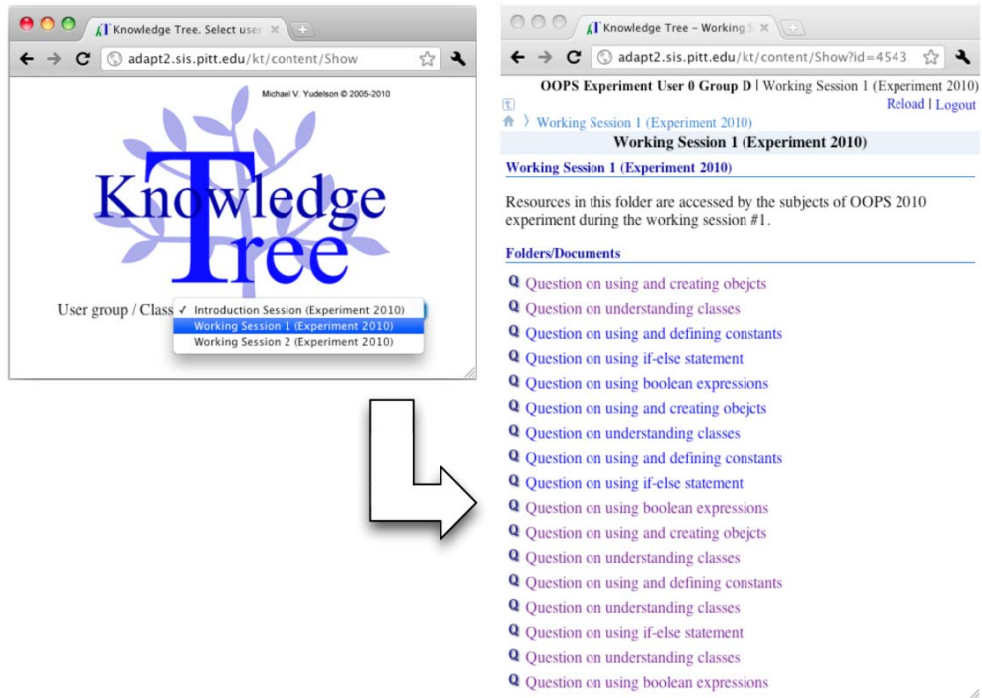


Figure 73: System interface: initial interaction (choosing a session and selecting a question)

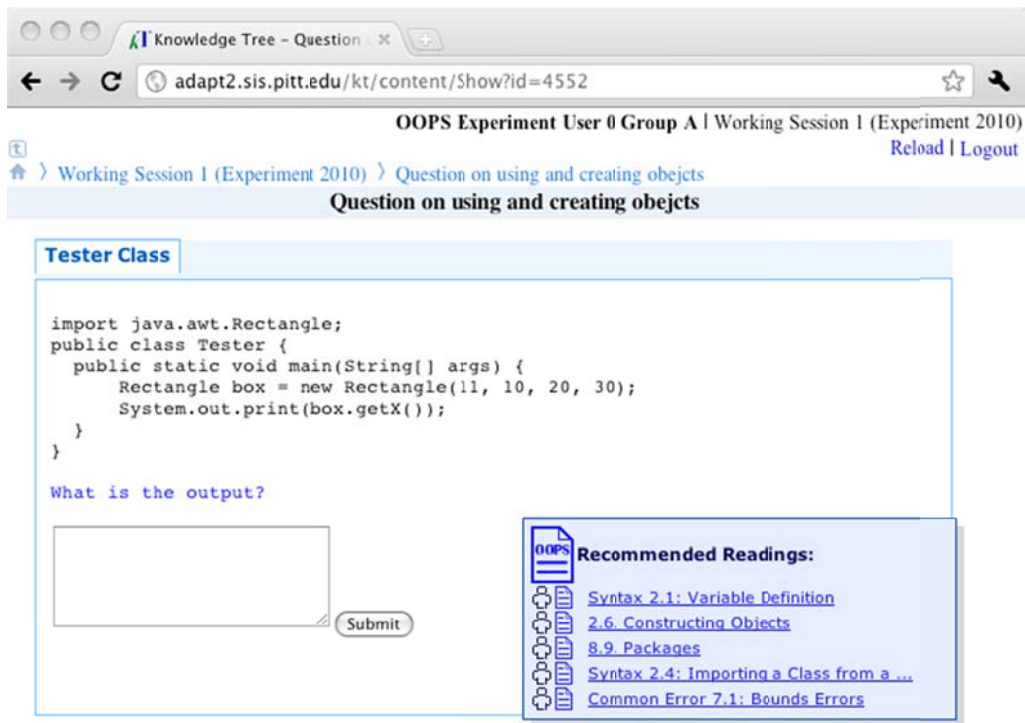
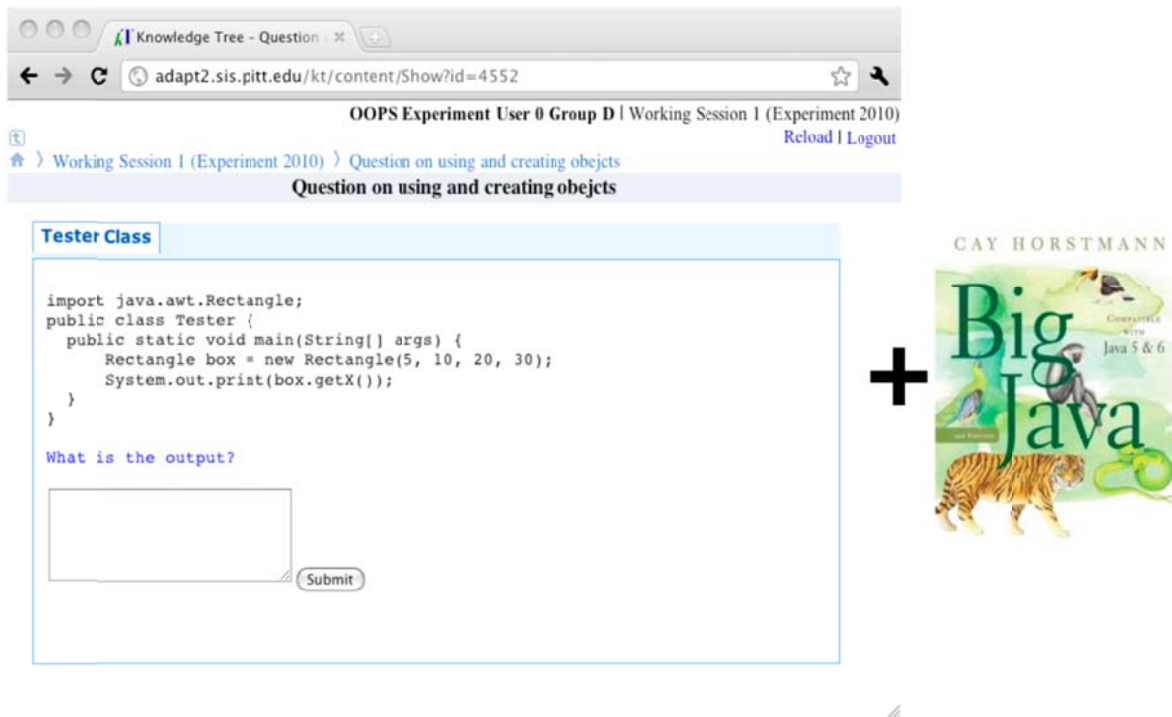


Figure 74: System interface: *experiment-condition* (*QuizJET* + *OOPS*-generated recommendations)

The *control*⁰-version of the system was implemented as a regular *QuizJET*'s interface (Figure 75), where students could analyze questions, submit answers, and navigate back to the question list. No recommendation was available for the students at this point. Instead, students have access to the hard copy of the same textbook.

The interface of the *control*¹-version of the system is identical to the interface of experiment-version: students also receive recommendations while solving *QuizJET* questions (Figure 76). However, the algorithms generating recommendations are different among these versions. The *control*¹-system produces recommendations based on the manual indexing of textbook pages with ontology concepts. In other words, this version of the system implements conventional closed-corpus adaptive recommendation. Section 5.5.3 provides a more detailed explanation.



The image shows a screenshot of a web browser window. The address bar displays 'adapt2.sis.pitt.edu/kt/content/Show?id=4552'. The page title is 'Knowledge Tree - Question'. The user is identified as 'OOPS Experiment User 0 Group D | Working Session 1 (Experiment 2010)'. The page content includes a navigation breadcrumb: 'Working Session 1 (Experiment 2010) > Question on using and creating objects'. The main heading is 'Question on using and creating objects'. Below this, there is a code editor titled 'Tester Class' containing the following Java code:

```
import java.awt.Rectangle;
public class Tester {
    public static void main(String[] args) {
        Rectangle box = new Rectangle(5, 10, 20, 30);
        System.out.print(box.getX());
    }
}
```

Below the code, the question asks: 'What is the output?'. There is a text input field and a 'Submit' button. To the right of the code editor is a plus sign (+) and the cover of the textbook 'Big Java: Concepts and Objects, Java 5 & 6' by Cay Horstmann. The cover features a tiger and a snake.

Figure 75: System interface: *control*⁰-condition (regular *QuizJET* interface + the hard copy of the textbook)

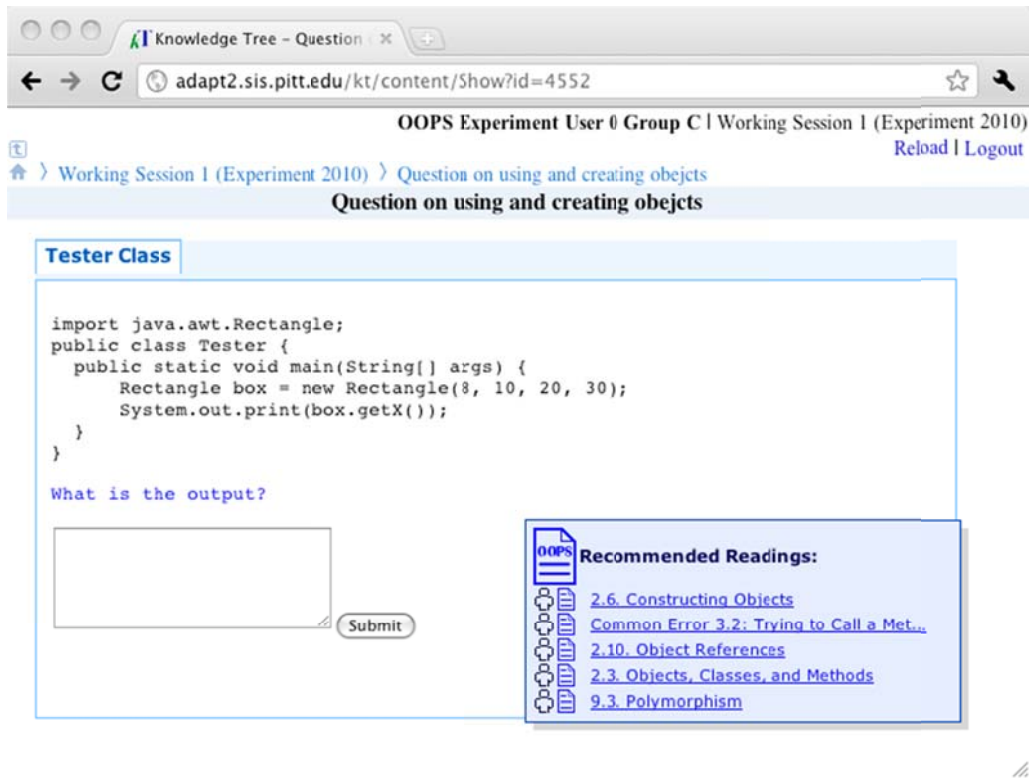


Figure 76: System interface: *control*¹-condition (*QuizJET* interface + closed-corpus recommendation)

7.2.2 Experiment Design

Broadly speaking, the subject domain of the experiment is Java programming language. However, in order for the experiment sessions to result in tractable learning in a reasonable amount of time, two sets of introductory Java topics and corresponding questions have been chosen (see Table 10). First set consists of easier topics covering the basics of handling classes and objects, as well as, the *if-else* construct. The second set includes more complex topics of loops and basic data structures (arrays and array lists). A subject solving questions of the second topic set is expected to have working knowledge of the topics from the first set. In addition, a short introductory set of topics/questions has been assembled. Questions from this set were used

during the system familiarization session; topics from this set were chosen to prevent any intersection with the topics from the two working sets.

In the course of experiment every subject explored both sets of topics. Every subject was exposed to the experiment-version of the system for one set of topics and to one of the *control*-versions for another set. The order of topic sets stayed always the same: topic set 1 was followed by topic set 2. The order of system versions varied.

Table 10: Java topics used in the experiment

Introductory Topics (6 questions)	Topic Set 1 (17 questions)	Topic Set 2 (16 questions)
Primitive Data Types (2 questions)	Using Objects (3 questions)	While loop (4 questions)
Variables (2 questions)	Defining Classes (5 questions)	Do-While loop (3 questions)
Strings (2 questions)	Constants (3 questions)	For loop (3 questions)
	Decisions (3 questions)	Arrays (4 questions)
	Boolean Expressions (3 questions)	ArrayList (2 questions)

Table 11 provides the details of the experiment design. Subjects were randomly assigned to one of the four groups: *A*, *B*, *C* or *D*. Groups *A* and *B* used experiment-version of the system while working with the topic set 1 and *control*¹- and *control*²-version respectively while working with the topic set 2. Groups *C* and *D*, on the other hand, used *control*¹- and *control*²-version of the system respectively for the simple topic set (set 1) and *experiment*-version for the complex topic set (set 2). This design allowed for between-subjects comparisons “*experiment vs. control*” and

“*experiment vs. control*”². It also enabled measuring any possible interactions between the topic complexity and the system condition.

Table 11: Design of the experiment

	Within subject	
	Session 1 / TopicSet₁	Session 2 / TopicSet₂
A (10 subjects)	Experiment (<i>open-corpus</i>)	Control1 (<i>Closed-corpus</i>)
B (10 subjects)	Experiment (<i>open-corpus</i>)	Control0 (<i>Book</i>)
C (10 subjects)	Control1 (<i>Closed-corpus</i>)	Experiment (<i>open-corpus</i>)
D (10 subjects)	Control0 (<i>Book</i>)	Experiment (<i>open-corpus</i>)

Figure 77 visualizes the detailed schedule of the experiment. The experiment started with the 15-minutes of introduction that included a brief explanation of the study, reading and signing of the consent form and the familiarization session with the system, during which subject explored introductory set of topics and questions and learned the interface of Knowledge-Tree, *QuizJET* and *OOPS*. This session was identical for all subjects and all groups (A, B, C, D). Then two 50-minutes working sessions followed. Each session contained a 10-minutes pre-test, a 30-minutes interaction with the system and a 10-minutes post-test. At the end, a 5-minutes questionnaire was administered to collect subjects’ evaluation of the system. Overall the experiment took 120 minutes.

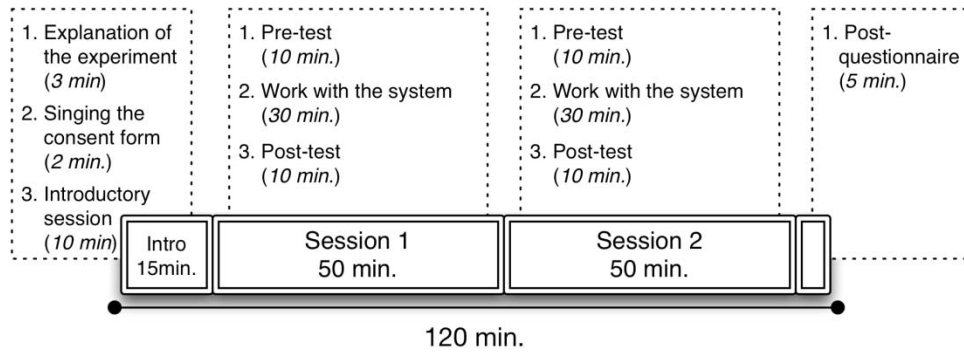


Figure 77: Experiment schedule

7.2.3 Overview of study materials

7.2.3.1 Textbook

“*Big Java*” is an introductory textbook on Java programming language and technology. It covers the basics of object-oriented programming and design with Java, overviews all main language constructs and main libraries of JDK, and introduces several advanced topics, such as multithreading, database connectivity, web programming, XML processing, etc. During the experiment, test questions were chosen only from 6 out of 24 textbook chapters (see Table 10).

7.2.3.2 *QuizJET* exercises

QuizJET questions trained students’ ability to analyze program code and predict the result of its execution. Questions for the working sessions were selected based on three criteria:

1. they had to cover target topics in a balanced way (not too many or too few per topic);
2. the number of exercises requiring subjects to specify the output of the program had to be minimal; answers to these exercises are often verbose and subjects are more likely to make a slip (out of 33 exercises in two sets combined, only three were output-based);

3. the exercises sets had to be comparable in terms of structural complexity (the number of concepts involved, the number of extra classes to analyze) to allow for balanced working sessions:
 - an average question from the Topic Set 1 was indexed with 19,53 concepts and contained 1.59 classes,
 - an average question from the Topic Set 2 was indexed with 20.01 concepts and contained 1.44 classes.

7.2.3.3 Pre- and post-tests

For every work session a pre-test and a post-test have been designed. Each of the pre-/post-tests consisted of eight questions. Unlike *QuizJET* questions, pre- and post-tests evaluated not only students' ability to analyze the code, but also their ability to write/modify a short code fragment, as well as some theoretical knowledge. Based on the results of the pre- and post-tests, knowledge gain values were computed to characterize the effectiveness of different system conditions.

7.2.3.4 Post-questionnaires

At the end of the experiment every subject filled-out a questionnaire evaluating her opinion about different aspects of the systems she used. As groups *A* and *C* were exposed to different a different set of system versions than groups *B* and *D*, their version of the questionnaire differed. Overall subjects from groups *A* and *C* had to answer 18 questions, while subjects from groups *B* and *D* – only 16.

7.2.4 Data collection

In the course of experiment, three kinds of data were collected about every subject:

- The results of pre- and post-quizzes for two work sessions (system conditions, sets of topics).
- The transactional log data of all students' interactions with the systems;
- The subjective evaluation of the system by the subjects based on the results of the questionnaire.

The log data came from several sources:

1. Students' clicks on any element of the Knowledge Tree interface recorded as:

<time; student-id; group-id; session-id; interface-element-id>

2. Students' answers to *QuizJET* questions recorded as:

<time; student-id; group-id; session-id; question-id; answer-correctness(yes/no)>

3. Recommendations generated by *OOPS* were recorded as:

<time; student-id; group-id; session-id; recommended-resource-id;

order-in-the-list-of-recommended-links(1..5)>

4. Student's actions in *OOPS* were recorded as:

<time; student-id; group-id; session-id; action-type; action-object-id; action-code>

Action codes are type-dependent. There are three possible action types:

- *<pick>* is generated, when a student clicks on a recommended list; the action code represents the order of the chosen recommendation;
- *<nav>* is generated, when a student decide to navigate to a previous or a following topic of the currently open recommendation; the action code represents the distance and the direction of the navigation target with regards to the initially recommended

page (e.g. if the subject navigates back, forward, forward, forward, her actions will be recorded as nav=-1; nav=0; nav=1; nav=2);

- *<like>* is generated, when a student decide to close the currently open recommendation by reporting whether she found it useful or not; the action code takes on of two values (1/-1) representing the subjects' opinion.

7.2.5 Subjects

The experiment took place in the spring and fall semesters of 2010. It was openly advertised in the University of Pittsburgh's School of Information Sciences and Department of Computer Sciences. Graduate and undergraduate students were equally eligible for the experiment. The main requirement to the subjects was a low level of programming knowledge.

Figure 78 presents the screening protocol used to decide if a subject can participate in the study. If a subject did not have any programming experience, she was not eligible, as she would hardly be able to demonstrate any progress in two hours training. If a subject had had formal Java education and got a good grade or she mastered Java on her own, she did not satisfy the pre-condition either, since the instructional material would have been too easy for her. The target group for the study was students who knew programming, but did not have previous knowledge of Java, or knew Java, but not well. Overall, forty subjects took place in the experiment, randomly assigned to the groups *A*, *B*, *C* and *D*. Subjects were reimbursed for their time.

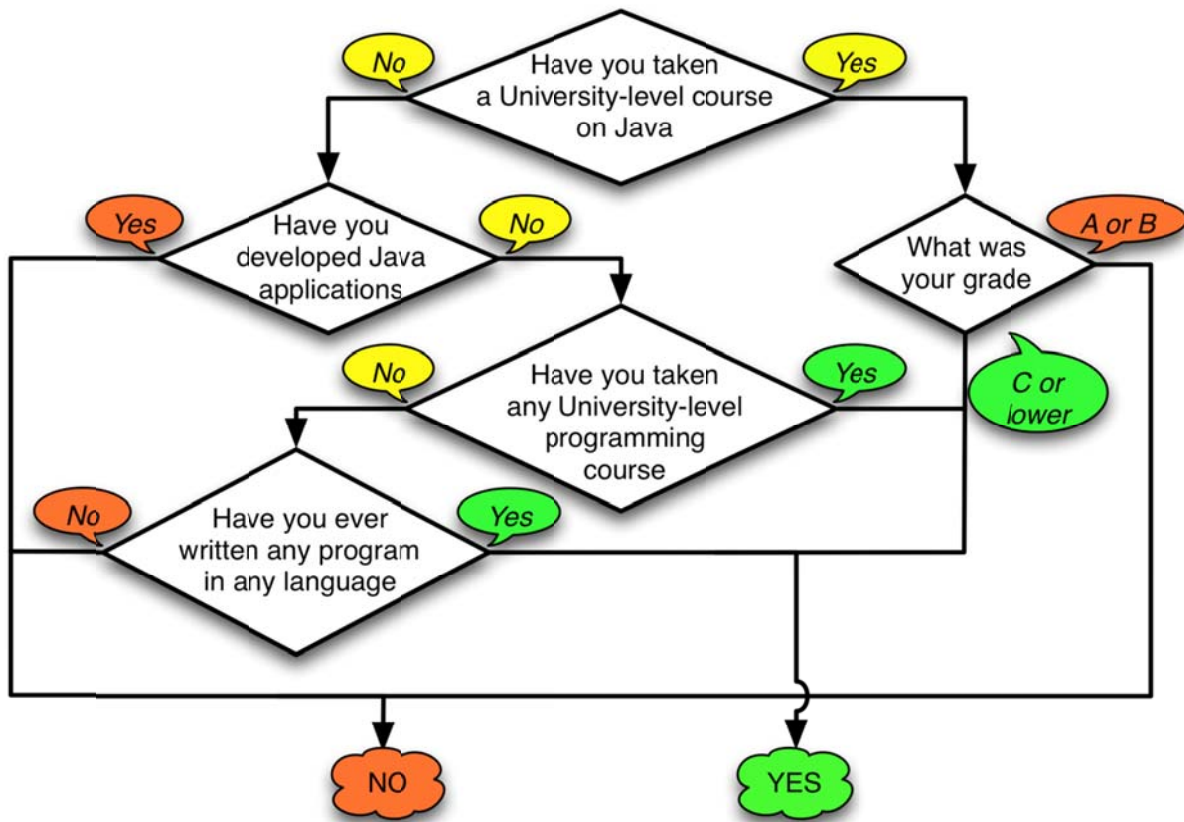


Figure 78: Subject screening protocol for the experiment

7.3 LEARNING EFFECTS

The main purpose of *OOPS* is to help student learn better. The amount of learning occurred during two treatment sessions was computed based on the results students showed on pre-tests ($Score_{pre-test}$) and post-tests ($Score_{post-test}$). The difference between these results is known as knowledge gain. Two equations have been used to compute knowledge gain. Equation 10 computes the adjusted knowledge gain ($KnowledgeGain_{adjusted}$). It is a modification of Equation 1 introduced in Section 3.4.3.2. The motivation for adjusting this equation was the situations when

the difference between the $Score_{post-test}$ and $Score_{pre-test}$ comes up negative. Such situations were not indicating that the level of knowledge dropped during the treatment, but rather that the subject slipped on a post-test, or that the test questions were not calibrated well enough.

Equation 10: Adjusted knowledge gain

$$KnowledgeGain_{adjusted} = \max(0, Score_{post-test} - Score_{pre-test})$$

Equation 11 introduces a new way to compute normalized knowledge gain ($KnowledgeGain_{adjusted}$). It modifies Equation 2 from Section 3.4.3.2 by using $KnowledgeGain_{adjusted}$ in the numerator.

Equation 11: Normalized knowledge gain

$$KnowledgeGain_{normalized} = \frac{\max(0, Score_{post-test} - Score_{pre-test})}{Score_{max} - Score_{pre-test}}$$

Table 12 summaries the mean and standard deviation values for pre- and post-tests, as well as both knowledge gain metrics for Session 1.

Table 12: Diagnostic tests and knowledge gain results (session 1)

Group	$Score_{pre-test}$	$Score_{post-test}$	$K-Gain_{adjusted}$	$K-Gain_{normalized}$
A	M=2.93 (SD=1.47)	M=5.19 (SD=1.83)	M=2.26 (SD=1.89)	M=0.47; SD=0.35
B	M=3.65 (SD=1.79)	M=5.97 (SD=1.68)	M=2.32 (SD=1.67)	M=0.55; SD=0.34
C	M=3.48 (SD=1.50)	M=5.62 (SD=1.52)	M=2.15 (SD=0.83)	M=0.50; SD=0.18
D	M=3.20 (SD=1.65)	M=5.00 (SD=2.19)	M=1.90 (SD=1.21)	M=0.45; SD=0.31

Table 13 summarizes equivalent data for Session 2.

Table 13: Diagnostic tests and knowledge gain results (session 2)

Group	<i>Score_{pre-test}</i>	<i>Score_{post-test}</i>	<i>K-Gain_{adjusted}</i>	<i>K-Gain_{normalized}</i>
A	M=2.30 (SD=2.67)	M=3.40 (SD=2.37)	M=1.30 (SD=1.57)	M=0.21 (SD=0.24)
B	M=4.30 (SD=2.00)	M=4.30 (SD=1.89)	M=0.60 (SD=0.97)	M=0.16 (SD=0.26)
C	M=3.60 (SD=2.12)	M=4.80 (SD=2.10)	M=1.50 (SD=0.97)	M=0.35 (SD=0.22)
D	M=3.30 (SD=2.21)	M=4.90 (SD=2.56)	M=1.60 (SD=1.51)	M=0.43 (SD=0.35)

Next four subsections describe the observed effects of the Experimental system on different aspects of students' learning.

7.3.1 General learning effect

Hypothesis 1.1:

The post-tests scores are significantly greater than the pre-test scores across all sessions and conditions.

In order to verify that work with the system actually leads to learning the material, pair-wise comparisons of scores on the pre-test and the post-test have been made (per group, per session). Table 14 summarizes the test results of eight paired samples t-tests for groups *A*, *B*, *C* and *D* and two working sessions. Significant learning has been registered for all groups and conditions during Session 1: across all conditions the results on the post-test are significantly higher than the results on the pre-test. For Session 2, the experimental condition (light-grey cells in Table 14)

and the closed-corpus condition (dark-grey cells in Table 14) resulted in significant (or bordering on significance) learning. At the same time, the Book-condition (white cells in Table 14) led to no learning.

Table 14: General learning effect statistics ($Score_{post-test}$ VS. $Score_{pre-test}$)

Group	Session 1		Session 2	
	t(9)	p-value	t(9)	p-value
A	3.787	0.004	1.941	0.084
B	4.409	0.002	0.0	1.0
C	8.213	<0.001	2.250	0.051
D	4.077	0.03	3.361	0.008

The main difference between Session 1 and Session 2 is material complexity. Session 1 covers general usage of objects and classes, as well as the conditional statements. Session 2 uses the knowledge introduced during Session 1 and extends it with loops, arrays and *ArrayLists*. The analysis of the general learning effect presented in this section has suggested that the recommendation of supplementary reading material can have a positive influence on learning the complex learning material. The next subsection directly investigates the effect of the system on learning material of different complexity.

7.3.2 Effect on learning complex material

Hypothesis 2.1:

During Session 1 (easy learning material), the knowledge gain for the Experimental condition (Open-corpus) is significantly greater than the knowledge gain for the Control⁰ condition (Book).

Hypothesis 2.2:

During Session 1 (easy learning material), there is no significant difference in knowledge gain between the Experimental (Open-corpus) and the Control¹ (Closed-corpus) conditions.

Hypothesis 2.3:

During Session 2 (complex learning material), the knowledge gain for the Experimental condition (Open-corpus) is significantly greater than the knowledge gain for the Control⁰ condition (Book).

Hypothesis 2.4:

During Session 2 (complex learning material), there is no difference in knowledge gain between the Experimental condition (Open-corpus) and the Control¹ condition (Closed-corpus).

Following the target trend of expected effects (Figure 72) the hypotheses were formulated with an expectation that the *Experimental* condition would outperform the *Control⁰* condition and would not be different from the *Control¹* condition. However, during Session 1 (easy learning material), none of the comparisons resulted in significant difference between $KnowledgeGain_{adjusted}$ and $KnowledgeGain_{normalized}$. Four tests on the knowledge gain scores

obtained for Session 1 confirm that there is no significant effect of the Experimental system on the effectiveness of learning easy material compared to any of the Control systems:

- there is no significant difference in the *KnowledgeGain_{adjusted}* scores during Session 1 (easy learning material) for Experimental (*Open-corpus*) condition (M=2.29; SD=1.73) and Control¹ (*Closed-corpus*) condition (M=2.15; SD=0.83): $t(28) = 0.309$; $p = 0.760$;
- there is no significant difference in the *KnowledgeGain_{normalized}* scores during Session 1 (easy learning material) for Experimental (*Open-corpus*) condition (M=0.51; SD=0.33) and Control¹ (*Closed-corpus*) condition (M=0.50; SD=0.18): $t(28) = 0.101$; $p = 0.921$;
- there is no significant difference in the *KnowledgeGain_{adjusted}* scores during Session 1 (easy learning material) for Experimental (*Open-corpus*) condition (M=2.29; SD=1.73) and Control⁰ (*Book*) condition (M=1.90; SD=1.21): $t(28) = 0.642$; $p = 0.526$;
- there is no significant difference in the *KnowledgeGain_{normalized}* scores during Session 1 (easy learning material) for Experimental (*Open-corpus*) condition (M=0.51; SD=0.33) and Control⁰ (*Book*) condition (M=0.45; SD=0.31): $t(28) = 0.477$; $p = 0.637$;

However, once the learning material became more complex (Session 2), the experimental system significantly outperformed the book:

- during Session 2, *KnowledgeGain_{adjusted}* scores (complex learning material) for Experimental (*Open-corpus*) condition (M=1.55; SD=1.23) are significantly higher than *KnowledgeGain_{adjusted}* scores for Control⁰ (*Book*) condition (M=0.60; SD=0.97): $t(28) = 2.124$; $p = 0.043$;
- during Session 2, *KnowledgeGain_{normalized}* scores (complex learning material) for Experimental (*Open-corpus*) condition (M=0.39; SD=0.29) are significantly higher than

KnowledgeGain_{normalized} scores for Control⁰ (*Book*) condition (M=0.16; SD=0.26): $t(28) = 2.107$; $p = 0.044$.

At the same time, no differences were observed between the closed-corpus and the open-corpus system when students were learning complex material:

- there is no significant difference in the *KnowledgeGain_{adjusted}* scores during Session 2 (complex learning material) for Experimental (*Open-corpus*) condition (M=1.55; SD=1.23) and Control¹ (*Closed-corpus*) condition (M=1.30; SD=1.57): $t(28) = 0.478$; $p = 0.636$;
- there is no significant difference in the *KnowledgeGain_{normalized}* scores during Session 2 (complex learning material) for Experimental (*Open-corpus*) condition (M=0.39; SD=0.29) and Control¹ (*Closed-corpus*) condition (M=0.21; SD=0.24): $t(28) = 1.634$; $p = 0.114$;

This is an important effect with a reasonable explanation. When learning easy material, students need less support from the system. Some of them do not even require extra reading – just by practicing with the self-assessment exercises of *QuizJET* they could improve their understanding of the domain. And if they need extra reading, it is easier for them to browse through the book pages and find a relevant chapter, as the time and the cognitive effort required to make a decision about the relevance of the page are low. On the other hand, when the material becomes complex, students can benefit from the recommendations relevant to the scope of the material they are learning and the exercises they are solving. They can also benefit from the adaptive annotations guiding them to the most important piece of reading material. Thus personalized learning support results in better learning only when the support is needed. The comparison of open-corpus and closed-corpus conditions show that they are equally effective in both sessions, which means the

quality of open-corpus personalization produced by *OOPS* is similar to the quality of traditional closed-corpus personalization.

7.3.3 Effect on weaker students

Hypothesis 3.1:

For weaker students, the knowledge gain for the Experimental condition (Open-corpus) is significantly greater than the knowledge gain for the Control⁰ condition (Book).

This hypothesis is related to the analysis presented in the previous session, but instead of harder learning material, it focuses on weaker students. It tries to give another answer to the question “*when is personalization needed?*”, namely: “*when the student does not know much*”. Figure 79 presents two scatter plots showing the correlation between $KnowledgeGain_{adjusted}$ and $Score_{post-test}$ for Experimental and Control⁰ (*Book*) conditions. The left plot visualizes Session 1 data, and the right plot – data from Session 2. In both sessions, one can see a clear negative correlation for the Experimental system data: students with lower pre-test scores gain more knowledge using the Experimental system than the student with higher pre-test score. There is no such correlation for the *Book*-condition. These plots suggest that lower-achieving students might benefit from the Open-corpus personalization more than from the Book. The comparison of Open-corpus and Closed-corpus systems is not presented, as no intuition or evidence has been found to support the existence of a significant difference between them.

To answer the question, whether the Experimental system helps weaker students more than the Book, only knowledge gains of students who scored badly on the pre-test should be compared between these conditions.

Table 15 summarizes the statistics obtained after filtering out students who scored 3 or more out of 8 on a pre-test.

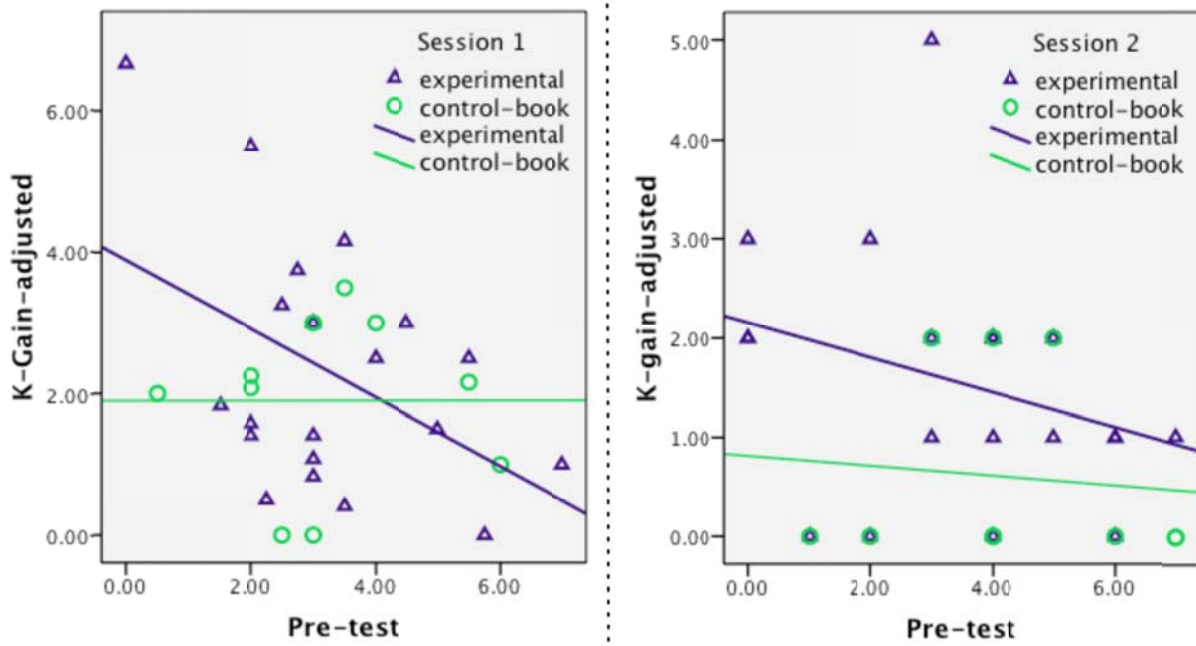


Figure 79: Knowledge gain vs. pre-test scores: Left - Session 1; Right – Session 2

Table 15: Weaker students statistics

	Session 1		Session 2	
	<i>Open-corpus</i>	<i>Book</i>	<i>Open-corpus</i>	<i>Book</i>
<i>Num. of Subjects</i>	8	4	6	2
<i>Score_{pre-test}</i>	M=1.86 (SD=0.85)	M=1.75 (SD=0.87)	M=0.83 (SD=0.98)	M=1.50 (SD=0.71)
<i>Score_{post-test}</i>	M=4.94 (SD=1.86)	M=3.23 (SD=1.10)	M=2.50 (SD=1.38)	M=1.50 (SD=0.71)
<i>K-Gain_{adjusted}</i>	M=3.06 (SD=2.15)	M=1.58 (SD=1.06)	M=1.67 (SD=1.37)	M=0.00 (SD=0.00)
<i>K-Gain_{normalized}</i>	M=0.49 (SD=0.31)	M=0.25 (SD=0.17)	M=0.23 (SD=1.20)	M=0.00 (SD=0.00)
<i>t-test for K- Gain_{adjusted}</i>	t(10)=1.276; p=0.231		t(6)=1.637; p=0.153	
<i>t-test for K- Gain_{normalized}</i>	t(10)=1.714; p=0.118		t(6)=1.535; p=0.176	

Unfortunately, the size of the experiment did not allow for sufficient numbers of subjects in the weaker students subcategories. Results of all four tests were not significant. Yet, the difference between the means show the traces of the potential effect.

7.3.4 Effect on learning conceptual material

Hypothesis 4.1:

For conceptual learning material, the knowledge gain for Experimental condition (Open-corpus) is significantly greater than the knowledge gain for Control⁰ condition (Book).

The personalization implemented by *OOPS* is aimed to achieve two instructional goals:

- Support students solving self-assessment exercises by bringing them the most relevant reading material;
- Balance students' learning by giving them the opportunity to read instructional texts in addition to practicing.

The second of these goals means that *OOPS* should contribute to improving not only the skills necessary for solving exercises, but also to the theoretical knowledge of important concepts and fact in the domain. Hypothesis 4.1 anticipates that *OOPS* should outperform the Book, on the gain of conceptual knowledge. In order to measure gain of theoretical knowledge, only conceptual questions from pre-tests and post-test should be taken into account. The pre-test and post-test of Session 1 have five conceptual and three computational questions. The pre-test and post-test of Session 2 have one conceptual question, four computational questions, and three theoretical/computational ones. The last three questions expect a number for an answer, but mostly require not the program analysis skills, but general understanding of a concept. These questions were included as 1/2 contributing to the computation of gain in conceptual knowledge. Table 16 summarizes the descriptive statistics and the independent samples t-tests comparing gain on theoretical knowledge for *Experimental* and *Control*⁰ conditions.

Hypothesis 4.1 is partially confirmed. There is a close to significant effect of Open-corpus recommendation on the gain in conceptual knowledge for both gain metrics in Session 1, and *KnowledgeGain_{normalized}* in Session 2. There is a significant effect registered for *KnowledgeGain_{adjusted}* in Session 2.

Table 16: Statistics on learning conceptual material

	Session 1		Session 2	
	<i>Open-corpus</i>	<i>Book</i>	<i>Open-corpus</i>	<i>Book</i>
<i>K-Gain_{adjusted}</i>	M=2.61 (SD=1.75)	M=1.75 (SD=1.15)	M=0.73 (SD=0.47)	M=0.30 (SD=0.42)
<i>K-Gain_{normalized}</i>	M=0.66 (SD=0.28)	M=0.45 (SD=0.29)	M=0.50 (SD=0.38)	M=0.21 (SD=0.33)
<i>t-test for K-Gain_{adjusted}</i>	t(28)=1.762; p=0.089		t(28)=2.403; p=0.023	
<i>t-test for K-Gain_{normalized}</i>	t(28)=1.880; p=0.071		t(6)=2.035; p=0.051	

7.4 USING THE SYSTEM

While working with the systems, students produce a large volume of log data. Section 7.2.4 gives an overview of various kinds of information that has been collected during the experiment.

Two important sources of data:

- interactions with *OOPS* recommendations (overall, 3618 such interactions have been registered during two working sections of the experiment, not including the transactions generated by *OOPS* automatically whenever a new set of recommendations is presented to a student),
- answers to *QuizJET* exercises (total number of answers submitted by students from all conditions and sections is 1656).

This section analyses the data collected from both systems, and computes several statistics to compare the usage of open-corpus and closed-corpus setups.

7.4.1 Quality of recommendation

Table 17 summarizes the basic statistics of using *OOPS* computed from the main three types of actions a student can perform (accessing a recommendation page, browsing to another page, giving feedback on a recommendation).

OOPS gives students a choice when it comes to accepting recommended reading material. The recommendation widget is designed in such a way that it does not obstruct work with *QuizJET* exercises. If students believe they can answer the current exercises without *OOPS* assistance they are free to ignore it. The first column contains average numbers of recommendations accessed by students (i.e., how many times students clicked on one of the recommended topic links).

When presenting a reading page, *OOPS* provides links to the previous and the next topic in the book (Area “*B*” of Figure 57). This way, it tries to remedy possible recommendation inaccuracies (when a recommended page is in the neighborhood of relevant learning material) and facilitate discovery of the exact page that helps answering the exercise. The second column presents the numbers of browsed pages. Students have showed two very different navigation patterns: about 2/3 of them mostly ignored “*next topic*” and “*previous topic*” links, while the other third used it much more often⁴⁴.

The design of *OOPS* implements dedicated interface elements allowing students to give immediate feedback on the usefulness of recommended material (Area “*C*” of Figure 57). When closing the current recommendation window, they had to click one of two buttons: “*This was*

⁴⁴ This is the reason for very high SD values in this column

USELESS” and “*This was USEFUL*”. The third column shows average percentage of recommendations reported as “*USEFUL*”.

Columns 4, 5 and 6 summarize corresponding data for Session 2.

Table 17: Basic recommendation statistics

Group	Session 1			Session 2		
	Pages accessed	Pages browsed	Ratio of “ <i>useful</i> ” recommendations	Pages accessed	Pages browsed	Ratio of “ <i>useful</i> ” recommendations
A	M = 25.89 (SD=18.50)	M = 9.56 (SD=15.63)	M = 68.88% (SD=17.44%)	M = 26.78 (SD=19.96)	M = 3.89 (SD=10.93)	M = 49.92% (SD=22.89%)
B	M = 28.90 (SD=20.74)	M = 7.80 (SD=11.92)	M = 43.74% (SD=25.93%)			
C	M = 27.20 (SD=17.44)	M = 8.60 (SD=8.58)	M = 58.11% (SD=22.01%)	M = 30.80 (SD=24.93)	M = 5.60 (SD=8.57)	M = 65.97% (SD=31.55%)
D				M = 22.00 (SD=14.75)	M = 13.10 (SD=13.09)	M = 78.55% (SD=16.12%)

7.4.1.1 Recommendation acceptance rate

Hypothesis 5.1:

There is no significant difference in the Recommendation Acceptance Rate between the Experimental condition (Open-corpus) and the Control¹ condition (Closed-corpus).

As mentioned before, *OOPS* recommends reading material, but does not force students to use it. When making a decision about clicking on a recommended link, they, first, consider whether or not they need assistance, and then decide which topic to access, based on its label and the adaptive icon annotating it. The average number of accessed pages may be not the best parameter to compare how often students accept a recommendation, as it does not take into account how

many times they have been given a recommendation, i.e. how many *QuizJET* exercises they solved.

Equation 12 provides the formula for recommendation acceptance rate. Columns 1 and 4 of Table 18 contain its values averaged per group and per session. The results of independent samples t-tests show that:

- there is no significant difference in the Recommendation Acceptance Rate during Session 1 between Experimental (*Open-corpus*) condition (M=79.21; SD=32.99) and Control¹ (*Closed-corpus*) condition (M=89.19; SD=22.86): $t(28) = 0.856$; $p = 0.339$;
- there is no significant difference in the *Recommendation Acceptance Rate* during Session 2 between Experimental (*Open-corpus*) condition (M=79.04; SD=32.93) and Control¹ (*Closed-corpus*) condition (M=78.59; SD=37.83): $t(28) = 0.033$; $p = 0.974$.

Equation 12: Recommendation acceptance rate

$$\text{Recommendation Acceptance Rate} = \frac{\text{Number of exercises solved after accepting a recommendation}}{\text{Total number of exercises solved}}$$

Table 18: Extra recommendation statistics

Group	Session 1			Session 2		
	Rec. Acceptance Rate	Average Accepted Rec. Rank	Browsing Occurrence Rate	Rec. Acceptance Rate	Average Accepted Rec. Rank	Browsing Occurrence Rate
A	M = 87.90% (SD=28.11%)	M = 2.18 (SD=0.79)	M = 22.50% (SD=40.93%)	M = 78.59% (SD=37.83%)	M = 2.35 (SD=0.63)	M = 10.51% (SD=21.31%)
B	M = 70.53% (SD=36.61%)	M = 2.64 (SD=0.44)	M = 10.67% (SD=13.96%)			
C	M = 89.19% (SD=22.86%)	M = 2.45 (SD=0.49)	M = 14.38% (SD=10.84%)	M = 79.67% (SD=35.42%)	M = 2.65 (SD=0.53)	M = 10.33% (SD=15.49%)
D				M = 78.41% (SD=32.15%)	M = 2.61 (SD=0.89)	M = 19.39% (SD=16.26%)

7.4.1.2 Average accepted recommendation rank

Hypothesis 5.2:

There is no significant difference in the Average Accepted Recommendation Rank between the Experimental condition (Open-corpus) and the Control¹ condition (Closed-corpus).

OOPS present recommendations to students in lists of five. The order of a link in a list is determined by the relevance of the page to the current *QuizJET* exercise. For the open-corpus system, the relevance metric is computed based on the mapping of the topic to one or several concepts indexing the exercise. For the closed-corpus system, this metric is computed based on matching between the concept-based model of the exercise and the concept-based model of the page. In both cases, the higher the link is in the recommended list, the more relevant it is to the exercise, as predicted to the system. One way to evaluate the accuracy of these predictions is to compute the average rank of recommended pages accepted by students⁴⁵. Columns 2 and 5 of Table 18 contain the values of *Average Accepted Recommendation Rank* computed for all groups and sessions. The rank takes values between 1 and 5, where 1 means the recommended link has been on the top of the list, and 5 means it has been at the bottom. The results of independent samples t-tests show that:

- there is no significant difference in the *Average Accepted Recommendation Rank* during Session 1 between Experimental (*Open-corpus*) condition (M=2.41; SD=0.67) and Control¹ (*Closed-corpus*) condition (M=2.45; SD=0.49): $t(28) = 0.856$; $p = 0.339$;

⁴⁵ This parameter will also depend on the presence of adaptive navigation. Nevertheless, it does not weaken the final conclusion, as both: closed-corpus and open-corpus systems implemented the same set of personalization techniques, and the main goal of this comparison is to show that the open-corpus system is capable of achieving similar quality of personalization, as the closed-corpus one.

- there is no significant difference in the *Average Accepted Recommendation Rank* during Session 2 between Experimental (*Open-corpus*) condition (M=2.63; SD=0.71) and Control¹ (*Closed-corpus*) condition (M=2.35; SD=0.63): $t(28) = 0.157$; $p = 0.876$.

7.4.1.3 Browsing occurrence rate

Hypothesis 5.3:

There is no significant difference in the Browsing Occurrence Rate between the Experimental condition (Open-corpus) and the Control¹ condition (Closed-corpus).

Students use links to the previous and the next topics to refine the accessed recommendation and explore the neighboring learning material. A separate study would be required to determine the reasons for such browsing and the instructional effect of it (if any). From one point of view, when a student chooses to browse away from the directly recommended page, it is an evidence of an imprecise recommendation. From another point, exploring the neighborhood of a relevant topic can benefit to students' learning. In both cases, comparison of how often student engage in browsing using the open-corpus system and the closed-corpus one, can help to evaluate the difference in their effectiveness.

Equation 13 provides a formula for computing the *Browsing Occurrence Rate*. Columns 3 and 6 of Table 18 contain the vales of Browsing Occurrence Rate for all groups and sessions. The results of independent samples t-tests show that:

- there is no significant difference in the *Browsing Occurrence Rate* during Session 1 between Experimental (*Open-corpus*) condition (M=16.59; SD=30.38) and Control¹ (*Closed-corpus*) condition (M=14.38; SD=10.34): $t(28) = 0.221$; $p = 0.827$;

- there is no significant difference in the *Browsing Occurrence Rate* during Session 2 between Experimental (*Open-corpus*) condition (M=14.86; SD=16.14) and Control¹ (*Closed-corpus*) condition (M=10.51; SD=21.32): $t(28) = 0.626$; $p = 0.537$.

Equation 13: Browsing occurrence rate

$$\text{Browsing Occurrence Rate} = \frac{\text{Number of accepted recommendations with browsing}}{\text{Total number of accepted recommendations}}$$

7.4.1.4 Perceived value of recommendation

Hypothesis 5.4:

There is no significant difference in the Perceived Value of Recommendation between the Experimental condition (Open-corpus) and the Control¹ condition (Closed-corpus).

Average user's assessment of the recommendation is the most direct indicator of its quality. As mentioned before, *OOPS* allows students to report their assessments of recommendation value (or usefulness) in an easy and unobtrusive way. The *Perceived Value of Recommendation* parameter can be computed as the average ratio of recommendations reported as "USEFUL". Columns 3 and 6 of Table 17 present *Perceived Value of Recommendation* computed for all groups and sessions. The results of independent samples t-tests show that:

- there is no significant difference in the *Perceived Value of Recommendation* during Session 1 between Experimental (*Open-corpus*) condition (M=57.09; SD=25.42) and Control¹ (*Closed-corpus*) condition (M=58.11; SD=22.01): $t(28) = 0.107$; $p = 0.915$;

- during Session 2, *Perceived Value of Recommendation* for Experimental (*Open-corpus*) condition (M=72.26; SD=25.23) was significantly higher than for Control¹ (*Closed-corpus*) condition (M=49.92; SD=22.89): $t(28) = 2.266$; $p = 0.032$.

The second result surprisingly shows that, during Session 2, open-corpus personalization outperforms closed-corpus one. A possible explanation for this effect is a higher amount of browsing that open-corpus groups (*C* and *D*) have done (see column 5 of Table 17). This might have allowed open-corpus students to find useful pages more often.

7.4.2 Effect on self-assessment exercises

Table 19 summarizes the most important descriptive statistics of students' work with self-assessment exercises. Columns 1 and 4 present mean and standard deviation values of the overall numbers of attempts to solve *QuizJET* exercises submitted by students of all four groups within Session 1 and Session 2 correspondingly. Columns 2 and 5 contain similar data for the average success of students' answers. Finally, columns 3 and 6 summarize the exercise material coverage, which is the percentage of all questions answered correctly at least once. If a student have never attempted to answer a question or never submitted a correct answer to it, this question is considered as not covered by the student.

Table 19: QuizJET statistics

Group	Session 1			Session 2		
	Attempts	Success	Coverage	Attempts	Success	Coverage
A	M = 16.20 (SD=7.04)	M = 58.98% (SD=15.06%)	M = 41.76% (SD=19.50%)	M = 16.70 (SD=8.69)	M = 47.54% (SD=30.35%)	M = 37.06% (SD=23.70%)
B	M = 25.10 (SD=10.83)	M = 57.70% (SD=13.81%)	M = 71.76% (SD=25.01%)	M = 28.00 (SD=26.09)	M = 54.99% (SD=25.72%)	M = 55.29% (SD=25.45%)
C	M = 15.20 (SD=8.59)	M = 54.68% (SD=13.02%)	M = 42.94% (SD=24.18%)	M = 12.60 (SD=8.67)	M = 48.85% (SD=33.85%)	M = 39.41% (SD=28.96%)
D	M = 29.90 (SD=10.96)	M = 60.67% (SD=16.64%)	M = 87.65% (SD=9.78%)	M = 21.90 (SD=5.28)	M = 46.42% (SD=17.70%)	M = 55.29% (SD=17.58%)

One can immediately observe that *Control*⁰ condition (*Book*) dominates other conditions in *Attempts* and *Coverage*⁴⁶. Naturally, Success rate is also higher for this condition (although the difference is not significant), as *Control*⁰ students practice with exercise more than students from other conditions. It means that when students have a textbook as a reading material they tend to use it less and spend more time solving exercises than reading. A connection can be drawn from this observation to the results of learning effect analysis (Section 7.3). Such learning leads to lower knowledge gain on complex topics, and lower knowledge gain on conceptual learning material; it is also less effective for weaker students.

No significant difference on any of the *QuizJET* usage parameters has been observed while comparing the *Experimental (Open-corpus)* and *Control*¹ (*Closed-corpus*) conditions.

⁴⁶ For Session 1, this difference is statistically significant; for Session 2, it is not.

7.5 SUBJECTIVE EVALUATION

To elicit students' opinion about different aspects of the systems a post-questionnaire was administered. Two versions of the questionnaire were developed: one for groups *A* and *C* that worked with the open-corpus and the closed-corpus versions of the system, and one for groups *B* and *D* that worked with the open-corpus and the textbook-based version of the system. Appendix F contains the full text of both versions. Table 20 presents a more consolidated view of the questionnaire, including the questions and students' answers averaged by group. The questionnaire used five-level Likert scale as a basis for students' input. To compute the average the following Likert level were converted to numbers: 1 – *Strongly Disagree*; 2 – *Disagree*; 3 – *No Strong Opinion*; 4 – *Agree*; 5 – *Strongly Agree*.

Subjects are more likely to agree with the question statements. To avoid the influence of questions favoring *OOPS*, some questions were formulated in a more negative format, critiquing *OOPS* features.

Several questions were introduced to verify the reliability of students' answers. Questions 5 and 7 ask students whether they had to browse in order to find a helpful page; Question 5 claims that browsing was necessary, while Question 7 rebuts the need for browsing. Questions 9 and 10 elicit students' opinion about the usefulness of adaptive annotations: Question 9 in a positive way and Question 10 – in negative. The Cronbach's Alpha test showed that there is an acceptable level of internal consistency between the answers student have given to these questions. For the inverse version of Question 5 and the Question 7, in Session 1, Cronbach's Alpha = 0.717; in Session 2, Cronbach's Alpha = 0.669. For Question 9 and the inverse version of Question 10, Cronbach's Alpha = 0.717.

Table 20: Post-questionnaire results

#	Question		A	B	C	D
1	I like the interface of the <i>OOPS</i> service.		3.9	3.5	3.5	4.0
2	I tried to use the help of the <i>OOPS</i> service.		4.3	4.1	4.0	4.3
3	I benefited from the materials recommended by the <i>OOPS</i> service in ...	S1	4.2	3.6	3.7	
		S2	3.7		3.7	3.7
4	The recommendations of the <i>OOPS</i> service did not help me to solve the questions in ...	S1	3.6	3.3	3.5	
		S2	3.3		3.3	3.8
5	Most often, when I clicked on a recommended link, I had to browse to find a helpful page in ...	S1	3.0	2.6	2.6	
		S2	2.9		2.3	2.6
6	Most often, I was not able to find a helpful page even by browsing in ...	S1	3.6	2.9	3.3	
		S2	3.5		3.5	3.7
7	Most often, I did not have to browse, one of the recommended pages was helpful for the question in ...	S1	3.7	3.8	3.1	
		S2	3.4		3.2	3.6
8	Most often, the helpful page was the first or second in the recommended list in ...	S1	3.0	3.6	3.2	
		S2	2.8		3.3	3.7
9	The adaptive annotations of the recommended links helped me to choose appropriate recommendations.		3.2	3.6	3.6	3.3
10	I did not use the adaptive annotations of the recommended when choosing the recommendations.		3.1	3.0	2.4	2.7
11	I think the idea of recommending reading material to help solve the question is good, but implementation could be better.		3.7	3.9	3.9	3.9
12	The usual textbook was more helpful for solving questions than the recommendations of the <i>OOPS</i> service.			2.5		2.4
13	It was easier to find a helpful page among the recommendations of the <i>OOPS</i> service than in a textbook.			4.2		3.9
14	<i>OOPS</i> recommendations were better only because they were accessible within the same system, which was easier to use than the book.			3.0		3.4
15	<i>OOPS</i> recommendations were better because <i>OOPS</i> brought the helpful pages much closer, and I did not have to look for them in the entire book.			3.8		4.0
16	If you have any additional comment on the work of the <i>OOPS</i> service, please provide it here:	X				

Session encoding (Q.3-Q8):	Open-corpus	Closed-corpus	Book
Answer encoding:	3.5-5.0: Agree	2.6-3.4: Neutral	1.0-2.5: Disagree
Question encoding	Questions favoring <i>OOPS</i>		Questions critiquing <i>OOPS</i>

7.5.1 Questionnaire: open-corpus vs. closed-corpus recommendation

Questions from 3 to 8 ask students to evaluate the quality of recommendation they receive from *OOPS*, separately for Sessions 1 and 2. Figure 80 presents an aggregated view on the answers from students evaluating the *Open-corpus* version of the system. Figure 81 visualizes similar data for the *Closed-corpus* sessions.

For Question 3, there is not strong difference between the open-corpus and the closed-corpus conditions: on both figures about 60% of answers are positive, which means most of the students felt that they benefited, in general, from *OOPS* recommendations.

Question 4: (“*The recommendations of the OOPS service did not help me to solve the questions*”) asks students if *OOPS* helped them to solve *QuizJET* exercises. The question is formulated in a negative way, thus students disagreeing with the question were favoring *OOPS* more. Both for open-corpus condition, and for closed-corpus one about 50% of students disagreed or strongly disagreed with the question, i.e. they felt that *OOPS* helped them to solve self-assessment exercises.

Overall, the results of Questions 3 and 4 are consistent with the system usage statistics and the learning effects analysis. *OOPS* was not less effective in helping students improve their exercises solving skills than mere practicing with exercises. However, it did help students to improve their conceptual knowledge and thus contributed to their learning in general.

Questions 5 and 7 ask whether students had to browse in order to find a helpful page. Again, the open-corpus and closed-corpus groups have shown very similar results. About 30% of students in both conditions answered to the Question 5 that they did not have to browse for a helpful page. When answering Question 7, about 60% in both conditions confirmed that most often one of the recommended pages was helpful right away.

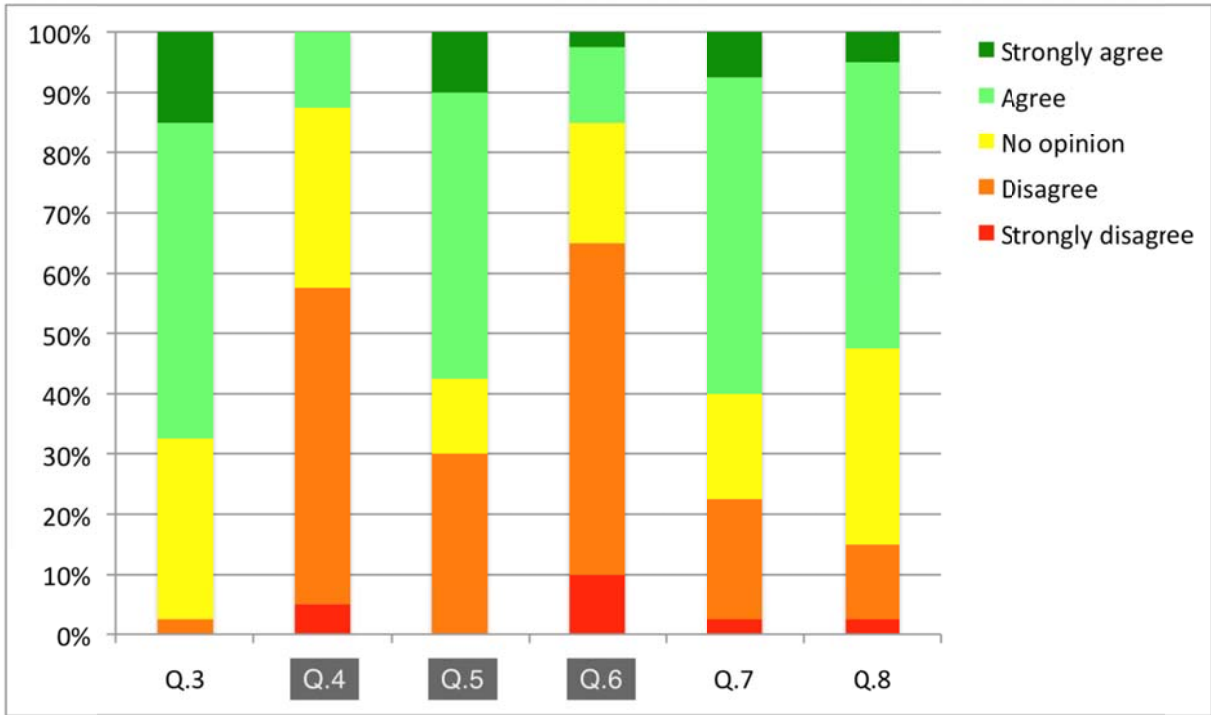


Figure 80: Questionnaire: open-corpus recommendation

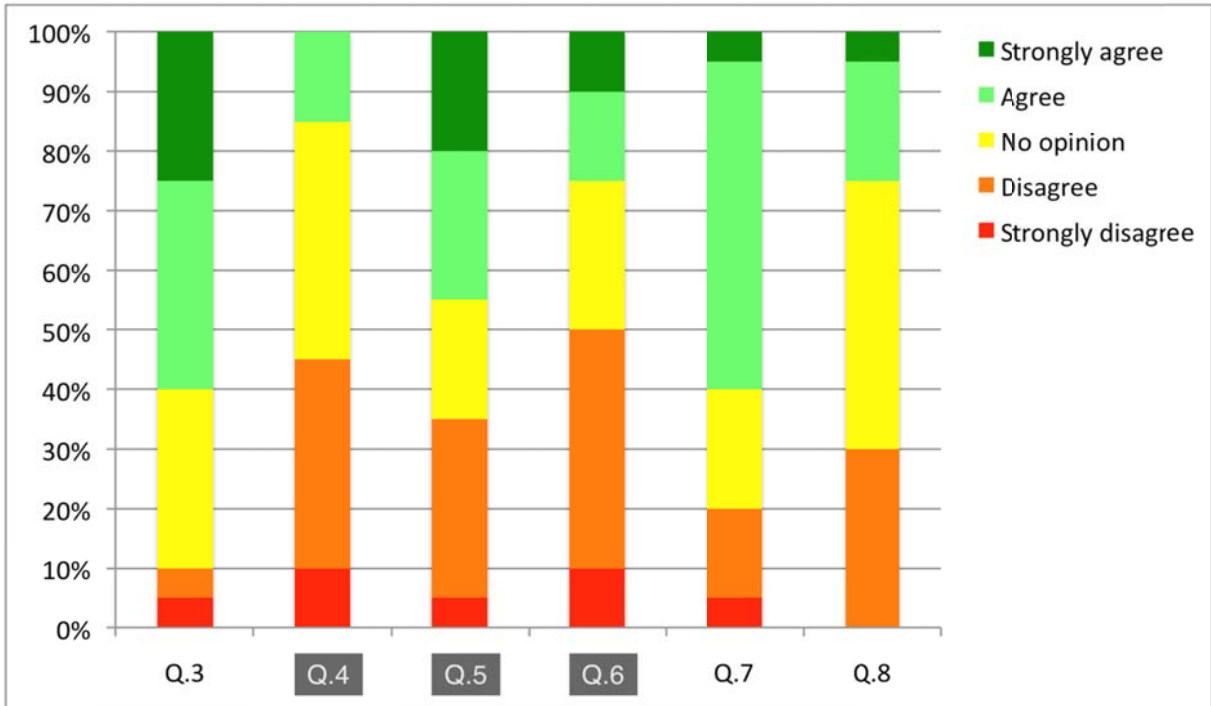


Figure 81: Questionnaire: closed-corpus recommendation

Question 6 is relevant to the previous two. It asks students to report whether they were able to find a helpful page at all. 65% of open-corpus students and 50% of closed-closed corpus students said that most often there could find a page they needed. These numbers are consistent with the immediate feedback students provided on recommendations during the working sessions: *Perceived Value of Recommendation* parameter computed in Section 7.4.1.4 varied from 50% to 72% and was also a little higher for open-corpus condition).

Finally, Question 8 estimates student opinion about the accuracy of the top two recommendations. Students using open-corpus systems felt a little more positive about the pages recommended as first or second item in the list: about 50% found them useful. Only 25% of students from the closed-corpus sessions answered this question positively.

7.5.2 Questionnaire: general questions

Figure 82 summarizes the rest of the questions in the questionnaire. They asked students about various features of the system from interface and implementation to the effectiveness of adaptive annotation and system usefulness compared to the textbook.

Overall, students liked the interface of *OOPS*. Almost 80% of answers to Question 1 were positive.

Question 2 asked if students use the help of *OOPS* at all, while solving self-assessment exercises. More than 90% answered affirmatively.

Questions 9 and 10 verified the helpfulness of adaptive annotation *OOPS* used to navigate students to the most important page according to their level of knowledge. More than 50% of students answered that these annotations helped them to choose the appropriate page

(Question 9). Almost 50% disagreed they did not use adaptive annotations when choosing among recommended links.

According to the answers to Question 11, students liked the overall idea of recommending reading material to help solving practice exercises; however they believed *OOPS* could do a better job implementing it (80% of answers).

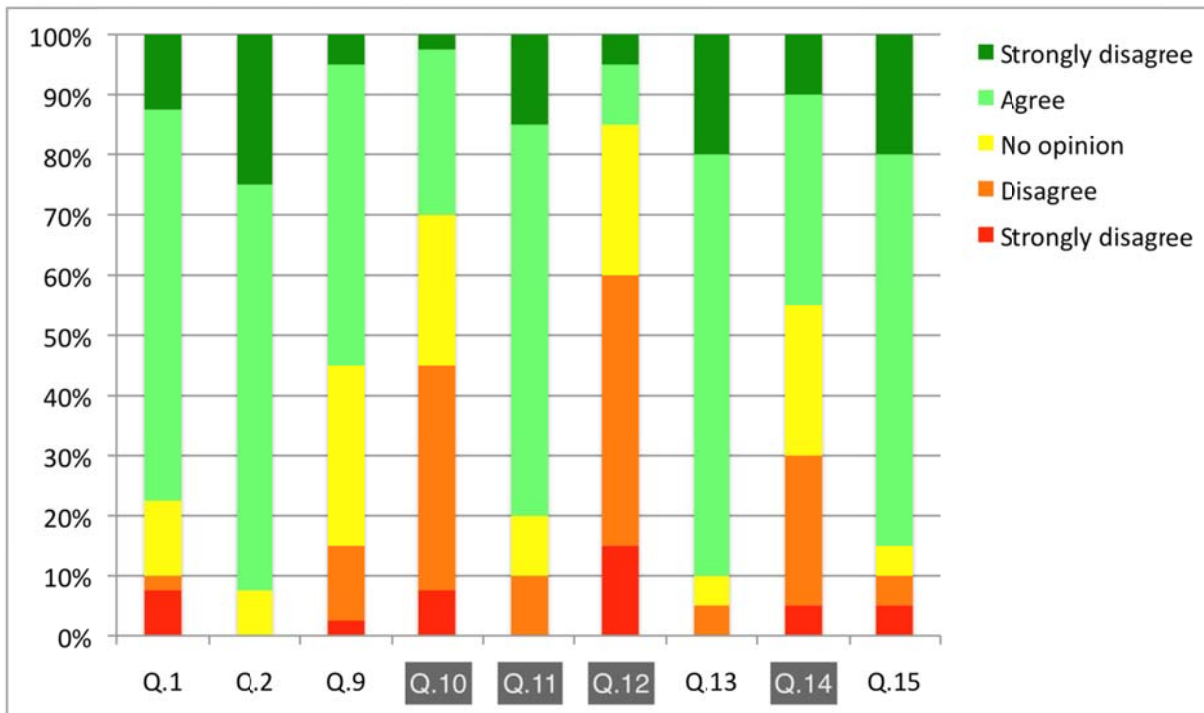


Figure 82: Questionnaire: general questions

Questions 12, 13, 14 and 15 compared *OOPS* with a textbook. Only students who used both the *Open-corpus* version and the *Textbook* version of the system (groups *B* and *D*) answered these questions. 60% of students disagreed that the textbook was more helpful than *OOPS*, and only 15% agreed (Question 12). 90% of students felt that “*It was easier to find a helpful page among the recommendations of the OOPS service than in a textbook*” (Question 13). Question 14

suggests a possible reason, why *OOPS* is more helpful: “*OOPS recommendations were better only because they were accessible within the same system, which was easier to use than the book*”. Students’ opinions about this statement were very divided: 45% agreed and 30% disagreed. Question 15 formulated the reason for the *OOPS* usefulness in more favorable way for the system: “*OOPS recommendations were better because OOPS brought the helpful pages much closer, and I did not have to look for them in the entire book.*” This statement tries to explain the overall approach implemented by *OOPS* in very plain words: to provide helpful reading content where and when students need it. Much more consensus demonstrated been shown when students were answering this question: 85% agreed that this is why *OOPS* is better.

Finally, Question 16 asked students to provide any additional comments about *OOPS*. Most of criticism was related to the quality of recommendation (eight subjects). Three students said that the interface of *OOPS* needed an improvement. Three asked for better reading material with more examples and explanations. Eleven students suggested some modifications, e.g. tracking of pages that the user has already read and adapting to it, or implementing search functionality. Seven students believed that the overall idea of recommending reading material is the strongest part of the system. Three students liked everything.

7.6 CONCLUSION

This chapter presented the design and the results of the evaluation of the developed e-Learning service *OOPS*. The evaluation was organized as a controlled experiment comparing different aspects of open-corpus personalization implemented by *OOPS* against two other versions of the system:

- the traditional closed-corpus version that apply the same personalization techniques with on the manually contracted concept-based content model, and
- the textbook-based version that did not provide any adaptive support to the students; instead they had to find helpful reading material in a regular textbook.

The most important results have been obtained when measuring the effects of *OOPS* on subjects' learning. Using open-corpus version of the system, students were able to achieve significant learning during both experimental sessions. *OOPS* significantly improved students' knowledge gain when they work with more challenging learning material. In comparison, students using the textbook demonstrated no significant learning while working with complex topics. It has been shown that weaker students benefited the most from using *OOPS*; the textbook did not have this effect on weaker student. *OOPS* helped to maintain a more balanced learning by significantly improving gain in conceptual knowledge, no such effect – for the textbook. In general, in all these comparisons the open-corpus *OOPS* outperforms the Textbook-based version of the system. On no tests, the results of *OOPS* were inferior to the traditional closed-corpus personalization.

The analysis of log-based data has shown that open-corpus version of the system is capable of providing personalization of similar quality to the close-corpus one. Students accessed similar percentage of recommended pages when using both systems. They chose recommendation of similar ranks, which indicated close performance in terms of computing page relevance. Students did not have to browse more in order to find a helpful page, when they used the open-corpus system. Most important, the perceived value of closed-corpus recommendation reported by students was not higher than the one of the open-corpus.

The post-questionnaire administered at the end of the experiment also showed, that, from the subjects' perspective, *OOPS* provides the recommendation of comparable quality to the closed-corpus system. Generally, students reported positive opinion about most aspects of the system, including its interface, recommendation and adaptive annotation helpfulness. When comparing *OOPS* with the textbook, students were strongly in favor of *OOPS*. Most of the students appreciated *OOPS* for its ability to bring helpful pages closer to the task at hand. Although several students criticized recommendation accuracy, they did so both for the closed and the open-corpus versions of the system.

The evaluation has successfully achieved its main goal by demonstrating that *OOPS* helps students to learn, that *OOPS* generally outperforms the *Textbook*, and that it can support effective personalization of similar quality to the traditional closed-corpus approaches, while retaining all the benefits of fully-automatic open-corpus content modeling.

8.0 CONCLUSION AND DISCUSSION

8.1 SUMMARY OF THE RESULTS

This dissertation has explored one of the important problems of adaptive e-Learning (and adaptation in general) – open-corpus personalization, – and proposed a solution for it. The first section of this final chapter recapitulates the main results of the dissertation. It revisits the list of research questions formulated in the Introduction and outlines the contributions of this work categorized by research areas.

8.1.1 Revisiting the research questions

1. *What are the principal ways to implement semantic modeling of open-corpus content and what are their pros and cons?*

Chapter 3.0 :

A comprehensive answer to this question was given in Chapter 3.0 . It provides a detailed description of the three principle approaches towards facilitating the creation of semantic models of open-corpus information resources. Every approach is illustrated with one or several project carried out by the author during his PhD study:

- Automatic indexing of structured content allows for creation of detailed semantic models without the help of a human author, but relies on domain-dependent tools and heuristics;
- Coarse-grained content modeling and personalization has been proven effective in the context of e-Learning, but large topics cannot support adequate precision of student modeling;
- Mediation of external models has demonstrated that automatic translation of knowledge models based on domain ontology mapping is possible to perform with an acceptable level of accuracy, but the number of content collections provided with ontology-based fine-grained content-models is rather low.

The most important outcome of this analysis has been the acknowledgement of a potential synergy between these three approaches. By putting them together it became possible for the author to bridge the individual weaknesses of these approaches and build up their main dissertation approach proposing a more comprehensive solution to the problem of open-corpus personalization.

2. *What is the potential source of domain knowledge in Web-content, and how this knowledge can be automatically extracted and used for modeling open-corpus content on a large scale and across multiple domains?*

Chapter 4.0 :

The important assumption of the main dissertation approach presented in Chapter 4.0 is the abundance of online content that has been created by domain experts, organized into easy-to-read collections and formatted and structured in a consistent way to facilitate its understanding. Section 4.2 presents this assumption on the conceptual level. Section 4.3

explains how this narrow hidden layer of domain semantics could be automatically extracted from a large variety of collections and leveraged to facilitate fully-automatic personalization of open-corpus content. Extracted open-corpus models express domain knowledge in terms of collection-specific topics. A reference domain ontology is used for mapping topic-based models and bringing them to the same semantic denominator.

3. *What techniques and sources of information can and should a mapping algorithm utilize in order to support mapping between ontologies and topic-based models?*

Chapter 6.0 :

Chapter 6.0 described the developed *ATOM* algorithm for mapping fine-grained well-designed ontologies into automatically harvested topic-based models of open-corpus content collections. It utilizes four principle techniques – each based on its own source of information:

- name-based mapper uses naming labels of concepts and topics,
- instance-based mapper uses concepts' *<rdfs:comment>*'s and pieces of open-corpus content associated with topics;
- structure-based mapper uses the relations between the elements of the two models;
- order-based mapper used the sequence of learning goals of the central system and the order of topics in the tutorial.

The resulting mapping computed as a combined product of these four mappers is used by *OOPS*. The results of the main study (demonstrated in Chapter 7.0 and, especially, in Section 7.4.1) show that *ATOM-OOPS* implementation is capable to support open-corpus personalization of high quality. However, a dedicated experiment is needed to establish the limits of this approach – how would it perform in a different domain, or for a different

collection. Yet, it should be noted, that from the mapping perspective, the settings of the proof-of-concept implementation were not restrictive. The requirements to the central domain ontology were fiery lenient; a more formal model could provide extra capabilities in terms of reasoning. The implementation of the open-corpus model extraction benefited only from the table of contents and topics headings; most of the collections discussed in the dissertation could provide more detailed information and produce a more descriptive model. Theoretically speaking, the minimal requirements for the mapping algorithm to produce some results would be that at least the name-based mapper had an input.

4. *Can the main dissertation approach based on automatic extraction of topic-based models and automatic mapping of these models into the domain ontology be used to support open-corpus personalization for e-Learning?*

Chapter 5.0 :

Chapter 5.0 describes the details of the *OOPS* service implementing the proposed approach. *OOPS* employs several adaptation techniques to provide student with adaptive access to the open-corpus supplementary reading material in the domain of Java programming. Chapter 7.0 presents an empirical evaluation of the service that has demonstrated its efficiency as an AES.

5. *Will the implementation of the main dissertation approach be able to contribute to students' leaning?*

Chapter 7.0:

Section 7.3.1 investigates the effectiveness of *OOPS* used together with the *QuizJET* system from the perspective of general learning. The controlled study based on a pre-test, a working session with the system and a post-test shows that users of *OOPS* were able to

significantly improve their knowledge of the material. This effect was observed for all sessions and all groups using *OOPS*.

6. *What will be the main effects of the designed personalization service?*

Chapter 7.0:

In addition to the general learning effect of using *OOPS*, the evaluation has been able to statistically prove two more significant effects. Section 0 shows that *OOPS* improves learning of more challenging material (more difficult topics) compared to a regular textbook. Section 7.3.4 demonstrates that *OOPS* increases knowledge gain of conceptual learning material compared to a regular textbook. Finally, Section 7.3.3 shows traces of a positive effect of working with *OOPS* for lower achieving students, however due to the small scale of the study it was impossible to achieve statistical significance.

7. *Will the designed service be able to maintain the adaptation of comparable quality to the conventional closed-corpus approach?*

Chapter 7.0:

In the course of the evaluation experiment the open-corpus version of *OOPS* has been compared with its closed-corpus version. Section 7.4 investigates their performance based on the analyses of the systems' logs. The collected statistics shows that on no metric evaluating the quality of personalization the open-corpus version of the system has been outperformed by its closed-corpus counterpart. The analysis of the students' answers to the post-questionnaire presented in Section 7.5 demonstrates that, from the point of perceived quality of personalization, both versions of the system were equally useful.

8.1.2 Contribution to the field of open-corpus personalization

The main scientific contribution of this dissertation is the new open-corpus personalization approach that possesses a combination of important characteristics:

1. it supports a full cycle of open-corpus personalization: from content organization and modeling, to user modeling and adaptation;
2. it allows for creation of *semantic*⁴⁷ models of open-corpus content: by connecting open-corpus resources with the concepts of a domain ontology it enables adaptation to students' knowledge;
3. it requires little to no involvement of a human author: only the entry page URL of an open-corpus collection has to be provided manually;
4. it does not rely on domain-dependent techniques: no domain-driven tools or heuristics is required, although their use could improve the quality of extracted models;
5. it is compatible with a wide range of adaptation technologies: the developed adaptation service implements adaptive recommendation/ordering and adaptive annotation, but other techniques are possible, as well;
6. it can support personalization of many type's of content collections: the only structural element used by the current implementation of open-corpus model extraction and mapping is table of contents;
7. it is applicable in several application fields beyond of e-Learning: many adaptive information systems supporting access to the open-corpus content could benefit from the OOPS-enabled coarse-grained personalization.

⁴⁷ Capability to support semantic modeling of content, domain and students is especially important for the field of e-Learning

None of these features is unique on its own. They have been implemented in other open-corpus personalization approaches. What distinguishes the solution presented in this dissertation is the combination of all these characteristics. This is especially important in the field of e-Learning. None of the other existing solutions could be applied on a large-scale for providing adaptive access to open-corpus instructional content (see Section 2.2 for a detailed analysis). The approach presented in this dissertation is the first to do so.

OOPS service has been developed as a proof-of-concept for the proposed solution. The design of *OOPS* and the following evaluation experiment helped to address and practically validate as many of the listed features as possible. Further work is needed to experimentally verify the independence of proposed approach from the subject domain, the content collection type and the application field.

8.1.3 Contribution to the field of semantic web and ontology mapping

Ontology mapping has been used rather as a tool than a subject of research in this dissertation. Yet the developed *ATOM* algorithm had to implement several unconventional solutions and introduce some new features into the arsenal of ontology mapping techniques in order to address the challenges of mapping a formal ontology to a topic-based model:

- the name-based mapper of *ATOM* makes a heavy use of IR techniques to handle long, subjective and often-irregular topic names;
- the instance-based mapper of *ATOM* cannot rely on the tradition notion of an instance, which is an individual of a class; instead it uses `<rdfs:comment>`'s of concepts and content of pages titled by topics as their instances;

- the structure-based mapper of *ATOM* has to take into account undefined semantics of relations in the topic-based model;
- finally, the order-based mapper exploits a unique source of information about the elements of the two models.

The last item in this list is especially important. None of the ontology and schema mapping techniques known to the author considers order of elements a useful source of information; and, from the point of the general problem of schema mapping it is a valid assumption. However, there exist a number of application fields, where the order of elements in a schema either has a stable meaning or simply follows a certain tradition.

In the case of e-Learning, the sequence of learning goals is a good example of meaning-driven ordering of knowledge. This meaning comes from the pedagogical model that defines a possible sequence of learning topics in this domain. It is important, that in the same domain, different content collections will have similar pedagogical models, which creates an opportunity to reuse them for schema mapping. There exist other examples, where ordering of elements is rather pragmatic than semantic, yet stable across schemas. For instance, multiple online forms tend to place similar fields in similar positions. The order of fields in different bibliography systems is also similar, even when they are named differently. It is important to mention, that order-based mapping cannot be the only technique used by the mapping algorithm. But it can help to refine the results of other mappers by leveraging the rules of elements sequencing as an extra evidence of their similarity.

In addition to ontology mapping, some of the results of this dissertation might be interesting for a general field of Semantic Web. For example, the developed Java ontology is an important practical contribution.

If considered in a broad sense, this dissertation also addresses the problem of knowledge acquisition bottleneck that is a classic challenge of Semantic Web (Maedche and Staab 2001), and Artificial Intelligence (Forsythe and Buchanan 1989) fields.

8.2 DISCUSSION

This section concludes the dissertation with the discussion of some limitations of the proposed approach and the outline of possible directions for future research.

8.2.1 Limitations of the proposed approach

An important limitation of the proposed solution is that the modeling of open-corpus content takes a lot of time. The model extraction and mapping cannot be done interactively. Any change to the content collection will result in complete remodeling of the entire collection. This restricts the applicability of the approach to the areas that do not require real-time personalization of dynamic online content and can combine the off-line content modeling with the online personalization.

One of the main assumptions of the approach is the existence of high-quality structured collections of online information resources. In the domains that lack such collection the algorithm will not be applicable. If the quality of the collection is low, the quality of the provided personalization will be also low.

The approach cannot function without a core domain ontology. Even though the number of high-quality ontologies is growing, not every domain has been provided with one. Moreover,

the approach assumes a certain level of agreement, when it comes to the ontology design and the material of open-corpus collections. There are three kinds of subject domain, where the proposed approach might be not applicable:

- Developing domains, where knowledge is dynamic, can quickly expire and/or extend. In such domains it is harder to build and maintain an ontology representing “a shared conceptualization”; it is also harder to find a set of high-quality information resources that cover the same area of knowledge and can be easily aligned with each other and with the central ontology (e.g. Social Web, Nanotechnology, Military Science);
- Controversial domains, where multiple viewpoints exist for the same facts, relations between the facts, even the existence of the facts. For a similar reason, it will not be easy to develop an ontology and find a set of information resources overarching the entire domain and agreeable for everybody (e.g. History, Ecology, Politics);
- Informal domains, where knowledge are poorly structured. In such domain ontology engineering is a challenging task. Authoring well-structured information resources presenting the domain knowledge in a semi-formal manner can be also problematic (e.g. Arts, Literature).

There is another important consideration (rather than a limitation) that is a potential source of problems for any open-corpus adaptive systems, especially in the field of e-Learning: content quality assurance. When the system is responsible for the full cycle of open-corpus personalization, filtering out erroneous, inconsistent, poorly written information resources becomes a challenge. The proposed approach solves this problem by making the human author (teacher) responsible for initial discovery and selection of the open-corpus content collections. Once the author ensures the quality of the collection, s/he provides its entry URL to OOPS; the

system becomes responsible for the remaining open-corpus personalization steps (see Section 1.1.3).

8.2.2 Future work directions

The future research is planned in the three main directions:

- further evaluation of the proposed approach;
- improvement of the current implementation;
- developing a bigger picture.

8.2.2.1 Further evaluation of the proposed approach

The conducted evaluation study has demonstrated a number of positive effects of the developed e-Learning service *OOPS* implementing the proposed approach. Unfortunately, the scale of the study was not big enough to fully verify all the hypotheses. A bigger number of users is necessary to statistically confirm (or reject) the effect of *OOPS*'s recommendations on lower achieving students.

A semester-long experiment within a formal university course would help to check how students use the system in meaningful settings for real learning. For example, during the controlled experiment, the amount of time to work with the system was fixed. As a result, students using the *Textbook* version of the system spent the time almost entirely on drilling with the same exercises again and again. Students using *OOPS*-enabled versions of the system had more material to explore. This, probably, provided them with a more meaningful learning experience, but reduced the amount of time they could invest into practice. Thus, *Textbook*-using students achieved higher rates of successfully answered *QuizJET* exercises. Yet, the author's

previous experience suggests that in real classroom settings, adaptation might increase the students' motivation to work with the self-assessment exercises and can help them to be more successful in answering them, especially for more complex topics in the course. On the other hands, it is possible that when students use the system for their own sake without being reimbursed for their efforts, they will not use *OOPS* that much.

Several features of the proposed approach are yet to be validated. Will *OOPS* be successful in another subject domain or with the material from another content collection? Will it be able to maintain adequate personalization of the content from multiple collections, especially if the content will be of multiple types? Will it require implementation of other adaptation technologies? All these questions can be answered only based on the results of dedicated user studies.

8.2.2.2 Improvement of the current implementation

Most if the new experiments mentioned in the previous subsection would require significant enhancements of the modeling component of *OOPS*. There are two main components that can be improved: the model extraction component and the model mapping component. The current implementation of the model extraction can take into account only the topic hierarchy provided by table of contents. Links between the topics, special formatting of text fragments, indexes and thesauri can be employed (if available) to improve the structure of the harvested models, the level of details, the meaning of relations, etc. The mapping algorithm could be modified to take into account more detailed models. *WordNet* (or domain-specific thesauri) can be employed to enhance term vectors with information about synonyms, homonyms, antonyms, etc. Structure-based mapper can be improved to process relations more intelligently.

Once the knowledge base of the service stores several open-corpus models, it will open the entirely new opportunity for improving the modeling procedure. Cross-mapping of several open-corpus models should provide *ATOM* with a lot of useful information. New topics can be mapped with the help of existing similarity tables, or, vice versa, existing mapping can be refined based on new findings.

On the interface level, one important improvement for the service would be to include students' activity with the open-corpus content in the student modeling and adaptation loop. The current implementation traces students navigation through the recommended pages, but does not react on it: if a student accesses a page, her knowledge of related concepts do not change (even though the system knows, which concepts they are) and next time she visits the page it looks exactly the same to her. There are several ways to address this. An interesting approach to try would be to add a social dimension to *OOPS*'s interface. Students' reports on whether the page was "*HELPFUL*" or not in solving an exercise could be aggregated and used to re-rank the list of recommended topics, hide unpopular topics from navigation, or add another layer of annotation based on social feedback.

8.2.2.3 Developing a bigger picture

WWW is not without structure and not without knowledge. A great deal of information on the Web is created by people who feel confident enough to engage in authoring information and sharing it with the rest of us. To an extent, Web is one large socially maintained information base – one large Wikipedia. This information is broken into pieces, encoded with CSS classes, HTML formatting, social tags, etc.; it has been linked to related pages; it has indexed by Google and is

now available through its search API⁴⁸. All this data could be extracted, processed and used to generate an extra value – a layer of thin semantics, a Web of semi-structured knowledge.

An important aspect of such project is that once it starts accumulating knowledge models, they would facilitate further information extraction. The knowledge based would use the existing knowledge to harvest and process the new pieces of semantics; and will use the new knowledge to refine the existing models.

Social media applications and their users could be employed for information quality assurance. The social network overlay can be used as another layer of structure facilitating the harvesting of new knowledge and refinement of existing models.

⁴⁸ <http://code.google.com/apis/customsearch/v1/overview.html>

APPENDIX A

ARCHITECTURE *ADAPT*²

*ADAPT*² (Advanced Distributed Architecture for Personalized Teaching and Training) is aimed at providing personalization adaptation in the distributed Web settings. It includes several important components (learning portals, user modeling servers, activity servers and value-adding services), where each component is a separate application. A learning portal provides the authentication of the users and in many respects act as a regular learning management system. The main difference of a learning portal is that it does not owe and does not serve the learning content. It stores the links to the learning content being served by the activity servers (or content providers). The activity servers supply content such as problems, example, tutorial, visualizations, etc. They can implement adaptive technologies or can simply provide the access to the non-adaptive content. Value-adding services are capable of enriching the existing content with new functionality, such as adaptation. The user modeling server tracks students' actions reported by other components and reports modeling information about students by request. The communication between the components of *ADAPT*² is performed through HTTP. The schema of the architecture is depicted below.

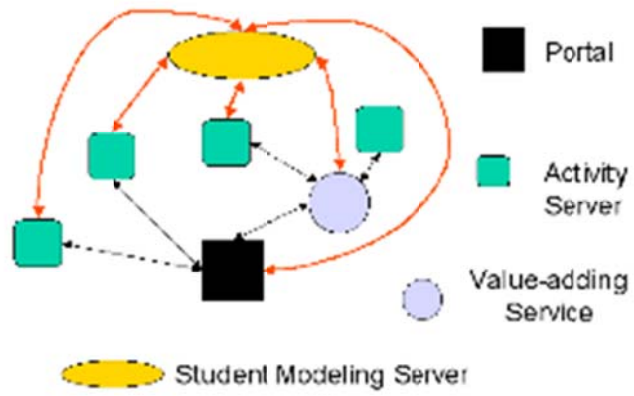


Figure 83: Main components of *ADAPT*²

APPENDIX B

USER MODELING SERVER *CUMULATE*

The *CUMULATE* server (Centralized User Modeling for User and Learner-AdapTive Environments) has been developed as a generic student-molding server for *ADAPT*². *CUMULATE* represents information about a student on two levels: the event storage and the inferred user model. All students' actions with activity servers are sent to *CUMULATE* using a standard HTTP-based event reporting protocol. The structure of this protocol allows an interactive component to report the kind of event (i.e., a specific learning activity or a specific step within more complex activities), the progress (for example, success or failure) and any additional component-specific information). *CUMULATE* adds a timestamp to each reported event and stores it permanently in the event storage. The event storage is open to a variety of inference agents that process this data in different ways and convert it into a more familiar form of name-value pairs that altogether form the inferred user models. Different agents attempt to infer different user parameters using different methods. For example, some agents could be focused on student's knowledge; other could monitor student's interests, yet others could infer their level of motivation. Different agents could attempt to infer the same parameters from the same event storage using different methods. The architecture anticipates the use of internal (i.e., in-server) and external inference agents. The latter kind requests information from event storage

and update inferred user models using dedicated protocols. Moreover, an application could access event storage directly and use some in-application inference agent to process the events in a specific way. The figure below represents the basic architecture of *CUMULATE*.

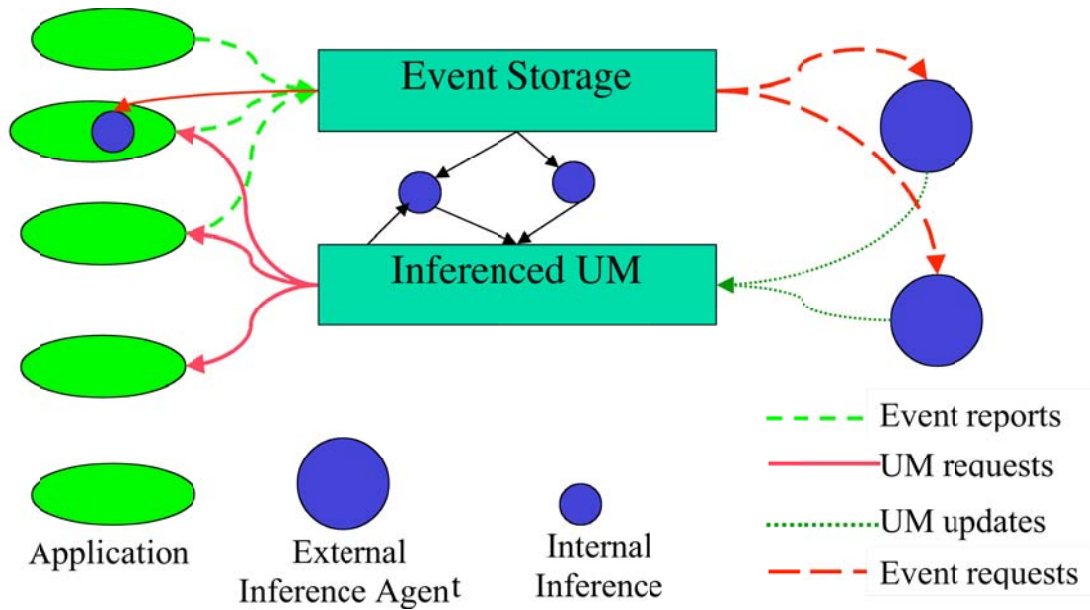


Figure 84: Principle architecture of *CUMULATE*

APPENDIX C

LEARNING PORTAL KNOWLEDGE TREE

A learning portal is one of the core components of *ADAPT*². It represents the needs of course providers – teachers (trainers) and their respective universities (or corporate training companies). A portal plays a role similar to modern Learning Management Systems (LMS) in two aspects. First, it provides a centralized single-login point for enrolled students to work with all learning tools and content fragments that are provided in the context of their courses. Secondly, it allows the teacher responsible for a specific course to structure access to various distributed fragments according to the needs of this course. Thus, a portal is a component of the architecture that is centered on supporting a complete course. Replicating the familiar functionality of an LMS, it provides a course-authoring interface for the teacher and maintains a runtime interface for the student. The difference between this and LMS is an architectural separation of the unique course structure created by the teacher or course author from reusable course content and services. A portal has the ability to query activity servers for relevant activities and launch remote activities selected by students or by the portal itself. Student interface of the *Knowledge Tree* learning portal is shown by two figures from Section 7.2.1. **Figure 73** demonstrates the login phase and choosing of a learning activity. **Figure 74** shows the work with a learning activity (a *QuizJET* exercise) launched within *Knowledge Tree*.

APPENDIX D

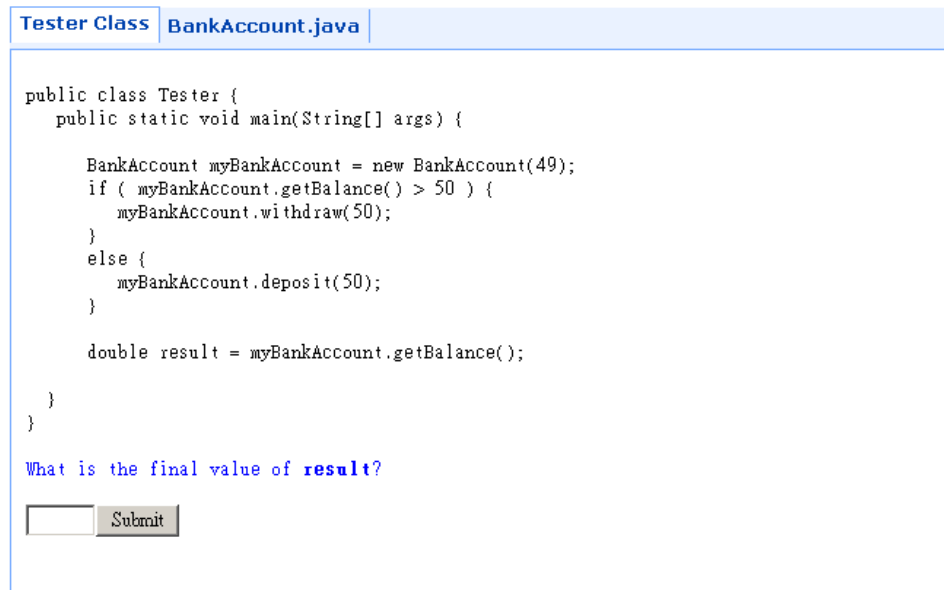
EXERCISE SYSTEM *QUIZJET*

QuizJET system has been designed to support Web-based authoring, delivery and evaluation of parameterized exercises for Java programming language. *QuizJET* can be used for assessment and self-assessment of students' knowledge of broad range of Java topics from language basics to advanced concepts, such as polymorphism, inheritance, and exceptions.

A typical *QuizJET* exercise consists of a small Java program. One (or several) numeric value in the text of the program is instantiated with a random parameter when the exercise is delivered to a student. As a result, the student can access the same exercise multiple times with different values of the parameter and different correct answers. To answer an exercise, the student needs to examine the program code and solve a follow-up task. The task can be one of two kinds: “*What will be the final value of an indicated variable?*” or “*What will be printed by the program to the standard output?*”

A tabbed interface design has been implemented to allow exercises consist of several classes. The driver class, containing the main function, is always presented on the first tab. It is the entry point to the exercise. The first tab also includes the exercise task and the field for a student's input. The system's feedback is also presented in the first tab after a student's answer

has been evaluated. An example of a *QuizJET* exercise is presented in Figure 85. By clicking on different tabs students can switch between the classes to access the full code of the program.



The screenshot shows a web interface with two tabs: "Tester Class" and "BankAccount.java". The "Tester Class" tab is active, displaying the following Java code:

```
public class Tester {
    public static void main(String[] args) {

        BankAccount myBankAccount = new BankAccount(49);
        if ( myBankAccount.getBalance() > 50 ) {
            myBankAccount.withdraw(50);
        }
        else {
            myBankAccount.deposit(50);
        }

        double result = myBankAccount.getBalance();
    }
}
```

Below the code, the question is: "What is the final value of **result**?"

At the bottom, there is an input field and a "Submit" button.

Figure 85: Presentation of a QuizJET exercise

Once a student enters an answer and clicks the “*Submit*” button, *QuizJET* reports the evaluation results and the correct answer (Figure 86). Whether the result was correct or not, the student can click the “*Try Again*” button to assess the same exercise with a different value of the generated parameters. This option provides students with an opportunity to master a particular topic.

Tester Class BankAccount.java

```
public class Tester {
    public static void main(String[] args) {

        BankAccount myBankAccount = new BankAccount(49);
        if ( myBankAccount.getBalance() > 50 ) {
            myBankAccount.withdraw(50);
        }
        else {
            myBankAccount.deposit(50);
        }

        double result = myBankAccount.getBalance();
    }
}
```

What is the final value of **result**?

CORRECT!

Your Answer is:
99.0

Correct Answer is:
99.0

Figure 86: Feedback in QuizJET

APPENDIX E

PRE- AND POST-TESTS OF JAVA KNOWLEDGE

E.1 PRE-TEST (TOPIC SET 1)

Login _____

Question 1

The _____ operator, followed by a class name and parameters, is used to construct objects.

Question 2

Class **Person** has a method **setName** that takes one String parameter and does not return any results. Write below a line of code that calls this method for an object **baby** of class **Person**, that sets the name of the baby to “**John**”.

Question 3

When an instance (data) field of a class is declared with this access specifier: _____, this means that the field can only be accessed by methods of the same class and not by any other method outside the class.

The name of the constructor is always the same as the name of the _____.

Question 4

Consider partial implementation of the class Rectangle:

```
public class Rectangle {  
  
    private double x;  
    private double y;  
    private double height;  
    private double width;  
  
    public Rectangle (double x, double y, double height, double  
width) {  
        this.x = x;  
        this.y = y;  
        this.height = height;  
        this.width = width;  
    }  
  
    .....  
}
```

Assume, that one more method has been added to the class:

```
public void move (int moveX, int moveY) {  
    x = x + moveX;  
    y = y + moveY;  
}
```

What would be the output of the following code fragment using the new method? Assume necessary methods are implemented

```
.....  
Rectangle myBox = new Rectangle(50, 40, 10, 10);  
myBox.move(10, 30);  
System.out.println(myBox.getX());  
System.out.println(myBox.getY());  
.....
```

Output:

Question 5

This special word _____ is used in Java to indicate that this variable is a constant.

Question 6

What is the output of the following code segment?

```
int a = 4 + 5;
int b = 5 + 4;

if (a != b)
    System.out.println(" Not equal ");

if (a == b)
    System.out.println(" Equal ");
```

Question 7

Consider the code fragment below.

```
int res = 10;
if (res < 4 && res > 0)
    res = res + 1;
else {
    res = res + 5;
}
res = res * 2;
```

What will be the final value of the variable `res`? _____

Question 8

Type _____ in Java is used to represent logical values.

E.2 POST -TEST (TOPIC SET 1)

Login_____

Question 1

To invoke (call) a method for an object, this symbol _____ is used between the name of the object and name of the method.

Question 2

Consider the code fragment below. Please mark the lines, where Objects are created

```
Rectangle rct = new Rectangle(10, 20, 100, 200);  
Point org = new Point(50, 60);  
int i = 5;  
String grt = new String("Hello, Rectangle!");  
System.out.println(grt + " Your area is " + rct.getArea());  
public static void main(String[] args)
```

Question 3

A standard method definition contains the following parts (choose 1 variant below):

1. the return type, the name of the method, and a list of the parameters (if any)
2. an access specifier, a list of the parameters (if any), and the body of the method
3. an access specifier, the return type, the name of the method, a list of the parameters (if any), and the body of the method
4. an access specifier and the the return type

Question 4

Consider partial implementation of the class Rectangle:

```
public class Rectangle {  
  
    private double x;  
    private double y;  
    private double height;  
    private double width;  
  
    public Rectangle (double x, double y, double height, double  
width) {  
        this.x = x;  
        this.y = y;  
        this.height = height;  
        this.width = width;  
    }  
  
    .....  
}
```

Assume, that one more method has been added to the class:

```
public void toSquare (int side) {  
    height = side;  
    width = side;  
}
```

What would be the output of the following code fragment using the new method? Assume necessary methods are implemented

```
.....  
Rectangle myBox = new Rectangle(50, 40, 10, 10);  
myBox.toSquare(25);  
System.out.println(myBox.getHeight());  
System.out.println(myBox.getWidth());  
.....
```

Output:

Question 5

What is the main difference between a constant and a regular variable:

Question 6

What is the output of the following code segment?

```
int a = 3 + 3;
int b = 2 + 2;

if (a != b)
    System.out.println(" Not equal ");

if (a == b)
    System.out.println(" Equal ");
```

Question 7

Consider the code fragment below.

```
int res = 10;
if (res < 4 || res > 0)
    res = res + 1;
else {
    res = res + 5;
}
res = res * 2;
```

What will be the final value of the variable `res`? _____

Question 8

There are two values of type `boolean`: _____ and _____.

E.3 PRE-TEST (TOPIC SET 2)

Login_____

Question 1

What is the final value of `result` in the following loop?

```
int i = 0;
int result = 0;
while (i < 5) {
    result = result + i;
    i = i + 1;
}
```

result: _____

Question 2

How many times will the loop from the question 1 execute?

Answer:_____

Question 3

What is the final value of `result` in the following loop?

```
int i = 3;
int result = 0;
do {
    result = result + i;
    i = i + 1;
} while (i <= 5);
```

result: _____

Question 4

How many times does the following loop execute?

```
for (int i = 0; i <= 5; i++) {  
    System.out.println( i * i);  
}
```

Answer: _____

Question 5

In question 4, what is the last line of console output

Answer: _____

Question 6

Based on the statement below, which of the following statements gives the number of elements in the array `data`?

```
double[] data = new double[10];
```

```
data[10]  
data.size()  
data.length  
data.size  
data.length()  
data.last
```

Question 7

Based on the following statement, `primes[3] = _____`.

```
int[] primes = {2, 3, 5, 7, 11};
```

Question 8

What would be the output of the following code fragment:

```
ArrayList<Double> list = new ArrayList<Double>();  
list.add(2.5);  
list.add(3.0);  
list.add(3.5);  
list.remove(1);  
for(int i = 0; i < list.size(); i++)  
    System.out.println(list.get(i));
```

Output:

E.4 POST-TEST (TOPIC SET 2)

Login _____

Question 1

What is the final value of `result` in the following loop?

```
int i = 0;
int result = 0;
while (i <= 5) {
    result = result + 2;
    i = i + 1;
}
```

result: _____

Question 2

How many times will the loop from the question 1 execute?

Answer: _____

Question 3

What is the final value of `result` in the following loop?

```
int i = 5;
int result = 0;
do {
    result = result + i;
    i = i - 1;
} while (i > 2);
```

result: _____

Question 4

How many times does the following loop execute?

```
for (int i = 3; i > 0; i--) {  
    System.out.println(i + i);  
}
```

Answer: _____

Question 5

In question 4, what is the last line of console output?

Answer: _____

Question 6

Based on the statement below, which of the following statements gives the number of elements in the array `data`?

```
double[] data = new double[10];
```

```
data[10]  
data.size()  
data.length  
data.size  
data.length()  
data.last
```

Question 7

ARRAYS

Based on the following statement, `primes[4] = _____`.

```
int[] primes = {7, 1, 3, 0, 2};
```

Question 8

What would be the output of the following code fragment:

```
ArrayList<Double> list = new ArrayList<Double>();  
list.add(1.5);  
list.add(4.0);  
list.add(2.5);  
list.remove(2);  
for(int i = 0; i < list.size(); i++)  
    System.out.println(list.get(i));
```

Output:

APPENDIX F

USER FEEDBACK QUESTIONNAIRES

F.1 POST-QUESTIONNAIRE (GROUPS A&C)

Login: _____

In this experiment you interacted with two systems: QuizJET system was delivering the self-assessment questions. The OOPS service was recommending reading materials relevant to QuizJET questions. This questionnaire will evaluate your opinion about the OOPS service.

1) I like the interface of the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

2) I tried to use the help of the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

3) I benefited from the materials recommended by the OOPS service.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

4) The recommendations of the OOPS service did not help me to solve the questions.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

5) Most often, when I clicked on a recommended link, I had to browse to find a helpful page.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

6) Most often, I was not able to find a helpful page even by browsing.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

7) Most often, I did not have to browse, one of the recommended pages was helpful.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

8) Most often, the helpful page was the first or second in the recommended list.

In Session 1:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

In Session 2:

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

9) The adaptive annotations of the recommended links helped me to choose appropriate recommendations.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

10) I did not use the adaptive annotations of the recommended when choosing the recommendations.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

11) I think the idea of recommending reading material to help solve the questions is good, but implementation could be better.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

12) If you have any additional comment on the work of the OOPS service, please provide it here:

F.2 POST-QUESTIONNAIRE (GROUPS B&D)

Login: _____

In the experiment, you interacted with two systems: QuizJET system was delivering the self-assessment questions. The OOPS service was recommending reading materials relevant to QuizJET questions. In one session, you used a regular textbook instead of the OOPS service. This questionnaire will evaluate your opinion about the OOPS service.

1) I like the interface of the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

2) I tried to use the help of the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

3) I benefited from the materials recommended by the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

4) The recommendations of the OOPS service DID NOT help me to solve the questions.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly agree

5) Most often, when I clicked on a recommended link, I had to browse to find a helpful page.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

6) Most often, I was not able to find a helpful page even by browsing.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

7) Most often, I did not have to browse, one of the recommended pages was helpful for the question.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

8) Most often, the helpful page was the first or second in the recommended list.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Agree

9) The adaptive annotations of the recommended links helped me to choose appropriate recommendations.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

10) I did not use the adaptive annotations of the recommended when choosing the recommendations.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

11) I think the idea of recommending reading material to help solve the questions is good, but implementation could be better.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

12) The usual textbook was more helpful for solving questions than the recommendations of the OOPS service.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

13) It was easier to find a helpful page among the recommendations of the OOPS service than in a textbook.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

14) OOPS recommendations were better only because they were accessible within the same system, which was easier to use than the book.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

15) OOPS recommendations were better because OOPS brought the helpful pages much closer, and I did not have to look for them in the entire book.

Strongly Disagree	Disagree	No Strong Opinion	Agree	Strongly Disagree

16) If you have any additional comment on the work of the OOPS service, please provide it here:

BIBLIOGRAPHY

- Anderson, J. R., A. Corbett, et al. (1995). "Cognitive tutors: Lessons learned." The Journal of the Learning Sciences **4**(2): 167-207.
- Apted, T. and J. Kay (2004). "MECUREO Ontology and Modelling Tools." International Journal of Continuing Engineering Education and Lifelong Learning **14**(3): 191-211.
- Apted, T., J. Kay, et al. (2003). Visualisation of learning ontologies. 11th International Conference on Artificial Intelligence in Education (AIED'2003), IOS Press.
- Apted, T., J. Kay, et al. (2004). Supporting Metadata Creation with an Ontology Built from an Extensible Dictionary. 3d International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2004), Eindhoven, The Netherlands, Springer.
- Apted, T., J. Kay, et al. (2003). Visualisation of ontological inferences for user control of personal Web agents. Seventh International Conference on Information Visualization.
- Ashburner, M., C. A. Ball, et al. (2000). "Gene Ontology: tool for the unification of biology." Nature Genetics **25**(1): 25-29.
- Bateman, S., C. Brooks, et al. (2006). Collaborative Tagging Approaches for Ontological Metadata in Adaptive E-Learning Systems. Workshop on Applications of Semantic Web Technologies for e-Learning (SW-EL) at AH'06. M. Dzbor, D. Dicheva and L. Aroyo. Dublin, Ireland.
- Bateman, S., R. Farzan, et al. (2006). OATS: The Open Annotation and Tagging System. Third Annual International Scientific Conference of the Learning Object Repository Research Network, Montreal, Canada.
- Bechhofer, S., C. Goble, et al. (2003). COHSE: Conceptual Open Hypermedia Service. Annotation for the Semantic Web. S. Handschuh and S. Staab, IOS Press: 193-211.
- Billsus, D. and M. J. Pazzani (1999). A hybrid user model for news story classification. 7th International Conference on User Modeling (UM'1999), Banff, Canada, Springer-Verlag, New York, Inc.
- Bloom, B. S. E. (1956). Taxonomy of Educational Objectives, the Classification of Educational Goals - Handbook I: Cognitive Domain. New York, Longmans, Green.

- Brickley, D. and L. Miller. (2005). "FOAF: Friend of a Friend." from <http://www.foaf-project.org>.
- Brusilovsky, P. (2001). WebEx: Learning from examples in a programming course. Proceedings of WebNet'2001, World Conference of the WWW and Internet, 2001. W. Fowler and J. Hasebrook. Orlando, FL, AACE: 124-129.
- Brusilovsky, P. (2004). KnowledgeTree: A distributed architecture for adaptive e-learning. 13th International World Wide Web Conference, WWW 2004 (Alternate track papers and posters), New York, NY, ACM Press.
- Brusilovsky, P. (2004). KnowledgeTree: A distributed architecture for adaptive elearning. 13th International World Wide Web Conference. New York, NY: 104-113.
- Brusilovsky, P. (2007). Adaptive navigation support. The Adaptive Web: Methods and Strategies of Web Personalization. P. Brusilovsky, A. Kobsa and W. Nejdl. Heidelberg, Germany, Springer Verlag: 263-290.
- Brusilovsky, P., G. Chavan, et al. (2004). Social adaptive navigation support for open corpus electronic textbooks. Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2004), Eindhoven, the Netherlands.
- Brusilovsky, P. and N. Henze (2007). Open Corpus Adaptive Educational Hypermedia. The Adaptive Web: Methods and Strategies of Web Personalization. P. Brusilovsky, A. Kobsa and W. Nejdl. Heidelberg, Germany, Springer Verlag: 671-696.
- Brusilovsky, P. and C. Peylo (2003). "Adaptive and intelligent Web-based educational systems." International Journal of Artificial Intelligence in Education **13**(2-4): 159-172.
- Brusilovsky, P. and S. Sosnovsky (2005). Engaging students to work with self-assessment questions: A study of two approaches. 10th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'2005). Monte de Caparica, Portugal, ACM Press: 251-255.
- Brusilovsky, P. and S. Sosnovsky (2005). "Individualized exercises for self-assessment of programming knowledge: An evaluation of QuizPACK." Journal on Educational Resources in Computing **5**(3): 6.1 - 6.22.
- Brusilovsky, P., S. Sosnovsky, et al. (2004). QuizGuide: Increasing the Educational Value of Individualized Self-Assessment Quizzes with Adaptive Navigation Support. E-Learn 2004. J. Nall and R. Robson. Washington, DC, USA, AACE: 1806-1813.
- Brusilovsky, P., S. Sosnovsky, et al. (2009). "Addictive links: the motivational value of adaptive link annotation." New Review of Hypermedia and Multimedia: Special issue on Adaptive Hypermedia **15**(1): 97-118.
- Brusilovsky, P., S. Sosnovsky, et al. (2005). Interactive Authoring Support for Adaptive Educational Systems. 12th International Conference on Artificial Intelligence in

- Education, AIED'2005. C.-K. Looi, G. McCalla, B. B. and B. J. Amsterdam, Netherlands, Amsterdam: IOS Press: 96-103.
- Brusilovsky, P., S. Sosnovsky, et al. (2008). User Model Integration in a Distributed Adaptive E-Learning System. Proceedings of the International Workshop on User Model Integration at AH'2008, Hannover, Germany. S. Berkovsky, F. Carmagnola, D. Heckmann, A. Krueger and T. Kuflik. Hannover, Germany: 1-10.
- Brusilovsky, P. and C. Tasso (2004). "Preface to Special Issue on User Modeling for Web Information Retrieval." User Modeling and User-Adapted Interaction **14**(2-3): 147-157.
- Bull, S., V. Dimitrova, et al. (2007). "Open Learner Models: Research Questions (Special Issue of IJAIED Part 1)." International Journal of Artificial Intelligence in Education **17**(2): 83-87.
- Carbonell, J. R. (1970). "AI in CAI: An Artificial Intelligence Approach to Computer-Assisted Instruction." IEEE Transactions on Man-Machine Systems **11**(4): 190-202.
- Carmagnola, F. and V. Dimitrova (2008). An Evidence-Based Approach to Handle Semantic Heterogeneity in Interoperable Distributed User Models. 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems. W. Nejdl, J. Kay, P. Pu and E. Herder. Hannover, Germany: 73-83.
- Carmagnola, F. and V. Dimitrova (2008). An Evidence-Based Approach to Handle Semantic Heterogeneity in Interoperable Distributed User Models. 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2008), Hannover, Germany, Springer Verlag.
- Carmona, C., D. Bueno, et al. (2006). SIGUE: Making Web Courses Adaptive. Adaptive Hypermedia and Adaptive Web-Based Systems. B. De, P. Brusilovsky and R. Conejo, Springer Berlin / Heidelberg. **2347**: 376-379.
- Carr, B. and I. Goldstein (1977). Overlays: A Theory of Modelling for Computer Aided Instruction, Massachusetts Institute of Technology, Cambridge AI Lab: 24.
- Carr, L., D. DeRoure, et al. (1995). "The distributed link service: A tool for publishers, authors and readers." World Wide Web Journal **1**(1): 647-656.
- Chen, L. and K. Sycara (1998). A Personal Agent for Browsing and Searching. 2nd International Conference on Autonomous Agents, Minneapolis/St. Paul.
- Chen, W. and R. Mizoguchi (1999). Communication content ontology for learner model agent in multiagent architecture. International Conference on Advanced Research in Computers and Communications in Education (ICCE'99).
- Cimolino, L., J. Kay, et al. (2004). "Concept mapping for eliciting verified personal ontologies." International Journal of Continuing Engineering Education and Lifelong Learning **14**(3): 212 - 228.

- Cowie, J. and W. Lehnert (1996). "Information extraction." Commun. ACM **39**(1): 80-91.
- DCMI. (1999). "Dublin Core Metadata Initiative." from <http://dublincore.org/>.
- Denaux, R., L. Aroyo, et al. (2005). An approach for ontology-based elicitation of user models to enable personalization on the semantic web. 14th international conference on World Wide Web (WWW '05), Chiba, Japan, ACM, New York, NY, USA.
- Denaux, R., L. Aroyo, et al. (2005). OWL-OLM: Interactive Ontology-based Elicitation of User Models. Workshop on Personalisation for the Semantic Web (PerSWeb'05) at UM'2005. Edinburgh, UK.
- Denaux, R., V. Dimitrova, et al. (2005). Integrating Open User Modeling and Learning Content Management for the Semantic Web. 10th International Conference on User Modeling (UM'2005), Edinburgh, Scotland, UK, Springer.
- Dicheva, D., S. Sosnovsky, et al. (2005). Ontological Web Portal for Educational Ontologies. Workshop on Applications of Semantic Web Technologies for e-Learning at AIED'2005. Amsterdam, Netherlands.
- Dieberger, A. and M. Guzdial (2002). CoWeb - Experiences with Collaborative Web Spaces. From Usenet to CoWebs: Interacting with Social Information Spaces. C. Lueg and D. Fisher, Springer Verlag.
- Ding, L., T. Finin, et al. (2004). Swoogle: a search and metadata engine for the semantic web. Thirteenth ACM International Conference on Information and Knowledge Management (CIKM '04), Washington, D.C., USA, ACM Press, New York, NY, USA.
- Doan, A., J. Madhavan, et al. (2002). Learning to map between ontologies on the semantic web. 11th International World Wide Web Conference, New York, NY, USA, ACM Press.
- Dolog, P. and W. Nejdl (2003). Challenges and Benefits of the Semantic Web for User Modelling. AH2003 workshop. P. De Bra. Budapest, Hungary.
- Dolog, P. and W. Nejdl (2007). Semantic Web Technologies for the Adaptive Web. The Adaptive Web: Methods and Strategies of Web Personalization. P. Brusilovsky, A. Kobsa and W. Neidl. Berlin Heidelberg New York, Springer-Verlag. **4321**: 697-719.
- Domingue, J., M. Dzbor, et al. (2004). Magpie: supporting browsing and navigation on the semantic web. 9th International Conference on Intelligent User Interfaces (IUI '04), Funchal, Madeira, Portugal, ACM Press, New York, NY, USA.
- Downes, S. (2001). "Learning Objects: Resources for distance education worldwide." The International Review of Research in Open and Distance Learning **2**(1): Article 2.1.6.
- Dzbor, M. and E. Motta (2007). Semantic Web Technologies to Support Learning about the Semantic Web. 13th International Conference on Artificial Intelligence in Education (AIED'2007), Marina Del Ray, CA, USA, IOS Press.

- Euzenat, J. and P. Shvaiko (2007). Ontology matching. Berlin Heidelberg, Germany, Springer-Verlag.
- Forsythe, D. E. and B. G. Buchanan (1989). "Knowledge acquisition for expert systems: some pitfalls and suggestions." IEEE Transactions on Systems, Man and Cybernetics **19**(3): 435-442.
- Gangemi, A., N. Guarino, et al. (2002). Sweetening Ontologies with DOLCE. 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web (EKAW'2002), Siguenza, Spain, Springer Verlag.
- Gomez-Perez, A. and D. Manzano-Macho (2005). "An overview of methods and tools for ontology learning from texts." The Knowledge Engineering Review **19**(3): 187-212.
- Gruber, T. R. (1995). "Toward Principles for the Design of Ontologies Used for Knowledge Sharing." International Journal of Human and Computer Studies **43**(5-6): 907-928.
- Guarino, N. (1998). Formal Ontology and Information Systems. 1st International Conference on Formal Ontologies in Information Systems (FOIS'98), Trento, Italy.
- Heckmann, D., T. Schwartz, et al. (2005). Gumo - The General User Model Ontology. 10th International User Modeling Conference, Edinburgh, UK, Springer Verlag.
- Henze, N., P. Dolog, et al. (2004). "Reasoning and Ontologies for Personalized e-Learning in the Semantic Web." Educational Technology & Society **7**(4): 82-97.
- Henze, N. and W. Nejdl (2001). "Adaptation in open corpus hypermedia." International Journal of Artificial Intelligence in Education **12**(4): 325-350.
- Henze, N. and W. Nejdl (2002). Knowledge Modeling for Open Adaptive Hypermedia. 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002), Malaga, Spain, Springer.
- Horstmann, C. (2007). Big Java, John Wiley and Sons, Inc.
- Hovland, C. I., A. A. Lumsdaine, et al. (1949). A baseline for measurement of percentage change. Experiments on mass communication. C. I. Hovland, A. A. Lumsdaine and F. D. Sheffield, Wiley: 77-82.
- Hovy, E. (1998). Combining and standardizing largescale, practical ontologies for machine translation and other uses. First International Conference on Language Resources and Evaluation (LREC'1998), Granada, Spain.
- Hsiao, I.-H., S. Sosnovsky, et al. (2010). "Guiding Students to the Right Questions: Adaptive Navigation Support in an E-learning System for Java Programming." Journal of Computer Assisted Learning **26**(4): 270-283.

- Hsiao, S., P. Brusilovsky, et al. (2008). Web-based Parameterized Questions for Object-Oriented Programming. E-Learn 2008, November 17-21. Las Vegas, NV, USA, AACE: 3728-3735.
- Huhns, M. N. and L. M. Stephens (1999). "Personal ontologies." IEEE Internet Computing **3**(5): 85-87.
- IEEE-LTSC (2001). Public and Private Information (PAPI) for Learners (PAPI Learner), IEEE, Inc.
- IMS (2001). Learner Information Package Specification, IMS/GLC, Inc.
- Jovanović, J., D. Gasevic, et al. (2006). "Ontology-based automatic annotation of learning content." International Journal on Semantic Web and Information Systems **2**(2): 91-119.
- Jovanovic, J., D. Gazevic, et al. (2006). Automating Semantic Annotation to Enable Learning Content Adaptation. 4th International Conference on Adaptive Hypermedia and Adaptive Web Systems (AH'2006), Dublin, Ireland, Springer.
- Kalfoglou, Y., J. Domingue, et al. (2001). MyPlanet: an ontology-driven Web-based personalised news service. Workshop on Ontologies and Information Sharing at IJCAI'01. Seattle, WA, USA: 140-148.
- Kalfoglou, Y. and M. Schorelmmmer (2003). "Ontology Mapping: the State of the Art." The Knowledge Engineering Review **18**(1): 1-31.
- Kay, J., B. Kummerfeld, et al. (2002). Personis: A server for user modeling. Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002), Málaga, Spain.
- Kay, J., R. J. Kummerfeld, et al. (2002). Personis: a Server for User Models. 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002), Malaga, Spain, Springer.
- Klamma, R., M. A. Chatti, et al. (2005). "LIP2PAPI Converter." from <http://dbis.rwth-aachen.de/lehrstuhl/staff/chatti/LIP2PAPICConverter/index.html>.
- Kobsa, A. and J. Fink (2006). "An LDAP-based User Modeling Server and its Evaluation." User Modeling and User-Adapted Interaction **16**(2): 129-169.
- Konstan, J. A., B. N. Miller, et al. (1997). "GroupLens: applying collaborative filtering to Usenet news." Communications of the ACM **40**(3): 77-87.
- Kuhn, H. W. (1955). "The Hungarian Method for the assignment problem." Naval Research Logistics Quarterly **2**: 83-97.
- Kumar, A. (2006). "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 **3**: in press.

- Lenat, D. B. (1995). "CYC: a large-scale investment in knowledge infrastructure." Communications of ACM, ACM Press, New York, NY, USA **38**(11): 33-38.
- Levenshtein, I. V. (1966). "Binary Codes capable of correcting deletions, insertions, and reversals." Cybernetics and Control Theory **10**(8): 707-710.
- Li, Y. and N. Zhong (2006). "Mining ontology for automatically acquiring Web user information needs." IEEE Transactions on Knowledge and Data Engineering **18**(4): 554-568.
- Lieberman, H. (1995). Letizia: An agent that assists web browsing. International Joint Conference on Artificial Intelligence, Montreal, Canada.
- Maedche, A. and S. Staab (2001). "Ontology learning for the Semantic Web." IEEE Intelligent Systems and Their Applications **16**(2): 72-79.
- Maedche, A. and S. Staab (2002). Measuring Similarity between Ontologies. 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web Springer-Verlag.
- Maedche, A. and S. Staab (2002). Measuring Similarity between Ontologies. 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web Springer-Verlag: 251-263.
- Melis, E., G. Gogvadze, et al. (2006). "Semantic-Aware Components and Services of ActiveMath." British Journal of Educational Technology **37**(3): 405--423.
- Micarelli, A., F. Gasparetti, et al. (2007). Personalized Search on the World Wide Web. The Adaptive Web: Methods and Strategies of Web Personalization. P. Brusilovsky, A. Kobsa and W. Nejdl. Heidelberg, Germany, Springer Verlag: 195-230.
- Middleton, S. E., D. C. De Roure, et al. (2001). Capturing knowledge of user preferences: ontologies in recommender systems. 1st International Conference on Knowledge Capture (K-CAP '01), Victoria, British Columbia, Canada, ACM Press, New York, NY, USA.
- Middleton, S. E., N. R. Shadbolt, et al. (2003). Capturing interest through inference and visualization: ontological user profiling in recommender systems. 2nd International Conference on Knowledge Capture (K-CAP '03), Sanibel Island, FL, USA, ACM Press, New York, NY, USA.
- Miller, G. A., R. Beckwith, et al. (1990). "Introduction to WordNet: An On-line Lexical Database." International Journal of Lexicography, Oxford University Press **3**(4): 235-244.
- Mitrovic, A. and V. Devedzic (2004). "A Model of Multitutor Ontology-based Learning Environments." Continuing Engineering Education and Life-Long Learning **14**(3): 229-245.

- Mitrovic, A., B. Martin, et al. (2007). "Intelligent tutors for all: Constraint-based modeling methodology, systems and authoring." IEEE Intelligent Systems, special issue on Intelligent Educational Systems **22**(4): 38-45.
- Mitrovic, A., M. Mayo, et al. (2001). Constraint-Based Tutors: A Success Story 14th Int. Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA/AIE-2001, Budapest, Springer-Verlag.
- Mizoguchi, R. and J. Bourdeau (2000). "Using Ontological Engineering to Overcome AI-ED Problems." International Journal of Artificial Intelligence in Education **11**(2): 107-121.
- Mizoguchi, R., K. Sinitsa, et al. (1996). Task Ontology Design for Intelligent Educational/Training Systems. Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs at ITS'1996, Montreal, Canada.
- Murray, T. (1998). "Authoring knowledge-based tutors: tools for content, instructional strategy, student model, and interface design." Journal of Learning Sciences **7**(1): 5-64.
- Murray, T. (1999). "Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art." International Journal of Artificial Intelligence in Education **10**: 98-129.
- Newell, A. and P. S. Rosenbloom (1981). Mechanisms of skill acquisition and the law of practice. Cognitive skills and their acquisition. J. R. Anderson. Hillsdale, NJ, Lawrence Erlbaum Associates, Inc.: 1-51.
- Niederée, C., A. Stewart, et al. (2004). A Multi-Dimensional, Unified User Model for Cross-System Personalization. Workshop on Environments for Personalized Information Access at AVI'2004. Gallipoli, Italy: 34-54.
- Niles, I. and A. Pease (2001). Towards a standard upper ontology. FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems, Ogunquit, Maine, USA, ACM Press.
- Novak, J. D. and D. B. Gowin (1984). Learning how to learn. New York, NY, Cambridge University Press.
- Noy, N. F. (2004). "Semantic integration: a survey of ontology-based approaches." SIGMOD Record **33**(4): 65-70.
- Noy, N. F. and M. A. Musen (2003). "The PROMPT suite: Interactive tools for ontology merging and mapping." International Journal of Human-Computer Studies **59**(6): 983-1024.
- Oracle. (2011, 07/20/2011). "The Java Tutorials." Retrieved 09/01/2011, 2011, from <http://download.oracle.com/javase/tutorial/>.
- Ounnas, A., I. Liccardi, et al. (2006). Towards a Semantic Modeling of Learners for Social Networks. International Workshop on Applications of Semantic Web Technologies for E-Learning at AH'2006. L. Aroyo and D. Dicheva. Dublin, Ireland.

- Pearl, J. (1988). Probabilistic reasoning in intelligent systems: networks of plausible inference. San Mateo, CA, Morgan Kaufmann.
- Pérez-Marín, D., E. Alfonseca, et al. (2006). Automatic Generation of Students' Conceptual Models Underpinned by Free-Text Adaptive Computer Assisted Assessment. International Conference on Advanced Learning Technologies (ICALT'2006), Kerkrade, The Netherlands.
- Pérez-Marín, D., E. Alfonseca, et al. (2007). Automatic Generation of Students' Conceptual Models from Answers in Plain Text. 11 International Conference on User Modeling (UM'2007), Corfu, Greece, Springer Berlin / Heidelberg.
- Pérez-Marín, D., I. Pascual-Nieto, et al. (2006). Automatic Identification of Terms for the Generation of Students Concept Maps. International Conference on Multimedia and Information Technologies for the Education (MICTE'2006).
- Popov, B., A. Kiryakov, et al. (2004). "KIM – a semantic platform for information extraction and retrieval." Natural Language Engineering **10**: 375-392.
- Porter, M. F. (1980). "An algorithm for suffix stripping." Program **14**(3): 130–137.
- Rich, E. (1979). "User Modeling via Stereotypes." Cognitive Science: A Multidisciplinary Journal **3**(4): 329-354.
- Roy, D., S. Sarkar, et al. (2008). "Automatic Extraction of Pedagogic Metadata from Learning Content." International Journal of Artificial Intelligence in Education **18**(2): 97-118.
- Salton, G., A. Wong, et al. (1975). "A vector space model for automatic indexing." Commun. ACM **18**(11): 613-620.
- Shamsfard, M. and A. A. Barforoush (2003). "The state of the art in ontology learning: a framework for comparison." The Knowledge Engineering Review **18**(4): 293-316.
- Sicilia, M.-A., E. Garcia, et al. (2004). "Using links to describe imprecise relationships in educational contents." International Journal of Continuing Engineering Education and Lifelong Learning **14**(3): 260 - 275.
- Sicilia, M. A., E. García, et al. (2002). LEARNING LINKS: Reusable Assets with Support for Vagueness and Ontology-based Typing. Workshop on Concepts and Ontologies in Web-based Educational Systems at ICCE'2002. L. Aroyo and D. Dicheva. Auckland, New Zealand: 37-42.
- Smith, A. S. G. and A. Blandford (2003). "MLTutor: An application of machine learning algorithms for an adaptive Web-based information system." International Journal of Artificial Intelligence in Education **13**(2-4): 235-261.

- Sosnovsky, S. and P. Brusilovsky (2005). Layered Evaluation of Topic-Based Adaptation to Student Knowledge. Workshop on Evaluation of Adaptive Systems at UM'2005. Eduburgh, UK: 47-56.
- Sosnovsky, S., P. Brusilovsky, et al. (2008). Re-assessing the Value of Adaptive Navigation Support in E-Learning Context. 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2008). W. Nejdl, J. Kay, P. Pu and E. Herder. Hannover, Germany, Springer Berlin / Heidelberg: 193-203.
- Sosnovsky, S., P. Brusilovsky, et al. (2004). Supporting adaptive hypermedia authors with automated content indexing. Proceeding of 2nd Workshop on Authoring Adaptive and Adaptable Educational Hypermedia, Eindhoven, the Netherlands, August 23, 2004: 380-389.
- Sosnovsky, S., P. Brusilovsky, et al. (2009). Semantic Integration of Adaptive Educational Systems. Advances in Ubiquitous User Modelling. T. Kuflik, Berkovsky, S., Carmagnola, F., Heckmann D., & Krüger, A. Berlin Heidelberg, Springer-Verlag: 134–158.
- Sosnovsky, S., P. Dolog, et al. (2007). Translation of Overlay Models of Student Knowledge for Relative Domains Based on Domain Ontology Mapping. 13th International Conference on Artificial Intelligence in Education (AIED'2007). R. Luckin, K. R. Koedinger and J. Greer. Marina Del Ray, CA, USA, IOS Press. **158**: 289-296.
- Sosnovsky, S., A. Mitrovic, et al. (2008). Ontology-based integration of adaptive educational systems. Proceedings of 16th International Conference on Computers in Education (ICCE'2008), Taipei, Taiwan, October 27-31, 2008: 11-18.
- Sosnovsky, S., A. Mitrovic, et al. (2008). Towards Integration of Adaptive Educational Systems: Mapping Domain Models to Ontologies. 6th International Workshop on Ontologies and Semantic Web for E-Learning (SWEL'2008) at ITS'2008. D. Dicheva, A. Harrer and R. Mizoguchi. Montreal, Canada.
- Sosnovsky, S., M. Yudelson, et al. (2007). Community-oriented Course Authoring to Support Topic-based Student Modeling. Fifth International Workshop on Ontologies and Semantic Web for E-Learning at AIED'2007. D. Dicheva, R. Mizoguchi, N. Capuano and A. Harrer. Marina Del Ray, CA, USA: 91-100.
- Stumme, G. and A. Madche (2001). FCA-Merge: Bottom-up merging of ontologies. 7th International Conference on Artificial Intelligence (IJCAI '01), Seattle, WA, USA.
- Suraweera, P., A. Mitrovic, et al. (2004). The role of domain ontology in knowledge acquisition for ITSs. 7th International Conference on Intelligent Tutoring Systems (ITS'2004), Maceio, Brazil, Springer-Verlag.
- Tang, T. and G. McCalla (2003). Smart Recommendation for an Evolving E-Learning System. Workshop on Technologies for Electronic Documents for Supporting Learning at AIED'2003. Sydney, Australia.

- Trella, M., C. Carmona, et al. (2005). MEDEA: an Open Service-Based Learning Platform for Developing Intelligent Educational Systems for the Web. Workshop on Adaptive Systems for Web-Based Education: Tools and Reusability at AIED'05, Amsterdam, The Netherlands.
- Uther, J. and J. Kay (2003). VIUM, a Web-Based Visualization of Large User Models. 9th International Conference on User Modeling (UM'2003), Johnstown, PA, USA, Springer.
- van der Sluijs, K. and G.-J. Houben (2009). Automatic Generation of Semantic Metadata as Basis for User Modeling and Adaptation. Advances in Ubiquitous User Modelling. T. Kuflik, S. Berkovsky, F. Carmagnola, D. Heckmann and A. Krüger, Springer Berlin / Heidelberg. **5830**: 73-93.
- W3C (2001). Representing vCard Objects in RDF/XML.
- W3C. (2001). "Semantic Web Activity." from <http://www.w3.org/2001/sw/>.
- W3C. (2007). "Semantic Web Health Care and Life Sciences Interest Group (HCLSIG)." from <http://esw.w3.org/topic/SemanticWebForLifeSciences>.
- Winter, M., C. Brooks, et al. (2005). Towards Best Practices for Semantic Web Student Modelling. 12th International Conference on Artificial Intelligence in Education (AIED'2005), IOS Press.
- Woods, W. A. (1975). What's in a link: foundations for semantic networks. Representation and Understanding. B. D. G. and A. Collins. New York, Academic Press: 35-82.
- Yesilada, Y., S. Bechhofer, et al. (2007). COHSE: Dynamic Linking of Web Resources, Sun Microsystems.
- Yudelson, M. and P. Brusilovsky (2005). NavEx: Providing Navigation Support for Adaptive Browsing of Annotated Code Examples. 12th International Conference on Artificial Intelligence in Education, AIED 2005. C.-K. Looi, G. McCalla, B. Bredeweg and J. Breuker. Amsterdam, IOS Press: 710-717.
- Yudelson, M., P. Brusilovsky, et al. (2007). A User Modeling Server for Contemporary Adaptive Hypermedia: An Evaluation of Push Approach to Evidence Propagation. 11th International Conference on User Modeling, UM 2007, Corfu, Greece, Springer Verlag.
- Yudelson, M., P. Brusilovsky, et al. (2007). A User Modeling Server for Contemporary Adaptive Hypermedia: An Evaluation of Push Approach to Evidence Propagation. 11th International Conference on User Modeling, UM 2007. C. Conati, K. McCoy and G. Paliouras. Corfu, Greece, Springer Verlag: 27-36.
- Zaiane, O. R. (2002). Building a recommender agent for e-learning systems. International Conference on Computers in Education.