Integrating Open User Modeling and Learning Content Management for the Semantic Web

Ronald Denaux[1], Vania Dimitrova[2], Lora Aroyo[1]

1 Computer Science Department, Eindhoven Univ. of Technology, The Netherlands
2 School of Computing, University of Leeds, UK
`r.o.denaux@student.tue.nl, l.m.aroyo@tue.nl, vania@comp.leeds.ac.uk`

Abstract. The paper describes an ontology-based approach for integrating interactive user modeling and learning content management to deal with typical adaptation problems, such as cold start and dynamics of the user's knowledge, in the context of the Semantic Web. An integrated OntoAIMS system is presented and its viability discussed based on user studies. The work demonstrates some novel aspects, such as (a) ontological approach for integration of methods for eliciting and utilizing of user models; (b) improved adaptation functionality resulted from that integration, validated with real users; (c) support of interoperability and reusability of adaptive components.

1 Introduction

A key factor for the successful implementation of the Semantic Web vision [2] is the ability to deal with the *diversity of users* (who differ in their capabilities, expectations, goals, requirements, and preferences) and to provide *personalized access* and *user-adapted services*. Recently, the Semantic Web community is acknowledging the need to consider the user's perspective to provide personalization functionality [8]. Personalization has been a prime concern of the user-modeling community which has developed methods for building user models (UMs) and using these models to tailor the system's behavior to the needs of individuals. However, for UM methods to be deployed on the Semantic Web, they should deal with semantics defined with *ontologies* [3], and should enable *interoperability* of algorithms that elicit and utilize UMs[8] based on common ontology specification languages, for example OWL [10].

We present here how interactive user modeling (UM elicitation) and adaptive content management (UM utilization) on the Semantic Web can be integrated in a *learning domain* to deal with typical adaptation problems, such as cold start, inaccuracy of assumptions about a user's cognitive state drawn only on the basis of interaction tracking data, and dynamics of the student's knowledge. The paper demonstrates the following novel aspects: (a) ontological approach for integration of methods for eliciting and utilizing user models; (b) improved adaptation functionality resulted from that integration, validated in studies with real users; (c) support of interoperability and reusability on the educational Semantic Web. We illustrate these in an adaptive system called OntoAIMS.

Our work on providing users with a structured way to search, browse and access large repositories of learning resources on the Semantic Web relates to research on adaptive *Learning Content Management Systems* (LCMS). Similarly to existing LCMS, e.g. [4, 12], OntoAIMS employs student models to allocate tasks and resources for individual students. Distinctively, we use OWL ontologies to represent the domain and the UM. The latter extends the notion of a domain overlay by including students' beliefs that are not necessarily in the system's domain knowledge. In contrast with more general UMs for the Web, e.g. Hera [11], which consider merely user attributes, OntoAIMS uses an enhanced, interoperable, ontology-based user model built via a dialog with the user.

The interactive user modeling component in OntoAIMS, called OWL-OLM, elicits an *OWL-based open learner model* built with the active user's participation. This extends an interactive open learner modeling framework [7] to deal with dynamic, ontology-based, advanced learner models [6]. Similarly to [9, 13], the open user modeling approach in OWL-OLM deals with a user's conceptual state. Distinctively, we exploit OWL-reasoning to maintain diagnostic interactions and to extract an enhanced user model represented in OWL. This shows a novel open user modeling approach that deals with important issues of modeling users to enable personalization and adaptation for the Semantic Web.

The focus of this paper is the *integration* and the *benefits* of both *interactive user modeling* and *adaptive task recommendation.* We first introduce the integrated architecture of OntoAIMS (Sect. 2) and then briefly describe its main components: ontology-based user modeling (Sect. 3) and task recommendation and resource browsing (Sect. 4). Section 5 discusses how users accept the integrated environment. Finally, we conclude and sketch out future work.

2 Integrated OntoAIMS Architecture

OntoAIMS[1] is an Ontology-based version of the AIMS Adaptive Information Management System [1] providing an information searching and browsing environment that recommends to learners the most appropriate (for their current knowledge) task to work on and aids them to explore domain concepts and read resources related to the task. Currently, OntoAIMS works in a Linux domain.

OntoAIMS uses ontologies (see Fig. 1) to represent the aspects of the application semantics, to allow a strict separation of domain-dependent data, application-related data and resources, and to further enable reusability and sharing of data on the Semantic Web. The learning material is specified in terms of a *Resource Model* that describes the documents in the resource repository and is linked to the *Domain Ontology* which represents the domain concepts and their relationships. The course structure is represented as a hierarchy of tasks in a *Course Task Model.* To enable adaptivity, OntoAIMS utilizes a *User Model* that covers learner preferences, personal characteristics, goals and domain understanding.

[1] The system is available with username *visitor* and password *visitor* at `http://swale.comp.leeds.ac.uk:8080/staims/viewer.html`.

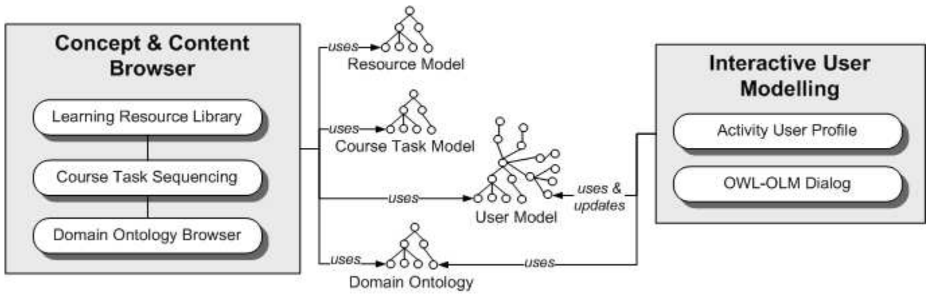


Fig. 1. OntoAIMS Integrated Architecture

The success of adaptation in OntoAIMS depends on the accuracy and completeness of the user model. Unobtrusive ways to collect user data have been considered. The user's interaction with the system (e.g. tasks/resources chosen and searches performed) is gathered in the *Activity User Profile*, which provides information about user preferences and personal characteristics. However, this data is insufficient for modeling a user's domain understanding (called here *conceptual state*, see Sec. 3.1), moreover, such data is unavailable when the user logs for a first time. Hence, to build a model of the user's conceptual state, OntoAIMS employs an interactive UM component that maintains a dialog to elicit a user's conceptual model, see Sect. 3.2. Both the User Model and the Course Task Model are used for recommending the learner a task to study, so that he can navigate efficiently through the course structure, while the Resource Model is used to allocate resources and rank them according to the appropriateness to the learning task, see Sect. 4.

3 Ontology-based User Modeling in OntoAIMS

Throughout the user interaction, information about concepts visited, searches performed, task status (current, started, finished, not started yet), resources opened, and bookmarks saved is stored in the Activity User Profile. It is useful for deducing user preferences and characteristics, and to make initial assumptions about the user's domain understanding. To have an *in-depth* representation of aspects of the user's domain understanding, OntoAIMS uses a User's Conceptual State. This section will outline how it is represented and maintained.

3.1 User's Conceptual State

The main reason for maintaining a conceptual state is to have an intermediate model that *links* a user's conceptualization to an existing domain ontology. The conceptual state is a model of the user's conceptualization inferred during interactions with the user. To distinguish between a temporary, short-term state that gives a snapshot of a user's knowledge extracted during an interaction session and a long-term state that is built as a result of many interactions with the system, we consider *short term conceptual states* (STCS) and a *long term conceptual state* (LTCS), respectively. The former is a *partial* representation of some aspects of a user's conceptualization and is used as the basis for extracting the latter that forms the User Model in OntoAIMS.

STCS is defined as a triple of URIs pointing to a *Conceptual model*, a set of *Domain ontologies* and a *LTCS*. The *Conceptual model* is specified in OWL, which is well-suited for defining conceptualization and for reasoning upon it. The Conceptual Model resembles an ontology specification, i.e. it defines classes, individuals, and properties, and uses OWL properties to define relationships. By using OWL, concepts in the conceptual state are mapped to the domain ontology.

Fig. 2 shows an extract from a conceptual model. The user has used the concept `Filesystem_node`[2] 12 times[3] - 10 of these cases are supported by the domain ontology and 2 are not. He has also stated 3 times that he knows the concept `Filesystem_node` and once that he does not know it. Fig. 2 also shows that conceptual models keep track of specific relationships between concepts like `Move_file_operation` being a subclass of the concept `Command`. Note that the last relationship has been marked as *used wrongly*, which means that it is not supported by the domain ontology and a *mismatch* between the user's conceptualization and the domain ontology is indicated. Note also that a mismatch only indicates that there is a discrepancy between the conceptual state and the domain ontology, it does not indicate that the relationship is not true.

```
<rdf:Description rdf:about="blo:Filesystem_node">
  <rdfs:comment rdf:datatype="xmls:string">
     Any set of data that has a pathname on the filesystem.</rdfs:comment>
  <rdfs:label>file</rdfs:label>
  <rdf:type rdf:resource="owl:Class"/>
  <aimsUM:times_used rdf:datatype="xmls:long">12</aimsUM:times_used>
  <aimsUM:times_used_correctly rdf:datatype="xmls:long">10</aimsUM:times_used_correctly>
  <aimsUM:times_used_wrongly rdf:datatype="xmls:long">2</aimsUM:times_used_wronlgy>
  <aimsUM:times_affirmed rdf:datatype="xmls:long">3</aimsUM:times_affirmed>
  <aimsUM:times_denied rdf:datatype="xmls:long">1</aimsUM:times_denied>
</rdf:Description>
<rdf:Description rdf:nodeID="A273">
  <rdf:type rdf:resource="rdf:Statement"/>
  <rdf:subject rdf:resource="blo:Move_file_operation"/>
  <rdf:predicate rdf:resource="rdfs:subClassOf"/>
  <rdf:object rdf:resource="blo:Command"/>
  <aimsUM:times_used rdf:datatype="xmls:long">1</aimsUM:times_used>
  <aimsUM:times_used_wrongly rdf:datatype="xmls:long">1</aimsUM:times_used_wrongly>
</rdf:Description>
```

Fig. 2. An extract from a student's Conceptual Model based on a dialog episode from the second study described in Sect. 5

[2] This concept is defined in the domain ontology, which can be found at `http://wwwis.win.tue.nl/~swale/blo`

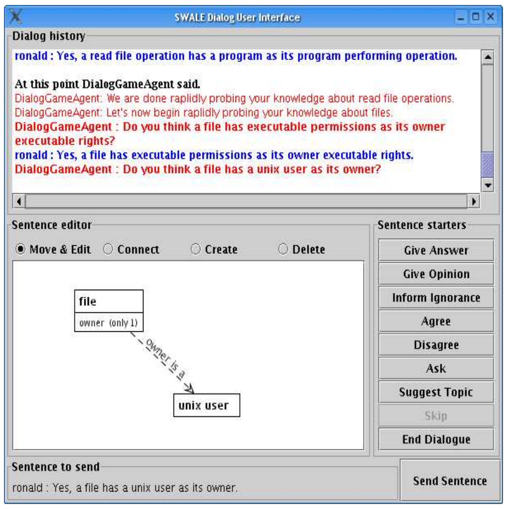
[3] The RDF specification of the properties used to annotate conceptual states can be found at `http://wwwis.win.tue.nl/~swale/aimsUM`

STCS is used to update LTCS which associates a *belief value* for each domain concept. The belief value is calculated based on the conceptual model. The first time the concept is used its belief value is assigned to 50 (out of 100). From then on, $Belief\_value(c) = x + (100 - x)/2$, if there is evidence for knowing c and $Belief\_value(c) = x/2$, if there is evidence for not knowing c, where x is the current belief value of c and evidence is calculated depending on the situation, e.g. `times_used_correctly/times_used`.

3.2 Dialog for Extracting a User's Conceptual Model

OntoAIMS uses OWL-OLM – an OWL-based framework for Open Learner Modeling to (a) validate the analysis of the user data, (b) elicit a learner's conceptual model, and (c) build and maintain a dynamic UM. OWL-OLM follows the STyLE-OLM framework for interactive open learner modeling [7] but amends it to work with a domain ontology and user model built in OWL. OWL-OLM builds a conceptual model of a user by interacting with him in a graphical manner (see Fig. 3). During these interactions, both OWL-OLM and the user can ask domain-related questions and give their opinions about domain-related sentences. To maintain the dialog, OWL-OLM uses a discourse model and infers knowledge from the domain ontology and from the current conceptual model. Because OWL is used as the representation language, OWL-OLM can deploy existing OWL reasoners for the Semantic Web, currently, it uses Jena [4].

The OWL-OLM screenshot in Fig. 3 shows the dialog history in the upper-left corner and the last utterance in the graphical area. The user composes utterances by constructing diagrams using basic graphical operations – 'create', 'delete' or 'edit' a concept or a relation between concepts – and selecting a sentence opener to define his intention, e.g. to 'answer', 'question', 'agree', 'disagree', 'suggest topic'. For a detailed description of OWL-OLM see [6, 5].



ronald : Yes, a read file operation has a program as its program performing operation.

At this point DialogGameAgent said.
DialogGameAgent: We are done raplidly probing your knowledge about read file operations.
DialogGameAgent: Let's now begin raplidly probing your knowledge about files.
DialogGameAgent : Do you think a file has executable permissions as its owner executable rights?
ronald : Yes, a file has executable permissions as its owner executable rights.
DialogGameAgent : Do you think a file has a unix user as its owner?

ronald : Yes, a file has a unix user as its owner.

Fig. 3. Graphical User Interface of OWL-OLM

OWL-OLM analyzes each user utterance to determine how to update the user's conceptual model based on

[4] http://jena.sourceforge.net/

the domain concepts and relations used in the utterance. It determines whether the user's statement is supported by the domain ontology, and, if this is not the case, marks a mismatch. Reasoning over the domain ontology and the conceptual model is also used to determine how OWL-OLM continues the dialog, e.g. asking the user a question, initiating a clarification dialog to discuss a mismatch, or answering a user's question.

Currently, OntoAIMS calls OWL-OLM to probe about a user's domain understanding, and, based on the task model, specifies what domain aspects have to be discussed. The dialog can be terminated either by the user whenever he wishes or by OWL-OLM when the required aspects of the user's conceptualization have been covered. OWL-OLM then uses the extracted STCS to update the belief values in the LTCS that is used by the task recommendation in OntoAIMS.

4 Task Recommendation and Resource Browsing

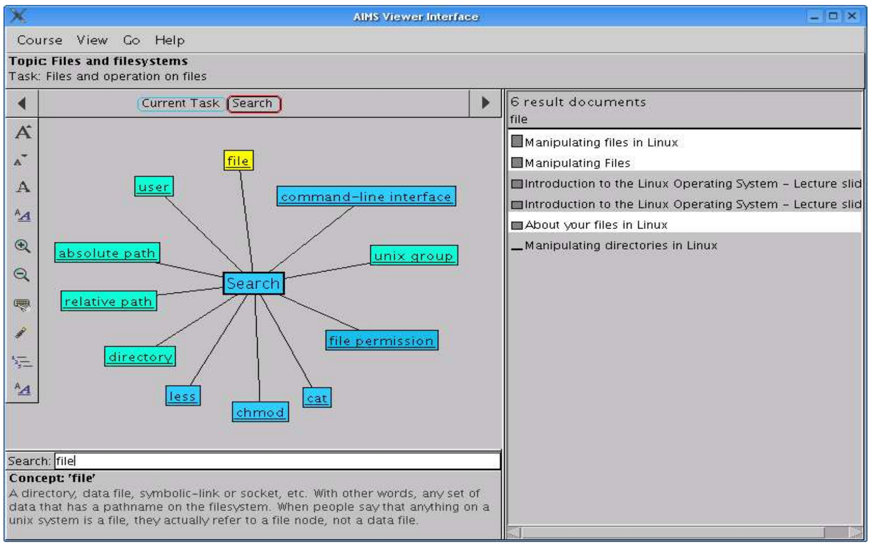
The OntoAIMS Course Task Model consists of a hierarchy of tasks. An example extract from a simplified representation of the Course Task Model used in the current instantiation of OntoAIMS is given below.

```
Course: Introduction to Linux
T1. Introduction
  T1.1 Operating Systems
    T1.1.1 Definition and Description; concepts={operating system, kernel, system program,...}
    ...
  T1.2 Files and Filesystems
    T1.2.1 Files and operations on files; concepts={file, filename, copy files, view files,...}
    ...
T4. The Gnome environment
```

Each course task `T` is represented as $(T_{ID}, T_{in}, T_{out}, T_{concepts}, T_{pre})$, where T_{in} is the input from the user's Activity Profile, T_{out} is the output for the user's Activity Profile and for STCS based on the user's work in `T`, $T_{concepts}$ is a set of domain concepts studied in `T`, and T_{pre} indicates the prerequisites for `T` (e.g. knowledge level for each $T_{concept}$, other tasks and resources required).

The task recommendation algorithm first selects a set of potential tasks to recommend from all tasks in the Course Task Model, by checking whether their T_{in} and T_{pre} are supported by the Activity Profile and the *belief_values* for the concepts in LTCS. OntoAIMS checks the concept knowledge threshold for the concepts in $T_{concepts}$ and recommends either to follow the task if the knowledge is not sufficient or to skip the task otherwise.

When the user chooses to perform the recommended task, the *OntoAIMS Resource Browser* (see Fig. 4) helps the user to learn more about the domain concepts related to that task. He can search for and read learning resources, browse domain concepts and study their definitions in the context of this task. For each resource in the search result list OntoAIMS provides two types of ranking - relevancy to the current task, and relevancy to the current user query. In this way, OntoAIMS can recommend resources for those concepts the user does not know or which contain mismatches with the domain ontology. The user's activities for a task and T_{out} are used to update the user model once the



Topic: Files and filesystems
Task: Files and operation on files

Search: file

Concept: 'file'
A directory, data file, symbolic-link or socket, etc. With other words, any set of data that has a pathname on the filesystem. When people say that anything on a unix system is a file, they actually refer to a file node, not a data file.

6 result documents
file

- Manipulating files in Linux
- Manipulating Files
- Introduction to the Linux Operating System – Lecture slid
- Introduction to the Linux Operating System – Lecture slid
- About your files in Linux
- Manipulating directories in Linux

Fig. 4. OntoAIMS resource browser

task is completed. It is possible to employ OWL-OLM to validate the updates to the Conceptual State in T_{out}, although this is not currently implemented.

5 Initial Evaluation of OntoAIMS with Users

We have conducted user studies with OntoAIMS to: (a) verify the functionality of its components (user modeling, task recommendation, and resource browser); (b) examine how users accept the integrated environment and its adaptive behavior; (c) identify how the system can be improved with additional adaptive features.

5.1 Observational studies

Two user studies were conducted with the current instantiation of OntoAIMS in a domain of Linux. Initially, *six users*, postgraduate students and staff from the universities of Leeds and Eindhoven, took part. An improved version of the system was used in a second study with *ten first year Computing undergraduates* at Leeds. It followed a two-week introductory course on Linux. In both studies, the users attended individual sessions, which lasted about an hour and were video recorded and monitored by an observer. OntoAIMS did not have data about the users prior to their logon to ensure realistic *cold start* conditions. The users were asked to study resources on Linux, which would be recommended by the system. At times, the observer interrupted the users to clarify their behavior with the system. At the end, the users were given a questionnaire related to their satisfaction and suggestions for system improvements.

5.2 Results and Discussion

We will discuss here the benefits of the integrated architecture, see [5] for more details about the OntoAIMS evaluation. OntoAIMS was regarded as helpful for both tuning one's knowledge in Linux and learning more about domain concepts. Every user appreciated the *integrated functionality* and worked with *all components*. Because at the start the system did not have any information about the users, it directed them to OWL-OLM (see Sect. 3.2), where users spent about half an hour on average. OWL-OLM probed their domain knowledge following the topics defined in the task ontology. Based on the dialog, the users were suggested tasks suitable for their level (see Sect. 4). They then spent about half an hour with the resource browser (see Sect. 4) exploring domain concepts and reading resources offered by the system.

Benefits from OWL-OLM The evaluation showed strong potential of OWL-OLM to deal with the *cold start problem*. OWL-OLM assessed the students' level of expertise and recommended them appropriate tasks to study. The expert users followed the OWL-OLM dialog answering most questions, and occasionally asked the system to confirm their domain statements. These users were pleased to be able to show their familiarity and to engage in discussions on more advanced domain topics. Less knowledgeable users struggled to answer the system's questions and often sought the answer from OWL-OLM. These users explored a variety of dialog moves, e.g. they disagreed with system's statements, composed new concepts and links, and asked several types of questions. There were occasions when discrepancies with the domain ontology were shown, which triggered corresponding clarification dialog games.

OWL-OLM was regarded by all users as a component that helped them learn. The students used the dialog to study about the domain and commented that the OWL-OLM dialog made them think about their knowledge, so they became aware of which concepts they were familiar with or struggling with. Indeed, as reported in [7], interactive open user modeling provides the means for reflection.

Benefits from task recommendation The user models generated with OWL-OLM were used to inform the task proposal. Some users were offered to skip tasks, as OWL-OLM found that they already knew quite a bit about that topic, while less knowledgeable users were directed to introductory topics. Most users agreed that the task recommended by the system was appropriate for their level, two students disagreed with this as they found the resources insufficient for the recommended topics. All users were pleased that the system could recommend them a task, they followed the recommended tasks, and regarded them as compliant with their learning goals. All users but one, said that they were aware why the task recommendation was made, which was due to the preceding OWL-OLM interactions. This gives a strong argument for the benefits of integartion.

Benefits from resource browsing The resource browser was regarded as "*a flexible way of looking at resources*". The users found it intuitive and easy to use. All users agreed that the graphical representation gave them an overview of the conceptual space: "*it allows to map your path through sequences of topics*", "*demonstrated exactly where I was in relation to the suggested topic*". The users were offered a set of resources ranked according to their appropriateness to the task. Depending on the goal (e.g. learning more about a concept, checking the syntax of a command, or tuning the student's domain knowledge), the resources the students were looking for differed in size, structure, and depth of domain knowledge. All users were pleased to see document ranking, but, again all of them, wanted this to be done not according to the the task but the user's preferences, knowledge, and current goal. This points at the need for further improvement of the integration, as discussed below.

Table 1. Additional adaptive features in OntoAIMS as pointed out by the ten users in the second study (the numbers show how many students support the feature).

Feature	**No**
I want the resources ranked according to my preferences and knowledge	10
I want the resources ordered according to my preferences and knowledge	8
I want the resources filtered according to my preferences and knowledge	4
I would like to be able to choose when the system should be adaptive	10
I would like to know what the system's thinks about my knowledge	10
I would like to know how my interaction with the system is used to form the system's opinion about my knowledge	8
I would like to know how the system's behavior is affected by its opinion about me	9
I would like to be able to inspect and change what the system thinks of me	10

Improving the integration and adaptation The evaluation pointed at improvements needed with regard to the integration between OWL-OLM and the resource browser. All students wanted to use a *flexible switch* between both modes. They stressed that this should be the user's choice, not something imposed by the system, and pointed at ways to implement it, e.g. offering the users a choice to go to the resource browser when they ask a question in OWL-OLM or enabling them go to a dialog to check their domain understanding after reading resources in the browser. The users in the second study were asked about *additional adaptive features*. The study pointed at future extensions of OntoAIMS to further integrate OWL-OLM and resource recommendation, see Table 1.

6 Conclusions and Future Work

The paper proposed an ontology-based approach for integrating interactive user modeling and learning content management to deal with typical adaptation problems on the Semantic Web, such as cold start, unreliability of user interaction for building conceptual UMs, and dynamics of a user's knowledge. We exemplified the approach in the integrated learning environment OntoAIMS for adaptive task recommendations and resource browsing on the Semantic Web. Initial results from two user studies were discussed. OntoAIMS shows a promising approach for dealing with adaptation on the Educational Semantic Web and contributes to this newly emerging strand.

Our immediate plans relate to improving OntoAIMS by adding additional integration and adaptation features, as suggested by the user studies. In the long run, we consider studies to (a) produce a good classification of users' mismatches and patterns for clarification dialog (b) design effective knowledge elicitation tools suited not for ontology engineers, but for users with a wide range of experiences, and (c) use Semantic Web services for the dynamic allocation of learning resources which are then flexibly integrated in OntoAIMS.

Acknowledgments. The research was supported by the UK-NL Partnership in Science and the EU NoE PROLEARN. The authors thank Michael Pye for implementing the OWL-OLM GUI and the participants in the evaluative study.

References

1. Aroyo, L., Dicheva, D.: Aims: Learning and teaching support for www-based education. Int. Journal for Continuing Engineering Education and Life-long Learning (IJCEELL) **11** (2001) 152–164
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001) 35–43
3. Bra, P.D., Aroyo, L., Chepegin, V.: The next big thing: Adaptive web-based systems. Journal of Digital Information, 5(1) (2004)
4. Brusilovsky, P., Eklund, J.: A study of user model based link annotation in educational hypermedia. Journal of Universal Computer Science, 4(4) (1998) 429–448
5. Denaux, R.: Ontology-based interactive user modeling for adaptive information systems. Master's thesis, Technische Universiteit Eindhoven (2005)
6. Denaux, R., Aroyo, L., Dimitrova, V.: An approach for ontology-based elicitation of user models to enable personalization on the semantic web. In: Proc. of WWW05. (to appear)
7. Dimitrova, V.: Style-olm: Interactive open learner modelling. Int. Journal of Artificial Intelligence in Education **13** (2003) 35–78
8. Henze, N.: Personalization functionality for the semantic web: Identification and description of techniques. Technical report, REWERSE project: Reasoning on the Web with Rules and Semantics (2004)
9. Kay, J.: The um toolkit for cooperative user modeling. User Modeling and User-Adapted Interaction **4** (1995) 149–196
10. McGuinness, D.L., van Harmelen, F.: Owl web ontology language overview. Technical report, W3C Recommendation (2004)
11. Vdovjak, R., Frasincar, F., Houben, G., Barna, P.: Engineering semantic web information systems in hera. Journal of Web Engineering, 2(1&2) (2003) 3–26
12. Weber, G., Kuhl, H., Weibelzahl, S.: Developing adaptive internet-based courses with the authoring system netcoach. In: Int. Workshop on Adaptive Hypermedia. (2001) 41–54
13. Zapata-Rivera, D., Greer, J.: Student model accuracy using inspectable bayesian student models. In: Proc. of AIED03. (2003) 65–72