Towards the Adaptive Semantic Web

Peter Dolog[1], Nicola Henze[2], Wolfgang Nejdl[1,2], and Michael Sintek[3]

[1] Learning Lab Lower Saxony,
University of Hannover,
Expo Plaza 1, D-30539 Hannover, Germany
dolog@learninglab.de
http://www.learninglab.de/~dolog
[2] ISI- Knowledge-Based Systems,
University of Hannover,
Appelstr. 4, D-30167 Hannover, Germany
{henze,nejdl}@kbs.uni-hannover.de
http://www.kbs.uni-hannover.de/~{henze,nejdl}
[3] German Research Center for Artificial Intelligence (DFKI) GmbH,
Knowledge Management Group,
Postfach 2080, D-67608 Kaiserslautern, Germany
Michael.Sintek@dfki.de
http://www.dfki.uni-kl.de/~sintek

Abstract. In this paper we show how personalization techniques from the area of adaptive hypermedia can be achieved in the semantic web. Our approach is based on rule-based reasoning enabled by semantic web technologies. The personalization techniques are formalized as reasoning rules. The rules are able to reason over distributed information resources annotated with semantic web metadata formats. This leads towards the realization of an adaptive semantic web idea which provides personalized, adaptive access to information, services, or other, distributed resources.

keywords: adaptive hypermedia, personalization, adaptive web, semantic web.

1 Introduction

Adaptive web, as envisioned in [10], should provide users with optimized access to distributed electronic information on the web according to particular needs of individual users or group of users. The main problem of current web systems their inability to support different needs of individual users. This problem is mainly due to their incapability to identify those needs, and insufficient mappings of those needs to available resources (information/document).

The *semantic web* [6] initiative reflects this problem by "giving information a well-defined meaning, better enabling computers and people to work in cooperation". This can be achieved by making metadata about different resources explicit using standardized descriptions.

Foundations for designing an *adaptive web* can be found in existing personalization and adaptive systems. For example, *recommender systems* (cf. [33, 4])

explore the usage of information entities (or products, services, etc.) in order to point out further interesting information, products, services to a user. Adaptive hypermedia systems (cf.provide individual guidance through the hyperspace by modeling and reasoning about explicit user models that contain preferences, goals, and further characteristics of individual users.

However, their closed architecture, own formats used for representing information about a user, documents and knowledge concepts does not allow to provide a personalization in wider distributed context. The exchange of the information and data between those applications and providing well formed adaptive functionality also to external parties cannot be achieved currently.

A step towards adaptive web can be made by employing standardized description formats for metadata. This will allow us to reason over facts described in standardized metadata formats. More important, reasoning can be performed in wider context over distributed data. In this paper we investigate how to provide reasoning on the semantic web with special focus on personalization techniques.

Based on our experience in developing adaptive hypermedia systems for e-learning, we propose separation of metadata about documents, information covered in the documents, users and observations. We propose a separation of adaptation rules from the metadata. These encapsulated reusable adaptive functionalities allow us to reason over distributed metadata in the web.

After a brief introduction to adaptive hypermedia systems (section 2), section 3 discusses how we can benefit from lessons learned from adaptive hypermedia systems in the adaptive web. Section 4 discusses a logic-based formalization of adaptive hypermedia. Section 6 describe how we can prototype an adaptive web utilizing the logical characterization of adaptive hypermedia. Related work is briefly reviewed in section 7. Finally we provide some remarks on further work in section 8.

2 Background: Adaptive Hypermedia Systems

Adaptive hypermedia is an alternative to the traditional "one-size-fits-all" static approach of hypermedia systems [9]. Adaptive hypermedia systems (AHS) enlarge the functionality of hypermedia systems to personalize the underlying hypermedia system to the individual user.

Adaptive hypermedia systems usually perform the adaptation based on several user features they maintain. The user features are used to determine appropriate information presentation and navigation sequences through the information. The systems are able to learn new user features from the user interaction with a system or from information he provided. Thus, each user has an individual view and individual navigational possibilities for working with the hypermedia system.

A hypermedia system can be described as a graph:

A set of nodes (of text) and a set of edges between these nodes (links) (cf. [31]). Thus personalization techniques in AHS can be grouped into techniques that adapt the nodes, which means to select/modify/rearrange the content of the documents (so called *content-level adaptation*) or the edges, which means to select appropriate hypertext links, delete links, or generate new, previously non existing links (so called *navigational-level adaptation*).

The content-level adaptation focuses mainly on improving a local navigation of a user and his orientation in currently presented page or fragment. For example, such adaptive systems can provide different text or different media variants, which serve information at different level of detail, to users with different level of knowledge or expertise in some field. They can switch between different media types according to different user preferences or his learning style. They are able to hide or appropriately annotate the parts of presented information which are not suited for current user based on values of his features the system maintains.

The navigational-level adaptation is trying to improve user global orientation in hyperspace. This includes for example a support for making exploration of required information easier such as enabling, disabling, showing, hiding, removing links when it is appropriate, or sorting links according to user goals or preferences. They can annotate links to indicate whether the links are appropriate or relevant. They are able to generate next appropriate information and thus to guide a user.

A specific research branch in the area of adaptive hypermedia, adaptive educational systems, improve learner guidance by showing e.g. the next reasonable learning step to take or by the individual creation of learning sequences. They have shown to improve orientation by annotating hypertext links with hints according to the students learning progress, by adapting the teaching or presentation style to the specific needs of the student, and by supporting learners to find their own optimal learning strategy. We are building on our experience in the area of educational systems in this paper as well.

3 From Adaptive Hypermedia to Adaptive Web

On the (semantic) web, we are confronted with a much more complex situation compared to the usually "closed" adaptive hypermedia settings: there is no central point of control and data storage, since on the web data and services are distributed and change rapidly and unpredictably. Furthermore, data and metadata is represented in a multitude of formats: while there are several upcoming standards, especially RDF/S [39, 38] and its extensions like DAML+OIL and OWL [15, 30], to define metadata schemas and (domain) ontologies, there are no commonly-agreed upon concrete schemas for metadata in specific domains like e-learning, nor are there domain ontologies shared by a whole community. Although first approaches to such generally useful metadata schemas and domain ontologies exist, like LOM [28] (building upon DC [20]) for the description of e-learning resources and classification systems like ACM [2] as a common domain ontology to characterize computer science content, it is not expected that they will be generally accepted as they cannot deal with the specific details of concrete (local) applications.

As a result, we need mechanisms on an adaptive web that allow resources, metadata, and ontology concepts to be mapped to each other and to the concepts found in user profiles. The technologies to accomplish this are currently being developed in the semantic web community. Promising approaches include those based on (logical) mapping rules and description logics [12, 29].

In order to provide personalized access to information on the web, we can therefore assume that the required mapping and integration technologies, in the style of wrappers and mediators in a web services (or P2P) architecture, are provided as part of the general semantic web infrastructure. What is missing for an adaptive web are the acquisition, representation, and usage of information about the users, i.e., their general preferences, interests, etc., and their specific situation and information need.

Because of the formal nature of resource metadata and domain ontologies on the semantic web, we propose a logic-based approach, building on the same technologies already developed for the semantic web. In the following sections, we will therefore first describe a general logic-based definition of adaptive educational hypermedia systems and then a specific prototype using the semantic web language TRIPLE [35].

4 Logic-based Definition of Adaptive Educational Hypermedia Systems

In order to develop design-proposals for the *adaptive web* based on AHS techniques we have to analyze architecture and, even more important, data resources and data flow in AHS. A functionality-oriented definition of adaptive hypermedia given by Brusilovsky is [9]: "By adaptive hypermedia systems we mean all hypertext and hypermedia systems which reflect some features of the user in the user model and apply this model to adapt various visible aspects of the system to the user."

A logic-based definition of adaptive educational hypermedia systems has been proposed in [25]. Components of an AHS are therefore the *hypermedia system* which includes information about documents and their relations, the *user model* which stores characteristics of a user and possibly reasoning rules to derive additional user characteristics, and the *adaptation component* which determines the adaptive treatment provided for the particular user. During runtime, the system monitors a user's interaction to update the user model, thus a component for *observations* is necessary, too. Basis data required by the AHS can be found in the components "hypermedia system" (e.g. metadata about documents) and "observations" (e.g. usage data during runtime). The "user model" component processes data from both "hypermedia system" and "observations" to describe and reason about a user's characteristics. The "adaptation component" finally decides about beneficial adaptive treatments for a user based on data of the other three components. The component "hypermedia system" has been generalized to *document space* due to the fact that a hypermedia system requires stronger assumptions about the modeling of documents than we can expect to have in the World Wide Web.

Definition 1 (Adaptive Educational Hypermedia System [25]). *An adaptive educational hypermedia system (AEHS) is a Quadruple*

$$(\mathit{DOCS}, \mathit{UM}, \mathit{OBS}, \mathit{AC}) \tag{1}$$

with

DOCS: Doc*ument* **S***pace: A finite set of first order logic (FOL) sentences with constant symbols for describing documents (and knowledge concepts), and predicates for defining relations between these (and other) constant symbols.*
UM: U*ser* **M***odel: A finite set of FOL sentences with constant symbols for describing individual users (user groups), and user characteristics, as well as predicates and formulas for expressing whether a characteristic applies to a user.*
OBS: Obs*ervations: A finite set of FOL sentences with constant symbols for describing observations, and predicates for relating users, documents / concepts, and observations.*
AC: A*daptation* **C***omponent: A finite set of FOL sentences with formulas for describing adaptive functionality.*

5 Example: Logically Describing an AEHS

In this section we will provide an example setting for illustrating the approach proposed in this paper. We will define an adaptive system called *SIMPLE* which can annotate information resources in order to give orientation by pointing out where to start learning, and by indicating more advanced concepts. The adaptation techniques in SIMPLE are well known techniques in existing adaptive hypermedia techniques. Our aim is to translate some well-explored techniques to the semantic web. Doing this, we show how personalization can be brought to the semantic web. Of course, more advanced adaptation techniques can be implemented, or new techniques can be investigated. The important step which will be illustrated in our example is to bring personalization to the logic level of the semantic web tower [5]: This requires the encapsulation of adaptation techniques for reuse (for example by formalizing adaptation techniques as logical formulas as done in [25]) and enabling adaptation techniques to reason over various information resource.

The following sections will describe the AEHS SIMPLE and its functionality in FOL, according to definition 1.

5.1 SIMPLE: Document Space

SIMPLE needs information about available information resources - like documents, knowledge concepts, etc. For the logical description we therefor need a set of n constant symbols (n corresponds to the number of documents in the document space) which name the documents (the URI of the documents respectively) and a set of s atoms (s corresponds to the number of concepts in the knowledge space) which name the knowledge concepts:

D_1, D_2, ..., D_n, C_1, C_2, ..., C_s.

The predicate "depends" states learning dependencies between these concepts: Concept C is required to understand C':

depends(C, C') for certain concepts $C \neq C'$.

Furthermore, the documents can be annotated with a non-empty set of concepts. This can be expressed by the hasConcept-predicate between documents and concepts. This can be compared to assigning a set of keywords to each document.

$\forall D\ \exists C$ hasConcept(D, C).

We cannot ensure that this constraint is fulfilled in the open world. However, this rule says that a valid metadata – annotation for the AEHS Simple requires that each document is related to at least one concept.

Further predicates between concepts and/or documents are possible but not required by this example of an AEHS.

5.2 SIMPLE: User Model

The adaptive system SIMPLE needs data from the user model - in order to reflect the user's actual learning progress and learning state. Again, the user model of SIMPLE is very straightforward and models only the user's knowledge state - further user characteristics can be considered as well.

SIMPLE's user model contains a set of m constant symbols, one for each individual (registered) user:

U_1, U_2, ..., U_m.

A rule defines that a concept C is assumed to be learned whenever the corresponding document has been visited by the user. Therefor, SIMPLE uses the constant symbol

Learned.

The rule for processing the observation that a concept has been learned by a user is given by:

$\forall U\ \forall C$
($\exists D$ keyword$(D, C) \wedge$ obs$(D, U$, Visited)
$\Longrightarrow$ p_obs$(C, U$, Learned).

5.3 SIMPLE: Observations

SIMPLE interpretes from the observations about a user's action with the personalized systems only whether a user has visited some page. The time s/he spend on this resource, or further information like information about the order in which the user has accessed resources, or further interactions, are not required for SIMPLE. Thus, one constant symbol for the observation whether a document has been visited is enough:

Visited.

A predicate is relating this observation to a user U and a document D:

obs(D, U, Visited) for certain D, U.

5.4 SIMPLE: Adaptation Component

The example system SIMPLE has the following adaptive functionality: It can annotate hypertext-links to documents by using the traffic light metaphor[9]. The traffic light metaphor belongs to the group of navigational-level adaptation techniques and assigns colors of a traffic light to point out the educational state of a hypertext link (more precisely, the educational state of the resource to which this hypertext link points to). E.g. a green color is used to indicate that a link leads to a resource which is *Recommended* to a learner (because he has sufficient pre-knowledge, or because of other reasons), a red color is used to show that visiting the hypertext link is *Not_Recommended* (because e.g. the user actually lacks some knowledge for successfully learn the resource in question). In addition, SIMPLE shows which links lead to documents that *Will_become_understandable* (annotated by a dark orange icon; the user has started to learn some of the necessary prerequisite knowledge), and documents that *Might_be_understandable* (translated into a yellow icon; the user has learned nearly all required prerequisite knowledge). A white icon in front of a link indicates that a user has *Already_visited* the resource belonging to the link.

To express the adaptation formulas, SIMPLE uses five constant symbols for representing the *learning state* of a document:

Recommended, Not_Recommended, Might_be_understandable,
Will_become_understandable, Already_learned.

The following formulas then describe the educational state of a document. Formula_1 states that a document is `Already_learned` whenever the user has read the document

$\forall U \ \forall D$
obs(D, U, Visited)
$\Longrightarrow$ learning_state(D, U, Already_visited).

Formula_2 states that a document is `Recommended` for learning if *all* prerequisites for the keywords of this document are learned

$$\forall U\ \forall D$$
$$\forall C\ \Big(\text{keyword}(D,C) \Longrightarrow \big(\forall C'\ \text{depends}(C,C') \Longrightarrow \text{p\_obs}(C',U,\text{Learned})\big)\Big)$$
$$\wedge\ \neg\ \text{learning\_state}(D,U,\ \text{Already\_visited})$$
$$\Longrightarrow \text{learning\_state}(D,U,\ \text{Recommended}).$$

Formula_3 states that a document `Might_be_understandable` if at least some of the prerequisites have already been learned by this user:

$$\forall U\ \forall D$$
$$(\ \forall C\ \text{keyword}(D,C) \Longrightarrow$$
$$(\ \exists C'\ \text{depends}(C,C') \Longrightarrow \text{p\_obs}(C',U,\ \text{Learned})\ )\ )$$
$$\wedge\ \neg\ \text{learning\_state}(D,U,\ \text{Recommended})$$
$$\Longrightarrow \text{learning\_state}(D,U,\ \text{Might\_be\_understandable}).$$

Formula_4 derives that a document `Will_become_understandable` if the user has some prerequisite knowledge for at least one of the document's keywords:

$$\forall U\ \forall D$$
$$\exists C\ \text{keyword}(D,C) \Longrightarrow$$
$$(\exists C'\ \text{depends}(C,C') \Longrightarrow \text{p\_obs}(C',U,\ \text{Learned})\ )$$
$$\wedge\ \neg\ \text{learning\_state}(D,U,\ \text{Might\_be\_understandable})$$
$$\Longrightarrow \text{learning\_state}(D,U,\ \text{Will\_become\_understandable}).$$

$$\forall U\ \forall D$$
$$\neg\ \text{learning\_state}(D,U,\ \text{Will\_become\_understandable})$$
$$\Longrightarrow \text{learning\_state}(D,U,\ \text{Not\_Recommended}).$$

The translation of the `learning_state` into `document_annotations` with green, red, orange, yellow or white icons is straightforward, for example to annotate a document with a green ball, is expressed by

$$\forall U\ \forall D$$
$$\text{learning\_state}(D,U,\ \text{Recommended})$$
$$\Longrightarrow \text{document\_annotation}(D,U,\ \text{Green\_Icon})$$

6 Prototyping the Adaptive Web

The logically described adaptive educational hypermedia system SIMPLE, proposed in section 5 will be implemented in TRIPLE. The formal description of adaptive systems in FOL [25] allows us to reuse the adaptation rules in different contexts, here we will use them to access learning resources from the Sun Java Tutorial [13, 14]. There exists a freely available online version[4] from the Sun Java Tutorial. We annotated the resources of the Sun Java Tutorial according to our document ontology as can be seen in the next section.

[4] http://java.sun.com/docs/books/tutorial/

6.1 TRIPLE Overview

TRIPLE [34] is a rule language for the Semantic Web which is based on Horn logic and borrows many basic features from F-Logic [27] but is especially designed for querying and transforming RDF models.

TRIPLE can be viewed as a successor of SiLRI (Simple Logic-based RDF Interpreter [16]). One of the most important differences to F-Logic and SiLRI is that TRIPLE does not have fixed semantics for object-oriented features like classes and inheritance. Its modular architecture allows such features to be easily defined for different object-oriented and other data models like UML, Topic Maps, or RDF Schema. Description logics extensions of RDF (Schema) like OIL, DAML+OIL, and OWL that cannot be fully handled by Horn logic are provided as modules that interact with a description logic classifier, e.g. FaCT [26], resulting in a hybrid rule language.

Namespaces and Resources TRIPLE has special support for namespaces and resource identifiers. Namespaces are declared via clause-like constructs of the form *nsabbrev* := *namespace*., e.g., rdf := $''http://www.w3.org/1999/02/22-rdf-syntax-ns\#''$. Resources are written as *nsabbrev:name*, where *nsabbrev* is a namespace abbreviation and *name* is the local name of the resource.

Statements and Molecules Inspired by F-Logic object syntax, an RDF statement (triple) is written as: *subject*[*predicate* → *object*]. Several statements with the same subject can be abbreviated as "molecules":

$$\text{stefan}[\text{hasAge} \rightarrow 33; \text{isMarried} \rightarrow \text{yes}; \ldots]$$

Models RDF models, i.e., sets of statements, are made explicit in TRIPLE ("first class citizens").[5] Statements, molecules, and also Horn atoms that are true in a specific model are written as *atom*@*model* (similar to Flora-2 module syntax), where *atom* is a statement, molecule, or Horn atom and *model* is a model specification (i.e., a resource denoting a model), e.g.:

$$\text{michael}[\text{hasAge} \rightarrow 35]@\text{factsAboutDFKI}$$

TRIPLE also allows Skolem functions as model specifications. Skolem functions can be used to transform one model (or several models) into a new one when used in rules (e.g., for ontology mapping/integration):

$$O[P \rightarrow Q]@\text{sf}(m1, X, Y) \;\longleftarrow\; \ldots$$

Logical Formulae TRIPLE uses the usual set of connectives and quantifiers for building formulae from statements/molecules and Horn atoms, i.e., $\wedge$, $\vee$, $\neg$, $\forall$, $\exists$, etc.[6] All variables must be introduced via quantifiers, therefore marking them is not necessary.

---

[5] Note that the notion of *model* in RDF does not coincide with its use in (mathematical) logics.

[6] For TRIPLE programs in plain ASCII syntax, the symbols AND, OR, NOT, FORALL, EXISTS, `<-`, `->`, etc. are used.

Clauses and Blocks A TRIPLE clause is either a fact or a rule. Rule heads may only contain conjunctions of molecules and Horn atoms and must not contain (explicitly or implicitly) any disjunctive or negated expressions. To assert that a set of clauses is true in a specific model, a model block is used: @*model* {*clauses*}, or, in case the model specification is parameterized:

$$\forall\ Mdl\ \ @model(Mdl)\ \{clauses\}$$

Semantics, Implementation TRIPLE has been implemented via a translation to Horn Logic plus enactment by XSB [36], i.e., a Prolog with tabled resolution, giving it the well-founded semantics [37]. Details on the model-theoretic semantics of TRIPLE can be found in [17].

6.2 Facts About a Document

A simple structure for document meta data and relationships to other structures is depicted in fig. 1. The class `Document` is used to annotate a resource which is a document. Documents describe some concepts. The concepts are the main information entities from domain knowledge communicated by the documents (cf. [32]). Concept and Documents are related through `dc:subject` property. Documents can have its prerequisites. This is annotated by `dcterms:requires` property.

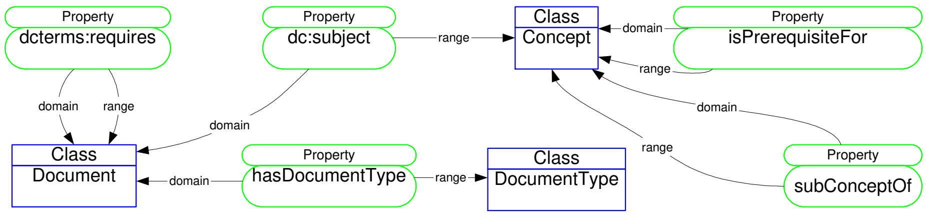

Fig. 1. Ontology for documents

An example of such resource can be a page describing object oriented class concept (URI of the resource is `sun_java:java/concepts/class.html`. Following example shows how such page can be annotated based on the document metadata structure.

```
sun_java:java/concepts/class.html[
rdf:type->doc:Document;
dc:subject->java:OO_Class;
...].

java:OO_Class[
rdf:type->doc:Concept;
```

```
doc:isPrerequisiteFor->doc:OO_Method;
...
].
```

The page is annotated with type `Document`. It describes information about classes (annotation `dc:subject -> OO_Class`). The `OO_Class` is of type Concept and is subconcept of `Classes_and_objects`. The `OO_Class` is prerequisite for the `OO_Method` concept.

The structure of the document metadata can be more complex. It can contain for example a slot for annotating a role of document, its type, level of covering particular concept in the document, roles of concept in particular document, position of particular document in document structure described by whole/part relationships and so on.

All of these relationships can enhance adaptation possibilities for example for construction of learning sequences based on user profile, annotating position of a user in the document structure, helping to identify main outcomes of a document based on roles and level of concept coverage, and so on.

6.3 Facts About a Domain

The `dc:subject` entries used in examples in previous section can be seen as facts about a domain. Facts about domain can form complex structures. For space limitation we show just a fragment of domain knowledge base about concepts for java programming. For the same reason we show just `isa` (subConceptOf) relationship between these concepts. Figure 2 depicts `Programming_Strategies` concept with its subconcepts: `Object_Oriented`, `Imperative`, `Logical`, and `Functional`. The `OO_Class`, `OO_Method`, `OO_Object`, `OO_Inheritance`, and `OO_Interface` are depicted as subconcepts of `Object_Oriented`.

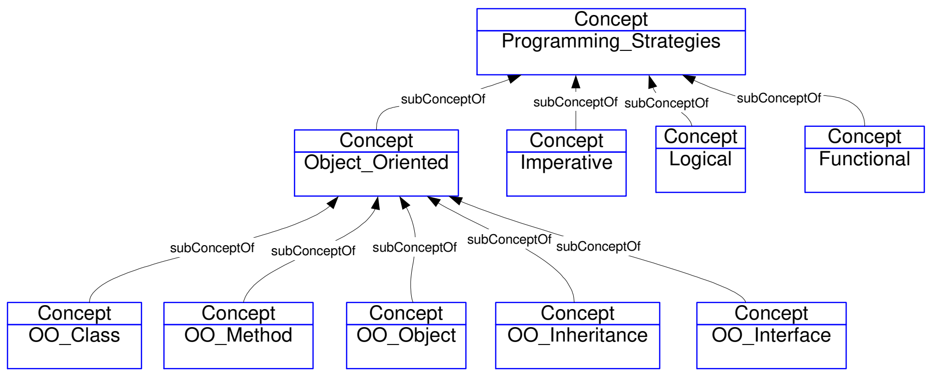

Fig. 2. Concept ontology for Java e-lecture

The facts about a domain and different relationships between the facts can be used for adaptation purposes quite successfully. The mentioned `subConceptOf` relationship can be for example utilized to recommend either more general documents introducing a concept of programming strategies in general, or to recommend more specific documents (resources) about object oriented programming strategy based on requirements, level of knowledge, or interest of a user.

Sequencing relationship is another relationship which can be used to recommend documents. A document (resource) which describes a concept (the concept appears in `dc:subject` slot in meta data about the document) from the beginning of the sequence will be recommended sooner than a document which describes a concept from the end of such a sequence.

A dependency relationship referring to whether a concept depends on another concept can be used as well. It can be used to recommend documents which describe dependent concepts together with a document describing a concept which was recommended by another rule.

6.4 Facts about the User Profile

We need to have explicit facts about a user to be able to recommend documents (resources) which are relevant to user interests. This facts can enclose information about user performance from courses within a domain, his goals and interests, preferences, and so on. Following simple example contains just some facts about resources visited by a user (for more complex user models see, e.g., [19]).

```
@simple:data{
...
kbs:henze[
        rdf:type->User;
        obs:visited->sun_java:'index.html'].

l3s:dolog[
        rdf:type->User;
        obs:visited->sun_java:'index.html';
        obs:visited->sun_java:'java/index.html';
        obs:visited->sun_java:'java/concepts/index.html';
        obs:visited->sun_java:'java/concepts/object.html'].

}
```

The user identified by `kbs:henze` URI has visited general sun java introduction page represented by `index.html`. The user identified by `l3s:dolog` URI has visited the general introduction, the java introduction (`sun_java:'java/index.html'`), java concepts introduction (`sun_java:'java/concepts/index.html'`), and objects introduction (`sun_java:'java/concepts/object.html'`).

Other facts and relationships between the facts can help to derive additional information which can improve recommendation results. For example a time spent on resource can be used to infer a level of knowledge gained from the resource.

6.5 Reasoning Rules for Adaptation

The adaptive system SIMPLE which we have implemented (see section 5) requires only view information about the user's characteristics. Thus, for our example we employed a very simple user model: This user model traces the users path in the learning environment and registers whenever the user has visited some learning resource.

The TRIPLE rules for adaptive functionality will be described in the following.

Following rule asserts that all triples from `@simple:data` view are also triples in the `@simple:adaptation`.

```
FORALL O,P,V O[P->V] <- O[P->V]@simple:data.
```

Following rules derive all documents, concepts, and users from metadata based on types taken from ontologies.

```
FORALL D document(D) <- D[rdf:type->Document].
FORALL C concept(C) <- C[rdf:type->Concept].
FORALL U user(U) <- U[rdf:type->User].
```

We need to derive observations about a user from metadata from user profile (see an example in section 6.4). The observations states whether particular user learned concept based on visiting a document describing the concept. The following rule derive all learned concepts.

```
FORALL C, U  p_obs(C, U, Learned) <- user(U) AND concept(C) AND
  EXISTS D (D[dc:subject->C] AND U[obs:visited->D]).
```

To derive appropriate recommendation annotation for particular user, prerequisite concepts of documents have to be learned by a user. Following rule derive concepts which are prerequisite concepts for a document being analyzed for recommendation. The prerequisite concepts are derived according to the `doc:isPrerequisiteFor` relationship.

```
FORALL D, C, Ck prerequisite_concepts(D, Ck) <-
  document(D) AND concept(Ck) AND
  D[dc:subject->C] AND Ck[doc:isPrerequisiteFor->C].
```

All previously mentioned rules are used in recommendation rules. We have five rules for recommendation annotations. The first one annotates a document which was visited.

```
FORALL D, U learning_state(D, U, Already_visited) <-
   user(U) AND document(D) AND U[obs:visited->D].
```

The second rule determines documents which are `Recommended`. The recommendation rule is performed according to a FOL sentence from section 5 that document is recommended if *all* prerequisite concepts for all of its concepts are learned.

```
FORALL D, U learning_state(D, U, Recommended) <-
   document(D) AND user(U) AND NOT learning_state(D, U, Already_visited)
   AND FORALL Ck (prerequisite_concepts(D, Ck) ->
      p_obs(Ck, U, Learned)).
```

The third rule derives less strong recommendation. It asserts that document `Might_be_understandable` if at least one prerequisite concept of for all of its concepts was learned.

```
FORALL D, U learning_state(D, U, Might_be_understandable) <-
   document(D) AND user(U)
   AND NOT learning_state(D, U, Recommended)
   AND NOT learning_state(D, U, Already_visited) AND
   FORALL C (D[dc:subject->C] ->
     (EXISTS Ck (Ck[doc:isPrerequisiteFor->C]
      AND p_obs(Ck, U, Learned)))).
```

A document is annotated as `Will_become_understandable` if at least one prerequisite concept among its concepts was learned.

```
FORALL D, U learning_state(D, U, Will_become_understandable) <-
   document(D) AND user(U)
   AND NOT learning_state(D, U, Recommended)
   AND NOT learning_state(D, U, Might_be_understandable)
   AND NOT learning_state(D, U, Already_visited)
   AND EXISTS Ck (prerequisite_concepts(D, Ck) AND p_obs(Ck, U,
   Learned)).
```

For all other cases a document is annotated as `Not_recommended`.

```
FORALL D, U learning_state(D, U, Not_recommended) <-
   document(D) AND user(U)
   AND NOT learning_state(D, U, Recommended)
   AND NOT learning_state(D, U, Might_be_understandable)
   AND NOT learning_state(D, U, Already_visited)
   AND NOT learning_state(D, U, Will_become_understandable).
```

A query which queries for documents and particular annotations about recommendation for particular user can look as follows.

```
FORALL U, D, L <- learning_state(D, U, L)@simple:adaptation.
```

A subset of results derived from our knowledge-base is:

```
U=doc:kbs:henze, D=sun_java:'index.html', L='Already_visited'
U=doc:l3s:dolog, D=sun_java:'index.html', L='Already_visited'
U=doc:kbs:henze, D=sun_java:'java/concepts/message.html',
       L='Not_recommended'
U=doc:l3s:dolog, D=sun_java:'java/concepts/message.html', L='Recommended'
...
```

The results show that the index.html document has learning state `Already_visited` for both users. Then the results say that the document message.html is `Not_recommended` to user identified by `kbs:henze` URI, and it is `Recommended` for the user identified by `l3s:dolog`.

The complete example of the rules used in this paper can be found in triple-file at `http://www.learninglab.de/~dolog/seminar/seminar03files/ppswr03_triple_example.triple`.

7 Related Work

Adaptive hypermedia has been studied normally in closed worlds, i.e. the underlying document space / the hypermedia system has been known to the authors of the adaptive hypermedia system at design time of the system. As a consequence, changes to this document space can hardly be considered: A change to the document space normally requires the reorganization of the document space (or at least some of the documents in the document space). To open up this setting for dynamic document or information spaces, approaches for so called *open corpus adaptive hypermedia systems* have been discussed [9, 23]. Our approach to bring adaptive hypermedia techniques to the web will therefore contribute to the open corpus problem in AH.

Contributions to open corpus adaptive hypermedia can be found in the area of so called *open hypermedia systems*, too. Open hypermedia is an approach to relationship management and information organization for hypertext-like structure servers. Key features are the separation of relationships and content, the integration of third party applications, and advanced hypermedia data models allowing, e.g., the modeling of complex relationships. Approaches to open hypermedia have been discussed, e.g., in [1]. [3] aims to describe adaptive hypermedia techniques for open hypermedia by relating basic fundamental open hypermedia model concepts with adaptive hypermedia techniques. The work presented in this paper settles on more general descriptions of the data objects used in open hypermedia: Instead of using specific kinds of data objects [3], we use resources that are annotated by general RDF metadata. Because we provide a more expressive language for specifying contexts — Triple based queries and constraints — we can have more complex rules for specifying accessibility and usage in general. Behavior like update of user profiles can also be associated within the RDF annotation of the resource and as Triple programs. RDF annotations provide several possibilities for specifying relationships and association, as defined by the RDF schema, and domain ontologies are defined as RDF data again in the form of domain ontologies. Our work is also related to [23, 24], and extends it by investigating the different standards relevant for adaptive functionalities in an open environment and how to use queries to implement that functionality. This work also extends our work published in [18] where we made first steps towards adaptive hypermedia based on logical characterization. We showed and implemented more complex rules in comparison to the work and we employed more powerful rule-based language for expressing adaptive functionality. The core differences are availability of models (views) in TRIPLE and direct support for namespaces and URIs.

If we compare our work with standard models for adaptive hypermedia systems such as the one used in AHA! [8] for example, we observe that they define several model like conceptual, navigational, adaptation, teacher and user models. Compared to our approach, these models either correspond to ontologies / taxonomies, to different schemas describing teacher and user profile, and to schemas describing the navigational structure of a course. We express adaptation functionalities as encapsulated and reusable Triple rules, while the adaptation model in AHA uses a rule based language encoded into XML. At the level of concept or information items AHA! provides functionalities to describe requirements [7] for the resource, which state what is required from a user to visit that information.

In our approach, we used the RDF-querying and transformation language TRIPLE. Related approaches in the area of querying languages for the semantic web can be found, e.g., in [11]. Here, a rule-based querying and transformation language for XML is proposed. A web language for rules is currently under development of the Rule Markup-Language (RuleML) Initiative [7]. A discussion of the interoperability between RuleML Logic programs and ontologies (coded in OWL[8] (web ontology language) or DAML+OIL[9] (Darpa Agent Markup Language + Ontology Inference Layer) can be found in [21].

Reasoning in open worlds like the semantic web is not fully explored yet, sharing and reusing of resources with high quality is still an open problem. In this paper, we discussed first ideas on the application of rules and rule-based querying and transformation language for the domain of adaptation.

8 Conclusions and Further Work

In this paper, we described an approach to personalization on the semantic web based on reasoning. We have shown how known techniques from adaptive hypermedia (which normally work in closed worlds) can be realized in an open world setting like the semantic web. The logical characterization of adaptive hypermedia enables the formalization of personalization techniques in a common language (FOL). The rule-based language TRIPLE allowed us to implement this functionality and reason over distributed metadata. The ontology based metadata descriptions and thorough use of the ontologies in the descriptions allowed us to employ monotonic reasoning.

However, the semantic web is characterized with non-complete information. We would like to continue with experiments with the environment where we will not have always complete information to derive conclusions. Thus additional experiments either with non-monotonic reasoning, with additional heuristics to

[7] http://www.dfki.uni-kl.de/ruleml/
[8] http://www.w3.org/2001/sw/WebOnt/
[9] http://www.w3.org/TR/daml+oil-reference

derive conclusions from not complete information, or experiments with local closed worlds [22] are needed.

On the other hand, we would like to investigate also more complex concept ontologies. In this paper we use just two types of relationships for relating concepts, namely *isPrerequisiteFor* and *subConceptOf*. This includes for example decomposition of sequences into smaller parts or content packaging with additional relationships in concept model (ontology), e.g., partOf, belongsTo, and alternatives. In addition, we can consider explicit ontologies for educational models which can provide us with complementary facts for improving conclusions from personalization reasoning rules. This enables to add additional rules to enhance adaptive functionalities based on the facts modeled in knowledge-base by utilizing additional relationships.

At the application level we also would like to experiment with different visualization strategies for displaying results of reasoning.

References

1. K. Anderson, R. Taylor, and E. Whitehead. Chimera: Hypermedia for heterogeneous software development environments. *ACM Transactions on Information Systems*, 18(3):211–245, 2000.
2. Association of Computing Machinery: The ACM computer classification system. http://www.acm.org/class/1998/, 2002.
3. C. Bailey, W. Hall, D. Millard, and M. Weal. Towards open adaptive hypermedia. In *Proccedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002)*, Malaga, Spain, May 2002.
4. C. Basu, H. Hirsh, and W. W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Fifteenth National Conference on Artificial Intelligence*, pages 714–720, Madison, USA, 1998.
5. T. Berners-Lee. The semantic web - mit/lcs seminar. http://www.w3c.org/2002/Talks/09-lcs-sweb-tbl/.
6. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.
7. P. D. Bra, A. Aerts, D. Smits, and N. Stash. AHA! version 2.0. In *Proceedings of the AACE ELearn'2002 conference*, pages 240–246, Oct. 2002.
8. P. D. Bra, G.-J. Houben, and H. Wu. AHAM: A dexter-based reference model for adaptive hypermedia. In K. Tochtermann, J. Westbomke, U. Wiil, and J. Leggett, editors, *Proc. of ACM Conference on Hypertext and Hypermedia*, pages 147–156, Darmstadt, Germany, Feb. 1999.
9. P. Brusilovsky. Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 11(1-2):87–100, 2001.
10. P. Brusilovsky and M. Maybury. *The Adaptive Web*. Communications of the ACM, 2002.
11. F. Bry and S. Schaffert. A gentle introduction into xcerpt, a rule-based query and transformation language for xml. In *International Workshop on Rule Markup Languages for Buisiness Rules on the Semantic Web*, Sardinia, Italy, June 2002.
12. D. Calvanese, G. D. Giacomo, and M. Lenzerini. Ontology of integration and integration of ontologies. In *Proceedings of the 2001 Description Logic Workshop (DL 2001)*, 2001.

13. M. Campione and K. Walrath. The java tutorial. http://java.sun.com/docs/books/tutorial/.
14. M. Campione and K. Walrath. *The Java(TM) Tutorial: A Short Course on the Basics (3rd Edition)*. Addison-Wesley, 2000.
15. DAML+OIL, 2001. http://www.daml.org/2001/03/daml+oil-index.html.
16. S. Decker, D. Brickley, J. Saarela, and J. Angele. A query and inference service for RDF. In *QL'98 — The Query Languages Workshop*, Boston, USA, 1998. World-WideWeb Consortium (W3C).
17. S. Decker, M. Sintek, and W. Nejdl. The model-theoretic semantics of triple.
18. P. Dolog, R. Gavriloaie, W. Nejdl, and J. Brase. Integrating adaptive hypermedia techniques and open rdf-based environments. In *Proc. of 12th International World Wide Web Conference*, Budapest, Hungary, May 2003.
19. P. Dolog and W. Nejdl. Benefits and challenges of the semantic web for user modelling. In *International Workshop on Adaptive Hypermedia and Adaptive Web-based Systems (AH 2003)*, Budapest, Hungary, 2003.
20. The Dublin Core Metadata Initiative. `http://dublincore.org/`.
21. B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Twelth International World Wide Web Conference*, Budapest, Hungary, May 2003.
22. J. Heflin and H. Munoz-Avila. Lcw-based agent planning for the semantic web. In *Ontologies and the Semantic Web. Papers from the 2002 AAAI Workshop WS-02-11*, pages 63–70, Menlo Park, CA, Nov. 2002.
23. N. Henze and W. Nejdl. Adaptation in open corpus hypermedia. *IJAIED Special Issue on Adaptive and Intelligent Web-Based Systems*, 12, 2001.
24. N. Henze and W. Nejdl. Knowledge modeling for open adaptive hypermedia. In *Proccedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002)*, Malaga, Spain, 2002.
25. N. Henze and W. Nejdl. Logically characterizing adaptive educational hypermedia systems. In *International Workshop on Adaptive Hypermedia and Adaptive Web-based Systems (AH 2003)*, Budapest, Hungary, 2003.
26. I. Horrocks. The FaCT System, 2001. URL: `http://www.cs.man.ac.uk/∼horrocks/FaCT/`.
27. M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741–843, July 1995.
28. LOM: Draft Standard for Learning Object Metadata. http://ltsc.ieee.org/wg12/doc.html.
29. Z. Miklós, G. Neumann, U. Zdun, and M. Sintek. Querying semantic web resources using TRIPLE views. In *Proceedings of the 2nd International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, Oct. 2003.
30. OWL, 2003. http://www.w3.org/2001/sw/WebOnt/.
31. R. Rada. *Interactive Media*. Springer, 1995.
32. Reference model for an open archival information system (oais), 2002. http://ssdoo.gsfc.nasa.gov/nost/isoas/.
33. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of CHI'95 – Human Factors in Computing Systems*, pages 210–217, 1995.
34. M. Sintek and S. Decker. TRIPLE—A query, inference, and transformation language for the semantic web. In *1st International Semantic Web Conference (ISWC2002)*, June 2002.

35. M. Sintek and S. Decker. Triple - an rdf query, inference, and transformation language. In I. Horrocks and J. Hendler, editors, *International Semantic Web Conference (ISWC)*, pages 364–378, Sardinia, Italy, 2002. LNCS 2342.
36. SUNY. The XSB programming system. Dept. of Computer Science, SUNY at Stony Brook, 2000. URL: `http://www.cs.sunysb.edu/~sbprolog/xsb-page.html`.
37. A. van Gelder, K. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.
38. W3C. Resource Description Framework (RDF) Schema Specification 1.0, 2001. URL: `http://www.w3.org/TR/2000/CR-rdf-schema-20000327/`.
39. W3C. Semantic Web Activity: Resource Description Framework (RDF), 2001. URL: `http://www.w3.org/RDF/`.