23

Semantic Web Technologies for the Adaptive Web

Peter Dolog[1] and Wolfgang Nejdl[2]

[1] Department of Computer Science, Aalborg University,
Fredrik Bajers Vej 7E, DK-9220 Aalborg, Denmark,
dolog@cs.aau.dk
http://www.cs.aau.dk/~dolog
[2] L3S Research Center, University of Hannover
Appelstrasse 9A, 30167 Hannover, Germany
nejdl@l3s.de

Abstract. Ontologies and reasoning are the key terms brought into focus by the semantic web community. Formal representation of ontologies in a common data model on the web can be taken as a foundation for adaptive web technologies as well. This chapter describes how ontologies shared on the semantic web provide conceptualization for the links which are a main vehicle to access information on the web. The subject domain ontologies serve as constraints for generating only those links which are relevant for the domain a user is currently interested in. Furthermore, user model ontologies provide additional means for deciding which links to show, annotate, hide, generate, and reorder. The semantic web technologies provide means to formalize the domain ontologies and metadata created from them. The formalization enables reasoning for personalization decisions. This chapter describes which components are crucial to be formalized by the semantic web ontologies for adaptive web. We use examples from an eLearning domain to illustrate the principles which are broadly applicable to any information domain on the web.

23.1 Introduction

Information access on the web is realized through the hypertext paradigm. Hypertext interlinks related pieces of information (pages) and allows the user to browse through the information space. The links are provided either explicitly, encoded by authors of the pages, or they are generated automatically, for example based on the results of a query.

Personalized information access in this context is concerned with user-centered bias of the hyperlinks to better support the current user context. Generating links automatically, taking user profiles into account, is a very attractive option but creates challenges as well. According to [5], adaptive web systems extend the adaptive navigation and presentation techniques from closed corpus adaptive hypermedia to the open corpus information resources available on the web and thus supporting personalized access on the web. In this chapter we discuss solutions based on semantic web techniques to

P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.): The Adaptive Web, LNCS 4321, pp. 697–719, 2007.

realize personalized link generation. Key aspects of this solution are ontologies and reasoning techniques. Ontologies represent shared and agreed upon conceptual models in a domain, which describe the main concepts of the domain and their relationships. Ontologies can thus serve as reference models for generating links in this domain, and represent hypertext, content and user information. Reasoning techniques can then work on metadata based on these ontologies, and generate links based on content, user context and user background.

As discussed in Chapter 8 [4] of this book, hypertext is a collection of text fragments interconnected by active links, used to access the information fragments addressed by them. Research in the hypertext community has concentrated on how to improve navigation in hypertext systems. The hypertext community has been concerned with several ways of browsing [18, 17]. Information retrieval concepts have been studied together with hypertext concepts [1, 35].

We can distinguish between two link concepts in hypertext: links maintained within the text (embedded links) and links maintained externally to the text as first class entities. Hypertext which utilizes the first view is often denoted as a *closed hypertext*, the latter one is denoted as an *open hypertext* [29]. Hypertext is used also in connection with hypermedia, i.e. text is augmented with other media types like pictures, video or audio.

The advantage of the embedded links is that they are bound directly to the information which utilizes the links to access related information. The advantage of the second kind of links is that we can maintain and exchange links which link information in different contexts and possibly for different users, thus providing a more flexible solution ready for personalized access. This separation of text/media items from link structures is now widely accepted in hypermedia systems [18, 17].

Information retrieval systems (especially the content-based ones) rely on index structures with terms from the documents they index. The index structures are used for making retrieval more efficient (see Chapter 10 [31] of this book for more details on content-based recommender systems). Advanced information retrieval systems maintain additional relationships between the index entries. Such structures can be seen as document models which are based on conceptual modeling approaches, semantic net approaches, Bayesian network approaches and so on (see Chapter 5 [6] of this book on document modeling). Open hypermedia research deals with links which are external to the content items. Such links can be seen as indexes of the content helping to browse and navigate the content items they index and map in an efficient way. Therefore, such conceptual structures are related to the document models and information retrieval approaches.

A notion of conceptual open hypermedia has been developed [8, 26, 33, 16]. Conceptual open hypermedia deals with knowledge representation of access structures to content items for particular context from a browsing point of view. Current semantic web technologies are very close to this notion of hypermedia, i.e. they can be used to model and represent such link structures and related objects for reasoning, querying, and processing purposes.

Though the domain ontologies are useful to generate links suited for a particular domain context, with huge corpuses it might result in too many links. Knowledge about

a user might help to further constrain the links, in particular to help with some hints or annotations or simply by hiding links not suitable to his goals. Chapter 1 [6] of this book reviews several approaches to user modeling for personalized access. Semantic web technologies can be utilized for representing, sharing, processing, and reasoning on knowledge about a user in a way similar to the conceptual structures for hyperlinks.

In the following, we will start by reviewing basic hypertext concepts. We will illustrate the concepts by two examples, first with links automatically generated for a page in an eLearning application based on underlying models of content and user, and second with links generated as search results of a user query, also in an eLearning application. We will use these examples throughout our chapter to discuss how to support link generation in those applications. The examples are originally described in [12, 19]. The examples are from operational systems, the personal reader system described in [11] and personal learning assistant in [12]. We then summarize basic principles of the semantic web in terms of representation models and reasoning on the semantic web. We share this idea on reasoning with [14, 10]. Based on this background, we introduce an ontology for providing ontological hypertext links on the semantic web. The links have to be bound to specific resources either manually or as a result of reasoning process. Metadata describing instances of ontological structures are used for the binding purposes as a result of a reasoning process. To support personalized access, knowledge about a user has to be maintained and provided to the link generation systems, and an ontology for a user of an eLearning application is introduced for this purpose. The appropriateness of a resource to be bound to links provided to a user is determined according to a knowledge about the user described by instantiating the user ontology. Finally, we show how links can be generated based on these ontologies and metadata applying semantic web reasoning languages.

23.2 Hypertext and Links

Links in conceptual open hypermedia are usually described as associations between source and target information fragments. The HTML implementation of a link is a bit limited because it refers to target only; i.e. the source of the link is the fragment/page where the link is placed/anchored. The target is identified by the URL which is used to identify pages and fragments on the World Wide Web. Some more advanced applications maintain other information together with the link, for example link type. Some of them allow links to multiple sources and targets and some allow references to other links used as in sources or targets of the links.

To facilitate exchange and reuse of links across hypertext applications, an open hypermedia model has been introduced based on the paradigm of links external to fragments. The light version of the fundamental open hypermedia model [25] treats links as associations between information fragments, as sources of the link and as targets of the link. Fragments are referenced by anchors which are placeholders for information nodes.

Figure 23.1 depicts an example of a link generated (externally to the information fragments) in the *Personal Reader* framework. The link is generated in the left frame

of the picture as a complex association consisting of sublinks. Sublinks are typed, providing different types of resources linked to the currently presented fragment as generalizations, details, summaries, and exercises. Furthermore, the link is annotated by a traffic light metaphor to inform the user which of the resources are ready for him to use, according to his background. The green symbol means that a link is recommended, red that it is not recommended and yellow means that user has to still acquire some prerequisite background needed to access the resource.

Such complex links in the Personal Reader system provide a user participating in a particular course with the context of currently presented information fragments relevant to his/her learning task.

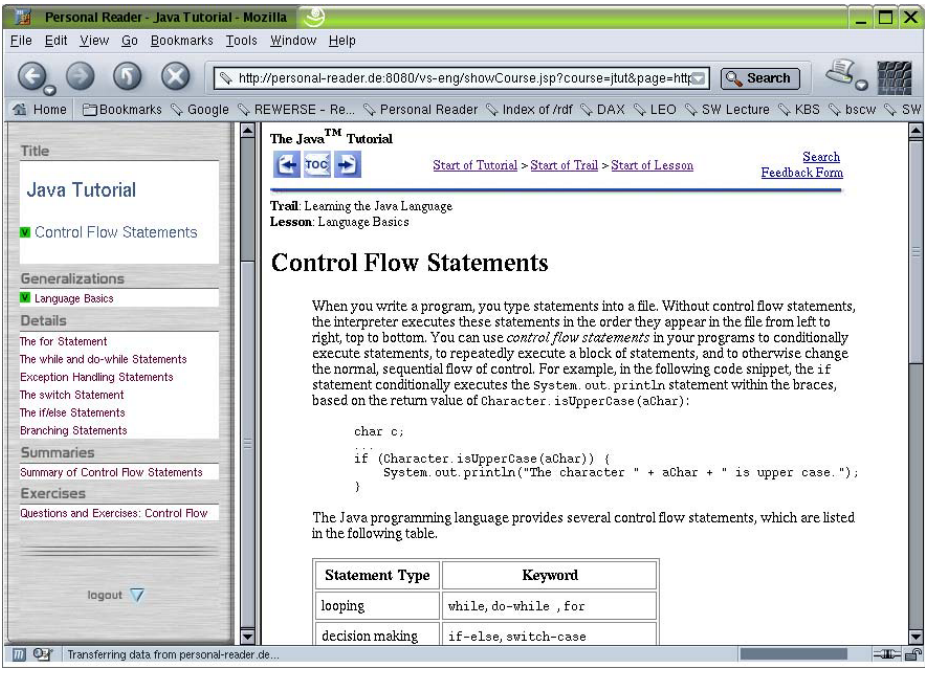

Fig. 23.1. Screenshot of the Personal Reader, showing the adaptive context of a learning resource in a course. The Personal Reader is available at www.personal-reader.de

Another example of link generation, this time in the Personal Learning Assistant (PLA), is depicted in fig. 23.2. Links are generated as search results and point to the resources relevant for a user query. These links are simpler than the one presented in fig. 23.1. Besides the identifier of a resource and its title used to generate the HTML link, it contains further information like the resource description and the concepts described by the resource. Similar to the Personal Reader links, it also provides personalization annotations as traffic light symbols. The concepts and resource descriptions are used to inform a user whether the resource really fits the user query typed at a user interface. Users formulate queries by using the concepts which annotate the resources.

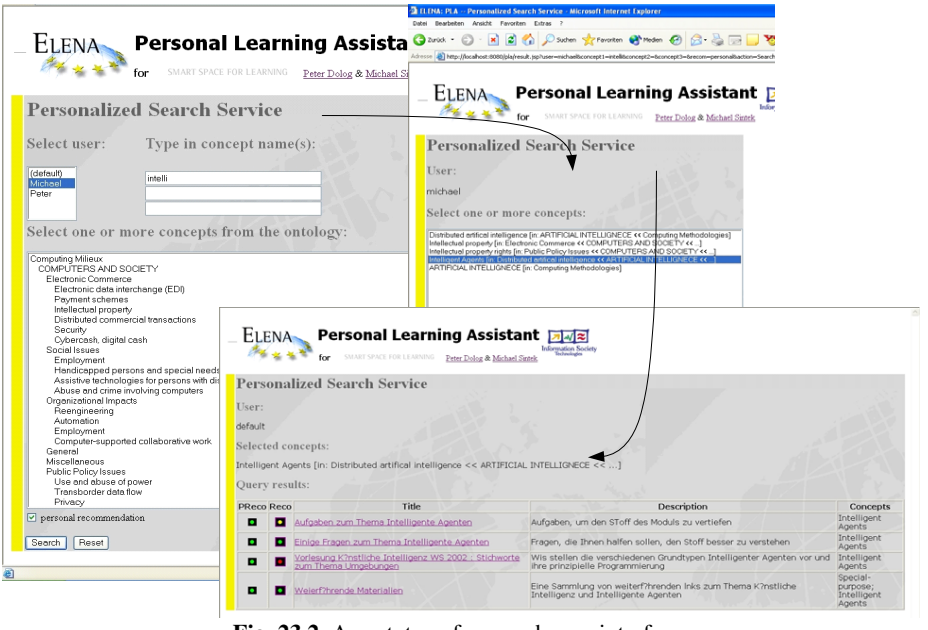

Fig. 23.2. A prototype for search user interface.

In both cases, links form more complex structures than the traditional HTML links to better support users with additional navigation information enabling them to decide which links to follow. Furthermore, links are ordered based on the knowledge about the user background. To be able to generate such links, conceptual structures for such links have to be introduced. In addition, to be able to decide on particular bindings to resources as targets of such links, information about the resources, domain of the resources and the user has to be available.

23.3 Metadata on the Semantic Web

Semantic web technologies like the Resource Description Format (RDF) [23] or RDF Schema (RDFS) [2] provide us with appropriate modeling constructs to model and represent the domain of resources, the resources themselves, as well as users and links. RDF is used to describe specific resources, RDFS serves to define domain-specific vocabularies for the metadata records represented as RDF descriptions. The following paragraphs summarize the basic principles of semantic web representation formats which we will use to describe vocabularies needed for personalized access to web resources. For more information we refer the reader to [9, 34][1].

On the Web, each resource has its own identifier provided, specified as a *Unified Resource Identifier (URI)* which is globally unique. Descriptions about resources are

[1] A reader who is familiar with the sematic web technologies might skip this section

represented as triples of `subject`, `object`, and `predicate`. For example, an assertion about the fact that the homepage of Peter Dolog was created by Peter Dolog is depicted in fig. 23.3.

Fig. 23.3. Example of an RDF graph

The subject of this triple is `http://www.cs.aau.dk/~dolog`, the predicate is `author` and the object is `Peter Dolog` as a literal. Predicates might be defined in different namespaces The URL prefix of `author` is a reference to a Dublin Core namespace in fig. 23.3. The Dublin Core is a standardization initiative for digital libraries metadata and has defined a set of predicates which are used for metadata annotations in the domain.

Object values can be `resources` or `literals`. Literals are strings of text, resources are referenced by URIs. Triples can be embedded in HTML files in an appropriate XML serialization.

Concepts and vocabularies can be provided explicitly on the semantic web and used for these RDF descriptions. The semantic web metadata model distinguishes three types of concepts: *fundamental concepts*, *schema definition concepts*, and *utility concepts*. Each concept has its own identifier in the form of an URI. The concept definitions are grouped into schemas or namespaces which are identified by URIs as well. It is possible to use abbreviated syntax for the concepts where a namespace is abbreviated into a string and separated from the concept identifier by a colon.

The *fundamental concepts* define the RDF triples, providing *rdf:Resource* as a subject, and *rdf:Property* as a predicate. A triple statement can be represented by *rdf:Statement* for reification purposes. These concepts are mandatory for all agents which claim to be developed for and operated on the semantic web.

The *schema definition concepts* are used to define custom vocabularies to be used with metadata descriptions. These concepts are usually domain specific and will be understood just by the domain specific agents, e.g. web applications for particular purposes. The new vocabulary is defined by means of classes (*rdfs:Class*). The classes can be extended with properties by defining a domain of properties (*rdfs:domain*), i.e. their inclusion in a particular class. Properties can be further restricted by defining their range of values (rdfs:Range). Classes and properties can be specialized by using subclassof and subpropertyof predicates (*rdfs:subClassOf* and *rdfs:subPropertyOf*). Any property defines a relation between resources. *rdfs:subPropertyOf* defines a subset of the property range. Similarly, *rdfs:subClassOf* relation between classes is defined as a subset inclusion. Classes define sets of resources of a certain kind. *rdf:type* is used to denote that a resource is an instance of a class or in other words that it belongs to a certain set of resources. Furthermore, typing the resources gives the resource a meaning in a certain context, defined and constrained by a schema.

The *utility concepts* are additional concepts used to define collections and for deploying RDF vocabulary on the web. Collections can be defined by one of the subclasses

of the *rdfs:Container* as a bag (*rdf:Bag*), ordered sequence (*rdf:Seq*), or alternatives (*rdf:Alt*). *rdfs:seeAlso* and *rdfs:isDefinedBy* are used to point to alternative descriptions of a resource. *rdfs:label* and *rdfs:comment* are used to add human readable descriptions of a resource.

The Web Ontology Language (OWL) extends RDFS with restrictions on properties, equality between classes and properties, intersection of classes, property characteristics, 0 and 1 cardinality restrictions, and versioning in its light version. OWL Full and DL (relates to description logic) add class axioms, arbitrary cardinality, filler information, and boolean combinations of class expressions.

23.4 Reasoning on the Semantic Web

Several query and reasoning languages have been introduced to query for, and reason on, metadata on the semantic web such as QEL [27] or SPARQL [15]. The semantics of the languages are often based on Datalog, as used in the Edutella Query Language (QEL) [27, 28], and extended rule and logic programming languages.

QEL offers a full range of predicates in addition to equality, general Datalog rules, and outer join (see [28]). An example for a simple QEL query over resources is the following:

```
s(X, <dc:title>, Y),
s(X, <dc:subject>, S),
qel:equals(S, <java:OO_Class>).
```

The query tries to find resources where `dc:subject` equals `java:OO_Class`. The prefixes `qel:`, `dc:`, and `java:` are abbreviations for URIs of the schemas used. Variable `X` will be bound to URIs of resources, variable `Y` will be bound to titles of the resources, and variable `S` will be bound to subjects of the resources.

A rule language especially designed for querying and transforming RDF models is TRIPLE [32]. Rules defined in TRIPLE can reason about RDF-annotated information resources, translation tools from RDF to TRIPLE and vice versa are provided.

TRIPLE supports *namespaces* by declaring them in clause-like constructs of the form *namespaceabbrev := namespace*, resources can use these namespaces abbreviations.

```
sun_java := "http://java.sun.com/docs/books/tutorial".
```

Statements are similar to frame logic (F-Logic) [22] object syntax: An RDF statement (which is a triple) is written as $subject[predicate \rightarrow object]$. Several statements with the same subject can be abbreviated in the following way:

```
sun_java:'index.html'[rdf:type->doc:Document;
  doc:hasDocumentType->doc:StudyMaterial].
```

RDF *models* are explicitly available in TRIPLE: Statements that are true in a specific model are written as "@model", for example:

```
doc:OO_Class[rdf:type->doc:Concept]@results:simple.
```

Connectives and quantifiers for building logical formulae from statements are allowed as usual, i.e. $\wedge$, $\vee$, $\neg$, $\forall$, $\exists$, etc. For TRIPLE programs in plain ASCII syntax, the symbols `AND`, `OR`, `NOT`, `FORALL`, `EXISTS`, `<-`, `->`, etc. are used. All variables must be introduced via quantifiers.

23.5 Ontologies and Metadata for Personalized Access

23.5.1 Link Structures

An ontology for link structures is used to describe structures relevant for visualization. Such an ontology adapted from FOHM [25] is depicted in fig. 23.4.

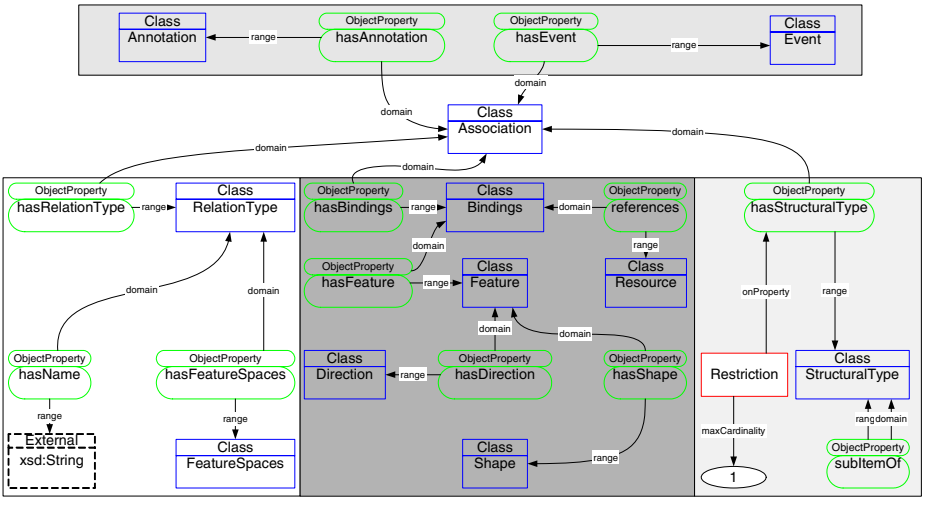

Fig. 23.4. An excerpt of the link ontology based on FOHM [25]

The main element of the ontology is the `Association` which links the information fragments/ pages which are relevant. Like in [25], the `Association` is built from three components: `Bindings`, `RelationType`, and `StructuralType` (in FOHM the association is a Cartesian Product of bindings, relation type and structural type). These three components (classes) are related to association through `hasBindings`, `hasRelationType`, and `hasStructuralType` properties.

A `StructuralType` is either a stack, link, bag, or sequence of resources. They are specialized forms of a general `Structure`. We use a `subItemOf` property for hierarchy specification (see fig. 23.5). The `Association` is restricted to have exactly one `StructuralType`.

`Bindings` references a particular `Resource` on the web (document, another association, etc.), and `Feature`-s. A `Feature` can be a `Direction`, `Shape`, etc. Entries for `Direction` are depicted in fig. 23.6b, entries for `Shape` are depicted in fig. 23.6c. The `RelationType` has a `Name` which is a string. The `RelationType`

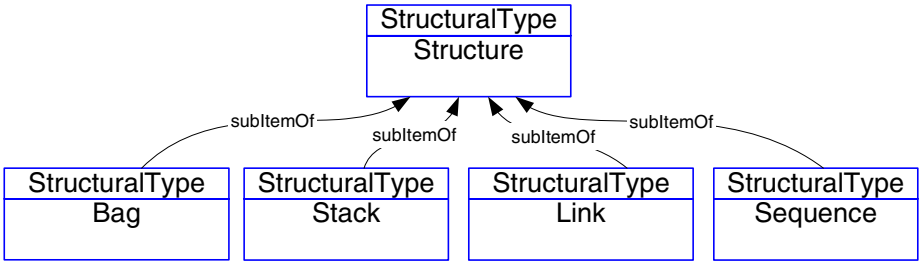

Fig. 23.5. Ontology for Structural Types.

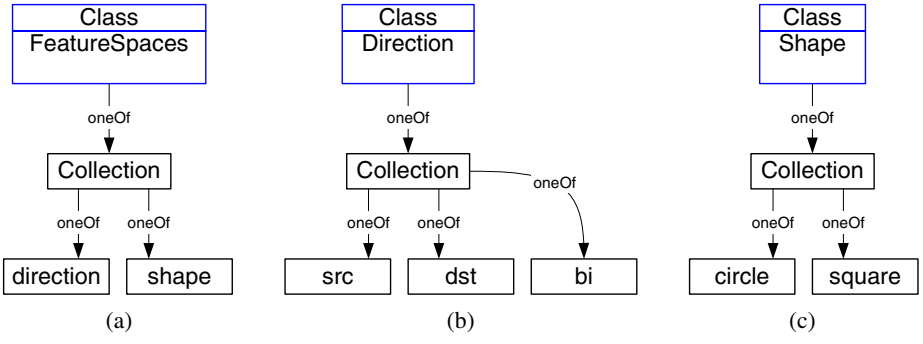

Fig. 23.6. Members of Collection of: (a) Feature Spaces, (b) Direction, (c) Shape.

also points to the `FeatureSpaces`. Entries for the `FeatureSpaces` are depicted in fig. 23.6a.

In addition, `Association` can have associated events (e.g. click events for processing user interactions) through the `hasEvent` property, and an annotation (e.g. green/red/yellow icon from traffic light metaphor technique from adaptive hypermedia [3]) through `hasAnnotation` property.

The `hasEvent` property defines an event which is provided within the document (to be able to get appropriate observation). Whenever the event is generated observation reasoning rules assigned to this type of event are triggered. The `represents` property references a resource, which is stored in observations about the learner, after an event is generated.

FOHM introduces *context* and *behavior* objects. Filtering and contextual restrictions maintained by the *context* objects in FOHM are substituted by richer reasoning language and rules in our approach. On the other hand, interactions and observations together with events substitute the notion of *behavior* objects.

Let us recall our two examples discussed in sec. 23.2, the Personal Reader and PLA. The links which are depicted there can be described using our ontology for link structures.

Figure 23.7 depicts an excerpt of a link structure visualized in fig. 23.1. The boxes represent instances (objects) and links represent specific relations between them. The box slots represent instantiations of the class attributes. The toolbar of the per-

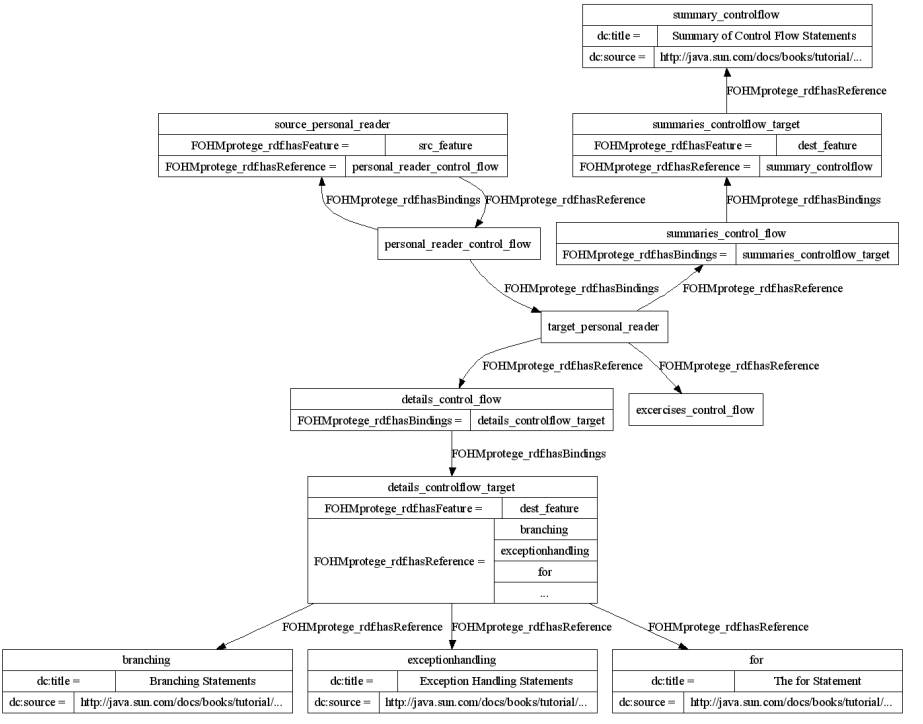

Fig. 23.7. An excerpt of a link instance for the Personal Reader and resource depicted in fig. 23.1

sonal reader is represented as a complex link (`personal_reader_control_flow`) pointing to sublinks for each part: generalizations (collapsed), details (`details_control_flow`), summaries (`summaries_control_flow`), and exercises (`exercise_control_flow`). These are treated as targets of the personal reader link and are instances of the ontology class `Association`. Furthermore, the link also contains a source (`source_personal_reader`). Each of the associations has a binding to its features which point to direct resources. For example, the `detail_control_flow` has a destination feature pointing to resources which are then used to generate click-able HTML links, i.e. URLs of web pages describing the JAVA language constructs for branching, exception handling, cycles (e.g., FOR, WHILE), and so on. The PLA links are represented similarly.

Note that the Personal Reader and PLA are just two examples of visualization agents of such links. The instantiated links can be stored for exchange and search purposes and visualized by other user interface agents in many different ways.

The resources bound to the links refer to resource metadata like title, source and others in this case from Dublin Core namespaces. They have to be selected and bound to such a link. The regeneration program is invoked whenever a user interacts with the link, i.e., the link is annotated with additional events to store user behavior and to invoke a program for regeneration.

To be able to select and bind resources to the links through its features, they have to be described in a certain way. The ontologies and metadata serve to represent knowledge about the resources and users to be used for the generation and visualization purposes.

23.5.2 Information Resources and Users

Specific domain information is usually described by concepts and their mutual relationships. The semantic web vocabularies (ontologies) in RDFS or OWL serve as domain specific models [19]. Domain ontologies consist of classes (classifying objects from a domain) and relationships between them.

The ontologies are used in annotations of specific documents/resources. The annotation metadata serves as knowledge about domain information, information fragments composition or index, and navigation which involve particular resources. In other words, the vocabulary defined by the domain specific ontologies are used to annotate/index information fragments, their compositions, and possible navigation directions in them.

The metadata can be created by the authors of information fragments or in some cases generated automatically. The ontologies described can be used to bias the descriptions of the resources and index them by the concepts from the ontology based on document analysis techniques. We have performed an experiment of automatic extraction of metadata within the framework of Personal Reader for realizing the global context. The external resources (in this case Java API) were indexed by terms from the java tutorial subject ontology (see Chapter 5 [24] and Chapter 10 [] for details on document analysis and modeling techniques which can be used to extract terms from document resources). To improve search results, a JAVA API ontology has been learned and used to cross annotate the java tutorial pages. In the following we show some examples of metadata which are created to support link generation and search.

Resource Indexing

Subject Ontologies. Subject ontologies represent organization of concepts, topics, knowledge items, or competencies in a particular domain. In the eLearning domain, the subject ontologies represent usually the domain to be taught. The concepts from ontologies are used to index information to be presented to a user and for retrieval purposes.

Figure 23.8 depicts an example of such a subject ontology, an excerpt of the java programming domain. We show a fragment of a domain knowledge base covering Java programming concepts with `isa` (subConceptOf) relationships between these concepts. Figure 23.8 depicts the `Programming_Strategies` concept with its subconcepts: `Object_Oriented`, `Imperative`, `Logical`, and `Functional`. The `Object_Oriented` concept is further specialized to `OO_Class`, `OO_Method`, `OO_Object`, `OO_Inheritance`, and `OO_Interface`. Other relations between concepts might be useful for personalization purposes as well, e.g. sequencing or dependency relations.

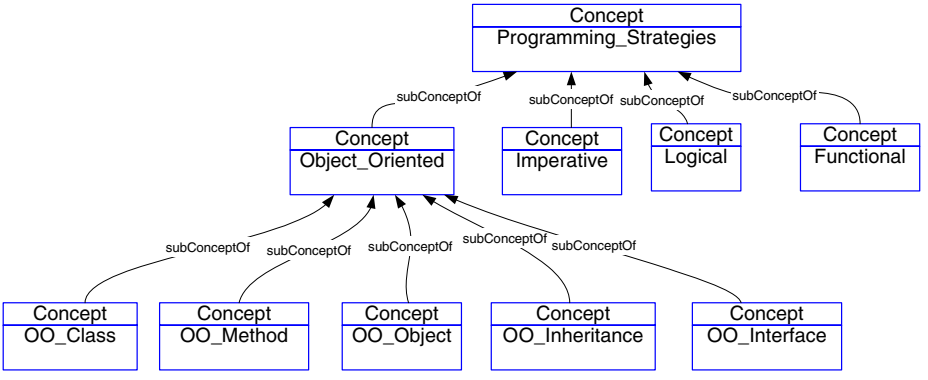

Fig. 23.8. An excerpt of application domain ontology for Java e-lecture

Resource Description Ontologies. The resource description ontologies represent the organization of metadata about resources on the web. They specify attributes which are used to describe resources and classes which categorize them.

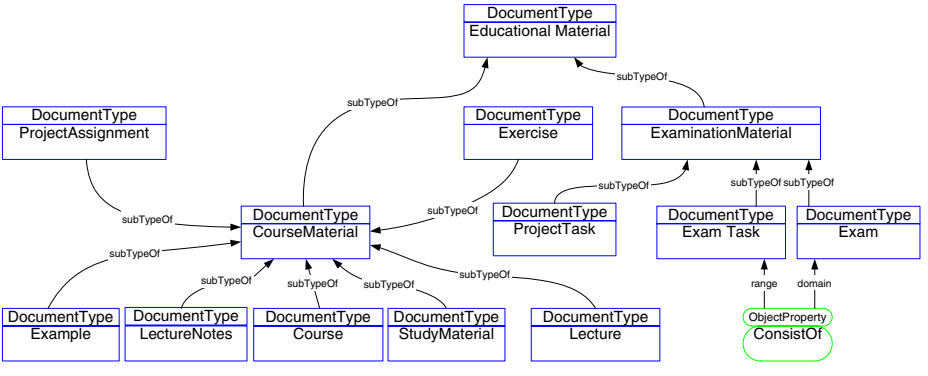

Fig. 23.9. An excerpt of environment ontology as document types hierarchy for eLearning applications

An example of a resource description ontology is depicted in fig. 23.9. The ontology depicts document types in the educational domain. The most general document type is `EducationalMaterial`. `EducationalMaterial` has two subtypes: `CourseMaterial` and `ExaminationMaterial`. `ExaminationMaterial` can be further specialized to `ProjectTask`, `ExamTask`, and `Exam`. The `Exam` can consist of the `ExamTask`-s. `CourseMaterial` can be further specialized into `Lecture`, `Example`, `LectureNote`, `Course`, `Exercise`, and `ProjectAssignment`.

An ontology for documents and their relationships to other components is depicted in fig. 23.10. The ontology represents a context of learning material which is usually provided as a document. The class `Document` is used to annotate a resource which is a

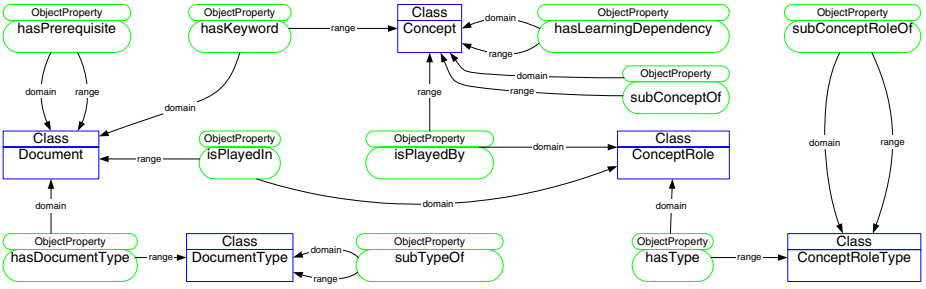

Fig. 23.10. An excerpt of environment domain ontology of documents

document. Documents describe concepts; we use class `Concept` to annotate concepts. Concepts and documents are related through the `hasKeyword` property.

Documents can be ordered by the `hasPrerequisite` property. There can be different ordering for and within applications, so multiplicity is allowed. The hasPrerequisite property is intended for navigation purposes.

Concepts play certain roles in particular document fragments. For example some concepts represent the crucial information, i.e. they are of the main information serving goal, while the others can just play a concretization role or role of comparison. In the ontology, we represent these facts by instances of `ConceptRole` class and its two properties: `isPlayedIn` and `isPlayedBy`. Document properties can be further extended by assigning a `DocumentType`. Similarly, the roles can be further extended by specifying their types. Concepts, concept role types, and document types can form hierarchies. We define `subTypeOf`, `subConceptRoleOf`, and `subConceptOf` properties for these purposes.

Information Composition and Indexing. The topics, concepts or competencies from subject ontologies represent specific content realization or composition in particular resources. An example of such a resource is a page describing sun_java: 'java/concepts/class.html'. The following example shows how such a page can be annotated based on the above mentioned resource and subject ontologies.

```
sun_java:'java/concepts/class.html'[rdf:type->doc:Document;
  hasTopic->doc:OO_Class].
doc:OO_Class[rdf:type->doc:Concept;
  doc:subConceptOf->doc:Classes_and_objects].
doc:ClassesIntroduction[rdf:type->doc:ConceptRole;
  doc:isPlayedBy->doc:OO_Class;
  doc:isPlayedIn->sun_java:'java/concepts/class.html';
  doc:hasType->doc:Introduction].
doc:Introduction[rdf:Type->doc:ConceptRoleType;
  doc:subConceptRoleOf->doc:Cover].
```

The page is a document (RDF type `Document`). The type specifies which environment the documents can be accessed through. The document describes information about classes (`OO_Class` concept). The `OO_Class` concept is annotated with type `Concept` and is a subconcept of the `Classes_and_objects` concept.

The relations and roles of concepts in particular information resources are represented by the ClassesIntroduction resource which is of type Concept-Role. The OO_Class concept plays a role of introduction (the Introduction role type) in the document which is annotated by using properties isPlayedBy and isPlayedIn respectively by references to OO_Class concept and the document. The Introduction is of type ConceptRoleType and means that the concept is covered by the content to a certain extent. Therefore, the Introduction is a subtype of Cover concept role type — a generic role type for stating that a concept is covered by document content.

Pedagogical prerequisites are encoded in the metadata to state which knowledge a user should have when accessing particular resources, (hasPrerequisite or inverse property isPrerequisiteFor of a concept or resource). In our example, the OO_Class concept is a prerequisite for the OO_Inheritance. Therefore, the above mentioned example is extended with the instance of this property.

```
sun_java:'java/concepts/class.html'[rdf:type->doc:Document;
  hasTopic->doc:OO_Class].
doc:OO_Class[rdf:type->doc:Concept;
  doc:subConceptOf->doc:Classes_and_objects;
    doc:isPrerequisiteFor->doc:OO_Inheritance].
doc:ClassesIntroduction[rdf:type->doc:ConceptRole;
  doc:isPlayedBy->doc:OO_Class;
  doc:isPlayedIn->sun_java:'java/concepts/class.html';
  doc:hasType->doc:Introduction].
doc:Introduction[rdf:Type->doc:ConceptRoleType;
  doc:subConceptRoleOf->doc:Cover].
```

User Modeling with Ontologies

User Ontologies. User modeling is used to gather knowledge about a user for personalization purposes. Personalization (or user-centered adaptation) decides about the presentation of variable resources and links on the web based on knowledge about a user. Data about a user serves to derive contextual structures. It is used to determine how to adapt the presentation of hypertext structures. In the eLearning domain, an ontology for a user profile based on IEEE [20] and IMS Global Consortium (e.g. [21]) specifications can be used. *Preference* indicates the types of devices and objects, which the user is able to recognize. Learner *Performance* and *Preference* are the main aspects relevant for personalization. Learner performance may further contain references to his *Portfolio* of projects, documents created, and experiences gained. For more discussion on learner modeling standards see for example [13].

Figure 23.11 depicts an example of an ontology for learner profiles. Learner performance is maintained according to a class Performance. Performance is based on learning experience (learningExperienceIdentifier), which is supported by particular documents. Experience implies a Concept learned from the experience, which is represented by learningCompetency property. Performance is certified by a Certificate, which is issued by a certain Institution. Performance has a certain PerformanceValue, which is in this context defined as a floating point number and restricted to the interval from 0 to 1.

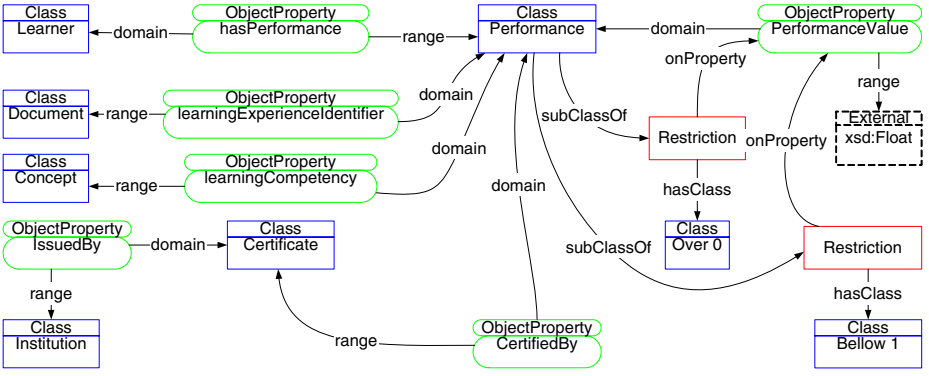

Fig. 23.11. Ontology for learner performance

Observations About a User. At run time, users interact with a web system. User interactions can be used to draw conclusions about possible user interests, user goals, tasks, knowledge, etc. These conclusions can be used for providing personalized views on hypertexts. An ontology of observations should therefore provide a structure of information about possible user observations, and - if applicable - their relations and/or dependencies.

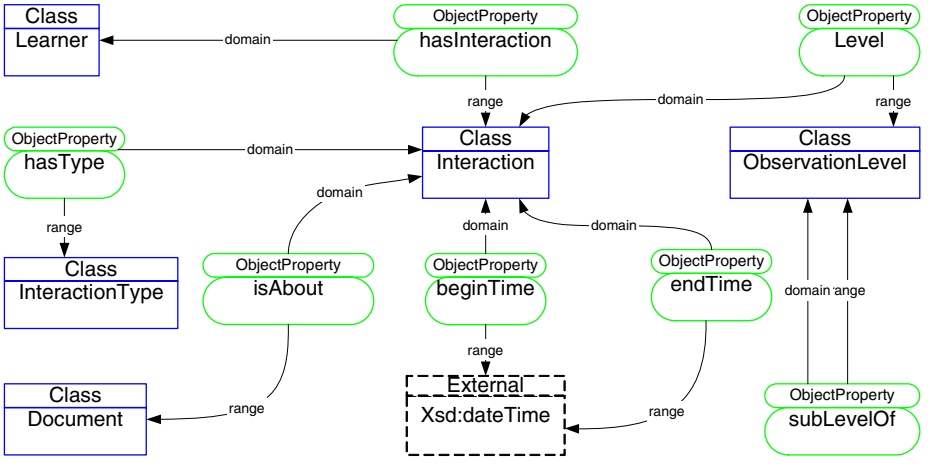

Fig. 23.12. Ontology for observations

A simple ontology for observations is depicted in fig. 23.12. The ontology allows us to state that a `Learner` interacted (`hasInteraction` property) with a particular `Document` (`isAbout` property) via an interaction of a specific type (`InteractionType`). Example of `InteractionTypes` are `access` or `bookmark`. The information that an interaction has taken place during a time interval is maintained by `beginTime` and `endTime` properties. The `ObservationLevel` describes particu-

lar activity types representing the purpose of the interaction. Examples for Observation-Levels are that a user has `visited` a page, has `worked` on a project, or has `solved` some exercise.

Runtime User Model. Based on these ontologies, a run-time user model can be derived, stored, maintained and used for personalization. The run-time user model is an instance of a user domain model selected for a particular application. For example, if a learner interacts with a course on JAVA, his learning performance is derived from the pages he has visited and the concepts he has worked with at particular pages. These concepts are taken from the metadata which annotate these pages. They are represented in a domain ontology for a learning outcome. Furthermore, if a page is linked to a learner assessment on particular topics, the results taken from such a learner assessment can be classified similarly as the pages and their metadata used to instantiate a learner performance record.

Let's take an example of a learner maintained as `user2` in a system. He has a performance record (maintained with `user2P` identifier in a system). Performance contains learning experience about the KBS Java objects resource. The concept covered in the resource is also stored in performance. A certificate about the performance with performance value and institution that issued the certificate is recorded in learner performance as well. Such a model in an RDF format would look like as follows:

```
user:user2[rdf:type -> learner:Learner;
  learner:hasPerformance -> user:user2P].
user:user2P[rdf:type->learner:Performance;
  learner:learningExperienceIdentifier->
     sun_java:'java/concepts/object.html';
  learner:learningCompetency->doc:OO_Object;
  learner:CertifiedBy->KBScerturi:C1X5TZ3;
  learner:PerformanceValue->0.9].
KBScerturi:C1X5TZ3[rdf:type->learner:Certificate;
  learner:IssuedBy->KBSuri:KBS].
KBSuri:KBS[rdf:type->learner:Institution].
```

23.6 Generating Links from Metadata

As discussed above, the link and hypertext paradigm can be employed in searching and in browsing. Links can be generated with the help of knowledge encoded in metadata, extracted from resources and biased by agreed upon ontologies and standards. In both cases, the process of link generation consists of several steps. Figure 23.13 describes examples of activities which support the interaction with an adaptive eLearning system. The user has the possibility of defining a learning goal, or the goal is defined implicitly by a lecture as in the Personal Reader system. In the Personal Learning Assistant, the user needs to define a learning goal by selecting some concepts from an appropriate ontology. The ontology contains competencies, skills, or concepts to be learned as described above.

If the user selects concepts from formal ontologies, a `Query` in a language appropriate for the repository can be constructed directly. If the user typed free text (into

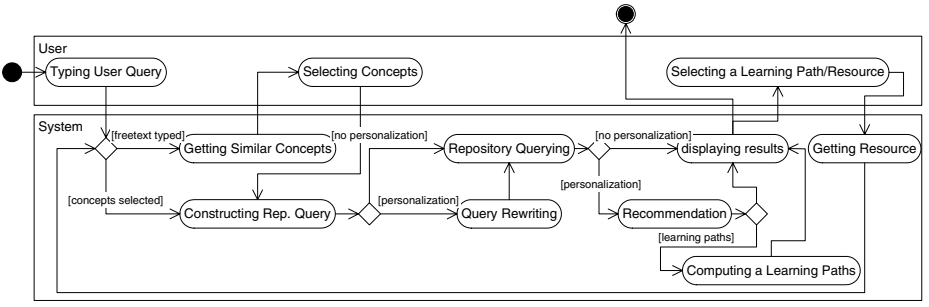

Fig. 23.13. Activities in adaptive system

fields for competencies) the system has to provide similar concepts from the ontologies for the purpose of refining his query.

After formulating such a personal learning goal, the system constructs the first version of a `Repository Query` which searches for appropriate learning resources or services. The query in Personal Reader is based on metadata on the currently presented resource. In addition, if the user requires personalization features, the query has to be rewritten taking the user profile into account (`Query Rewriting`). Such a query can then be sent to a repository.

Results returned from the repository can be either processed for display (`Displaying Results`), or the results are postprocessed by personalization algorithms (`Recommendation`) and learning path planning algorithms (`Computing Learning Path`). In both cases, the returned resources are used for generating bindings in the associations from the link ontology depicted in fig. 23.4. The sequencing information and similarities between topics in resources are used to order the bindings in associations when presented to the user. Associations are generated either based on predefined templates for user queries (specifying which information to present) or according to the local neighborhood given by several relations in the case of browsing. In both cases, annotations are used to express personalization/recommendation information. The structural types for ordering the resources bound and the visualization types represented by feature space, direction and shape are part of the specification for a particular application.

Query Rewriting. Since annotations of web resources will often vary (simpler ontologies, missing metadata, and even inconsistent metadata), we need heuristics to construct queries that cope with these difficulties. If the exact query returns no or too few results, the query needs to be relaxed by replacing some restrictions with semantically similar (usually, more general) ones, or by dropping some restrictions entirely. For this, we also need a strategy to decide which attributes to relax first (e.g., first relax dc:subject, then relax type).

The following TRIPLE predicate `similar_concept(C, CS, D)` shows how to enumerate, for a given concept `C`, similar concepts `CS` by traversing the underlying ontology and extracting superconcepts, subconcepts, and siblings with a given maximum distance `D` from `C` in the ontology. We assume that the predicate `direct_super` connects concepts with their direct superconcepts.

```
FORALL C, CS similar_concept(C, CS, 1) <- // direct super/subconcept
   direct_super(C, CS) OR direct_super(CS, C).
FORALL C, CS, D, D1  similar_concept(C, CS, D) <- // recurse
   D > 1 AND D1 is D - 1 AND similar_concept(C, CS1, D1) AND
   (direct_super(CS, CS1) OR direct_super(CS1, CS))
   AND not unify(C, CS).
```

This predicate is used iteratively to relax the query: get all similar concepts with `D` = 1, relax the query (by query rewriting), and send it to the remote repositories. If the returned result set is empty or too small, increment `D` and reiterate. The maximum number of iterations should be significantly smaller than the hierarchy depth of the ontology to avoid completely meaningless results.

Queries can also be expanded by additional restrictions from the user profile (e.g. language preferences). We have implemented a query rewriting service which adds additional constraints to a QEL query created based on the concepts selected by a user. These constraints reflect concepts and language preferences maintained in user profiles.

We illustrate query rewriting on the following simple restriction profile, implemented in TRIPLE.

```
@edu:p1 {
    edu:add1[rdf:type -> edu:AddSimpleRestriction;
        rdf:predicate -> dc:lang;
        rdf:object -> lang:de].

    edu:add2[rdf:type -> edu:AddTopicRestriction;
            edu:addTopic -> acmccs:'D.1.5'].}
```

This heuristic is used to extend a QEL query with a constraint which restricts the results to learning resources in German language (restriction `edu:add1`).

Another restriction derived from the user profile is a restriction on resources about *object-oriented programming* (`edu:add2`). The ACM Computer Classification System [30] is used to encode the subject. In that classification system, the *object-oriented programming* can be found in the category `D` representing *software*. The subcategory `D.1` represents *programming techniques* with the fifth subcategory being *object-oriented programming*. Heuristics for query rewriting especially in case of concept or subject restrictions are usually more complex. They depend on concepts being selected or typed as a user query.

The derived restrictions profile is used in a TRIPLE view which takes as an input the profile and QEL query model. The following illustrates one of the rules for reasoning over language restrictions profiles. The view `@edu:p1` encapsulates the restrictions model.

```
FORALL QUERY, VAR, PRED, OBJ, NEWLIT
 QUERY[edu:hasQueryLiteral -> edu:NEWLIT] AND
 edu:NEWLIT[rdf:type -> edu:RDFReifiedStatement;
            rdf:subject -> VAR;
            rdf:predicate -> PRED;
            rdf:object -> OBJ]
 <-
 EXISTS LITERAL, ANY (
```

```
QUERY[rdf:type -> edu:QEL3Query;
      edu:hasQueryLiteral -> LITERAL]
AND
LITERAL[rdf:type -> edu:RDFReifiedStatement;
        rdf:subject ->
           VAR[rdf:type -> edu:Variable];
        rdf:predicate -> dc:ANY])
AND
EXISTS A
  A[rdf:type -> edu:AddSimpleRestriction;
    rdf:predicate -> PRED;
    rdf:object -> OBJ]@edu:p1
AND
unify(NEWLIT, lit(VAR,PRED,OBJ)).
```

Recommendation Annotations. Recommendations can be expressed as an additional property of a resource; i.e. can annotate learning resources according to their educational state for a user. The recommendation property can take on the value of *recommend*, which specifies that a resource is recommended to a specific user, or can take a weaker value of recommendation like *might be understandable*. It can be a *not recommend* learning resource or point out that this learning resource leads to a page that the user has already visited.

To derive appropriate recommendation annotations for a particular user, prerequisite concepts for a learning resource have to be mastered by the user. The `lr:isPrerequisiteFor` relationships of concepts covered in a learning resource are analyzed for this purpose. On the other hand, a user performance profile and competencies acquired and maintained in that profile are analyzed in comparison to the prerequisites of particular learning resource.

One example of a recommendation rule is a rule which determines learning resources which are `Recommended`. A learning resource is recommended if *all* prerequisite concepts of all of concepts it covers have been mastered by a user:

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

Predicates used in the rule derive concepts like learning resource, concepts, users, observations and learning states from metadata based on types taken from ontologies described above. We have implemented other rules to compute less strong recommendations. This includes for example a recommendation that a resource `Might_be_understandable` if at least one prerequisite concept has been learned.

This kind of recommendation can be used as a link annotation technique in the area of adaptive hypermedia [7], or to annotate query results with the recommendation information. On the user interface side, it is often implemented using the already mentioned traffic lights metaphor.

23.7 Discussion and Conclusions

This chapter discussed adaptive navigation support with the help of ontologies. The information access on the Web is realized through a hypertext paradigm, i.e. through provision of links. The links which are provided directly by an author usually reflect a particular context the author had when he created them. On the other hand, link generation procedures based on pure document analysis techniques may result in too many links.

The ontologies as shared conceptual models of the domain, provide context of that domain, i.e. they can be used to generate links relevant for the domain. They may serve as an input to the document analysis algorithms to take only those terms similar to the concepts from the ontology into account. Furthermore, the ontologies for a user help to further restrict the set of links which are generated only to the ones a user is interested in the most or annotate them appropriately.

The semantic web technologies in this context provide the following advantages:

– improved interoperability,
– explicit semantics,
– formal representation,
– formal reasoning.

Information resources are provided by several independent systems used in a specific context. The semantic web representation models provide uniform ways to describe, share and exchange knowledge about information resources, domains (subjects) they describe, users who use them and further knowledge needed and acquired in those systems automatically or semi-automatically. Therefore, those systems are able to *interoperate better* providing users with an extended access to information resources. In this chapter, we have shown how the ontologies represented in the semantic web format for subject, resource, user, and link can be used to realize the personalized access to information on the semantic web.

Subject ontologies which are used to index the information resources provide the *explicit semantics* about the information resource discourse which helps systems to better understand how they fit to user query, goal and background. Furthermore, user profile ontologies in semantic web representation format provide an *explicit semantics* about certain user aspects, his activities, and features what helps to improve personalization and a user satisfaction.

The semantic web technologies provide *formal representation* for knowledge on the web, thus enabling *formal reasoning* on top of them. Therefore, deduction rules can be employed for personalization. Observations about users are used to bias and reorder resources bound to the links. User models are used for personalized selection of resources. We have shown how rule-based reasoning techniques can be applied to generate such links and annotate them with recommendation information. The principles described in this chapter are general purpose but illustrated for the eLearning domain. For different domains, different ontologies have to be used, but applied similarly as we have shown in the examples from the eLearning domain.

There are three main aspects for further research challenges in this area: knowledge representation, technological, and computational. From the knowledge representation

point of view, procedural knowledge in addition to the propositional knowledge about content and user is important especially in business domains and collaborative learning in workplaces. The procedures which correspond to problem solving and are related to a user's activity can be used to guide them through the problem according to real workplace settings and workflows. The connection between procedural and propositional knowledge and personalization has to be further studied.

From the technological point of view, heterogeneity of information resource is a big challenge. Information integration and approximation approaches are possibly relevant when searching large collections of heterogeneous information sources. From a practical point of view, another challenge is how to combine statistical, information retrieval models with reasoning techniques while still employing semantic web technologies. We have shown certain combinations of document analysis techniques, used to re annotate web resources of JAVA API, and formal reasoning on top of generated metadata. Further investigations are needed in this context.

From the computational point of view, performance of the reasoners is a big issue, especially when considering large semantically interconnected collections of objects on the web. For practical applications, the performance issues related with reasoning should be researched.

References

1. Agosti, M., Crestani, F., Melucci, M.: Design and implementation of a tool for the automatic construction of hypertexts for information retrieval. Inf. Process. Manage. **32**(4) (1996) 459–476
2. Brickley, D., Guha, R.V.: Resource Description Framework (RDF) Schema Specification 1.0 (2002) `http://www.w3.org/TR/rdf-schema`.
3. Brusilovsky, P.: Methods and techniques of adaptive hypermedia. User Modeling and User-Adapted Interaction **6**(2-3) (1996) 87–129
4. Brusilovsky, P.: Adaptive navigation support. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization. Volume this volume of Lecture Notes in Computer Science., Springer Verlag (2007)
5. Brusilovsky, P., Maybury, M.T.: From adaptive hypermedia to the adaptive web. Commun. ACM **45**(5) (2002) 30–33
6. Brusilovsky, P., Millán, E.: User models for adaptive hypermedia and adaptive educational systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization. Volume this volume of Lecture Notes in Computer Science., Springer Verlag (2007)
7. Brusilovsky, P., Nejdl, W.: Adaptive hypermedia and adaptive web. In Singh, M., ed.: Practical Handbook of Internet Computing. CRC Press (2004) 1–1 – 1–12
8. Bruza, P.D.: Hyperindices: a novel aid for searching in hypermedia. (1992) 109–122
9. Champin, P.A.: Rdf tutorial. `http://www710.univ-lyon1.fr/ champin/rdf-tutorial/rdf-tutorial.html` (April 2001)
10. Denaux, R., Dimitrova, V., Aroyo, L.: Integrating open user modeling and learning content management for the semantic web. In Ardissono, L., Brna, P., Mitrovic, A., eds.: User Modeling 2005, 10th International Conference, UM 2005. Volume 3538 of Lecture Notes in Computer Science., Edinburgh, Scotland, UK, Springer (July 2005) 9–18

11. Dolog, P., Henze, N., Nejdl, W., Sintek, M.: The personal reader: Personalizing and enriching learning resource using semantic web technologies. In Nejdl, W., Bra, P.D., eds.: Proc. of AH2004 — International Conference on Adaptive Hypermedia. Volume 3137 of LNCS., Einghoven, The Netherlands, Springer (August 2004) 85–94
12. Dolog, P., Henze, N., Nejdl, W., Sintek, M.: Personalization in distributed e-learning environments. In: Proc. of WWW2004 — The Thirteen International World Wide Web Conference, New Yourk, ACM Press (May 2004) 170–179
13. Dolog, P., Schäfer, M.: A framework for browsing, manipulating and maintaining interoperable learner profiles. In Ardissono, L., Brna, P., Mitrović, A., eds.: Proc. User Modeling 2005: 10th International Conference, UM 2005. Volume 3538 of LNAI., Edinburgh, Scotland, UK, Springer (July 2005) 397–401
14. Domingue, J., Dzbor, M.: Magpie: supporting browsing and navigation on the semantic web. In Vanderdonckt, J., Nunes, N.J., Rich, C., eds.: Proceedings of the 2004 International Conference on Intelligent User Interfaces, Funchal, Madeira, Portugal, ACM (January 2004) 191–197
15. for RDF, S.Q.L.: Sparql query language for rdf (2006) published online at `http://www.w3.org/TR/rdf-sparql-query/`.
16. Goble, C., Bechhofer, S., Carr, L., Roure, D.D., Hall, W.: Conceptual open hypermedia = the semantic web? In Decker, S., Fensel, D., Sheth, A., Staab, S., eds.: Proceedings of the SemWeb2001, The Second International Workshop on the Semantic Web at World Wide Web Conference — WWW10, Hong Kong (May 2001) Available at: `http://CEUR-WS.org/Vol-40/Goble-et-al.pdf`.
17. Grønbæk, K., Trigg, R.H.: Design issues for a dexter-based hypermedia system. Commun. ACM **37**(2) (1994) 40–49
18. Hall, W.: Ending the tyranny of the button. IEEE MultiMedia **1**(1) (1994) 60–68
19. Henze, N., Dolog, P., Nejdl, W.: Towards personalized e-learning in a semantic web. Educational Technology and Society Journal. Special Issue on Ontologies and the Semantic Web for E-learning **7**(4) (October 2004) 82–97
20. IEEE: IEEE P1484.2/D7, 2000-11-28. draft standard for learning technology. public and private information (papi) for learners (papi learner) (2000) Available at: `http://ltsc.ieee.org/archive/harvested-2003-10/working_groups/wg2.zip`. Accessed on December 20, 2003.
21. IMS: IMS learner information package specification Available at: `http://www.imsproject.org/profiles/index.cfm`. Accessed on October 25, 2002.
22. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. J. ACM **42**(4) (1995) 741–843
23. Lassila, O., Swick, R.: W3C resource description framework (rdf) model and syntax specification Available at: `http://www.w3.org/TR/REC-rdfsyntax/`. Accessed on October 25, 2002.
24. Micarelli, A., Sciarrone, F., Marinilli, M.: Web document modeling. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization. Volume this volume of Lecture Notes in Computer Science., Springer Verlag (2007)
25. Millard, D.E., Moreau, L., Davis, H.C., Reich, S.: FOHM: a fundamental open hypertext model for investigating interoperability between hypertext domains. In: Proceedings of the eleventh ACM on Hypertext and hypermedia, ACM Press (2000) 93–102
26. Nanard, J., Nanard, M.: Using structured types to incorporate knowledge in hypertext. In: HYPERTEXT '91: Proceedings of the third annual ACM conference on Hypertext, New York, NY, USA, ACM Press (1991) 329–343
27. Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., Risch, T.: EDUTELLA: a P2P Networking Infrastructure based on RDF. In: In Proc. of 11th World Wide Web Conference, Hawaii, USA, ACM Press (May 2002) 604–615

28. Nilsson, M., Siberski, W.: RDF Query Exchange Language (QEL) - Concepts, Semantics and RDF Syntax. Available at: http://edutella.jxta.org/spec/qel.html. Accessed: 20th September 2003 (2003)
29. Nürnberg, P.J., Leggett, J.J., Wiil, U.K.: An agenda for open hypermedia research. In: HYPERTEXT '98. Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia: Links, Objects, Time and Space - Structure in Hypermedia Systems, Pittsburgh, PA, USA, ACM (June 1998) 198–206
30. of Computing machinery, A.: The acm computer classification system. http://www.acm.org/class/1998/ (2002)
31. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization. Volume this volume of Lecture Notes in Computer Science., Springer Verlag (2007)
32. Sintek, M., Decker, S.: TRIPLE—A query, inference, and transformation language for the semantic web. In Horrocks, I., Hendler, J.A., eds.: Proc. of ISWC 2002 — 1st International Semantic Web Conference. Volume 2342 of LNCS., Sardinia, Italy, Springer (June 2002) 364–378
33. Tudhope, D., Taylor, C.: Navigation via similarity: automatic linking based on semantic closeness. Inf. Process. Manage. **33**(2) (1997) 233–242
34. W3C: Owl web ontology language semantics and abstract syntax. Technical Report. Available at: http://www.w3.org/TR/owl-semantics/. Accessed: 20th September 2003 (August 2003)
35. Weiss, R., Vélez, B., Sheldon, M.A., Namprempre, C., Szilagyi, P., Duda, A., Gifford, D.K.: Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In: Hypertext '96, The Seventh ACM Conference on Hypertext, Washington DC, USA (March 1996) 180–193