A. Grigorov, A. Georgiev, P. Anagnostou. Building Owl Ontology in Fedora Digital Repository. В: Т. Атанасова (Редактор), Сборник "Моделиране и управление на информационните процеси", София, България, 2010, КТП, София, 2010, стр. 94-102

BUILDING OWL ONTOLOGY IN FEDORA DIGITAL REPOSITORY

Alexander Grigorov[1,2], Atanas Georgiev[1], Panagiotis Anagnostou[1]

[1] Faculty of Mathematics and Informatics, St. Kl. Ohridski University of Sofia

[2] Institute of Mathematics and Informatics, Bulgarian Academy of Sciences

alexander_grigorov@fmi.uni-sofia.bg, atanas@fmi.uni-sofia.bg, pananagnostou@yahoo.gr

Abstract

Digital repositories find more and more applications in various information systems and especially in e-learning systems. Along with storing digital objects the repositories need to also to represent domain knowledge, user profiles, multilanguage classification taxonomies, etc. This could be achieved by using ontologies represented in OWL.

The Fedora digital repository is one of most powerful repositories. It supports relationships between the digital objects that are expressed in the RDF language. This allows implementation of OWL ontologies in Fedora. In this paper we describe in details our approach to represent OWL Lite ontologies in Fedora.

We have also implemented a web based ontology editor that allows easy definition, modification and storing OWL ontologies in Fedora. The editor is written in PHP, uses the Fedora services for the interface with the repository and AJAX (jQuery) for building the user interface. The paper discusses also the architecture and the implementation of the editor.

Keywords: *H.1 MODELS AND PRINCIPLES, H.4 INFORMATION SYSTEMS APPLICATIONS*

Introduction

Digital research repositories are already well established throughout many countries in the European Union. Recent surveys in the US show similar results. Digital repositories are on their way to become a permanent part of the scholarly communication and documentation research infrastructure.

Along with storing and retrieving digital objects digital repositories need to also to represent domain knowledge, user profiles, multilanguage classification taxonomies, etc. One possible solution is to ontologies represented in OWL [4, 5].

The Fedora digital repository [1, 2] is one of most powerful repositories. It supports relationships between the digital objects that are expressed in the RDF language. This allows implementation of OWL ontologies in Fedora.

Asger Blekinge-Rasmussen [3] describes a method for the implementation of an OWL LITE ontology for Fedora 3.0 data objects based on the CMA. But this approach has some limitations on the ontology and is not quite suitable for the implementation of ontology for education. So based on some of the ideas in [3] we have developed a different model that is described below.

Fedora object model

The Fedora object model [2] supports the expression of many kinds of complex objects, including documents, images, electronic books, multimedia learning objects, data sets, computer programs, and other compound information entities.

The Digital Object is the basic unit for information aggregation in Fedora. At a minimum a digital object has a persistent identifier (PID) and Dublin Core metadata [20] that provide a basic description of the digital object.

A Datastream is a component of a digital object that represents a data source. A digital object may have just the basic Dublin Core datastream, or any number of additional datastreams. Each datastream can be any mime-typed data or metadata, and can either be content managed locally in the Fedora repository or by some external data source (and referenced by a URL).

The architectural view of Fedora digital object model is shown below [2].

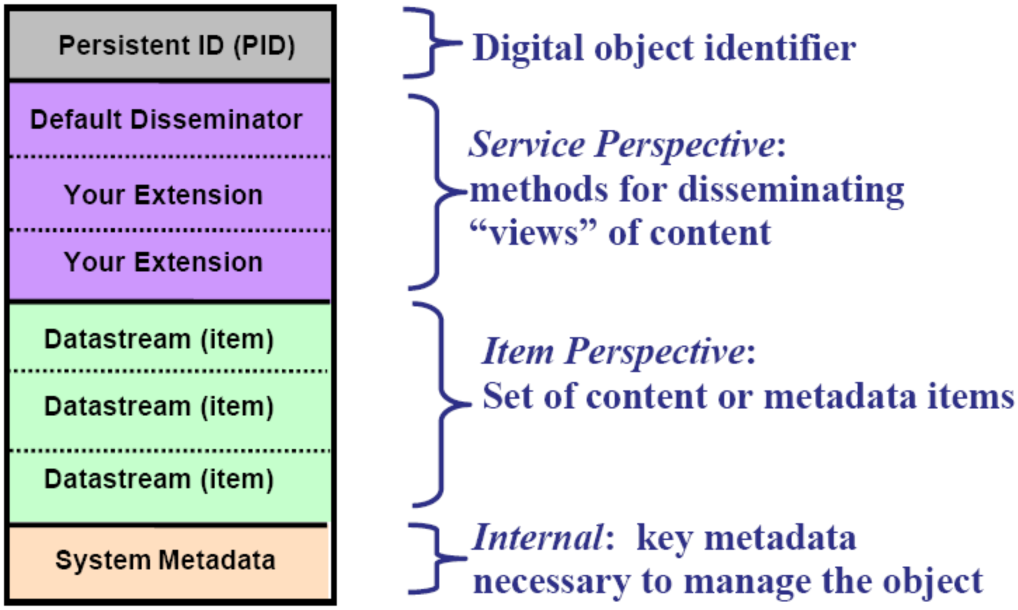

Persistent ID (PID)
Default Disseminator
Your Extension
Your Extension
Datastream (item)
Datastream (item)
Datastream (item)
System Metadata
Digital object identifier
Service Perspective: methods for disseminating "views" of content
Item Perspective: Set of content or metadata items
Internal: key metadata necessary to manage the object

Fig. 1. Fedora Digital Object Model

The Fedora object model [2] allows the definition of virtual representations of a digital object. Such a virtual representation, known as *dissemination*, is a view of an object that is produced by a service operation (a method invocation) that can take as input one or more of the datastreams of the respective digital object.

Restrictions on OWL

For the implementation of the ONTO-EDU ontology in Fedora (we will call it **ONTO-EDU** for short) we will restrict to **OWL Lite** but with some extensions and/or further restrictions.

Classes

OWL distinguishes six types of class descriptions [5]:

1. a class identifier (a URI reference)
2. an exhaustive enumeration of individuals that together form the instances of a class
3. a property restriction
4. the intersection of two or more class descriptions
5. the union of two or more class descriptions
6. the complement of a class description

For the representation of ONTO-EDU we will restrict only to simple classes (i.e. classes of type 1) and will also use some property restrictions.

The only construct that **can** be used is **rdfs:subClassOf** where the parent class should be **owl:Thing**, another class or a property restriction.

For the URIs of the classes we will use the corresponding URIs of the Fedora objects.

Properties

OWL distinguishes two types of properties [4, 5]:

- *datatype properties*, relations between instances of classes and RDF literals and XML Schema datatypes
- *object properties*, relations between instances of two classes

In the implementation of ONTO-EDU in Fedora the following constructs are allowed:

- **rdfs:subPropertyOf**
- **rdfs:domain -** the value of rdfs:domain should be a class identifier
- **rdfs:range** – the value of rdfs:range should be a class identifier for object properties, or XML Schema datatype (for example xsd:string, xsd:integer, etc.) for datatype properties.

Individuals

When defining an individual in ONTO-EDU the facts about class memberships can be expressed using **rdf:type**. The description of an individual must also contain facts about the property values for this individual.

DC annotations and multicultural support

The ONTO-EDU ontology can be annotated using Dublin Core (DC) metadata. We have chosen DC because Fedora itself has full built-in support for DC in a separate datastream and the metadata is automatically indexed on all DC fields. All OWL objects (classes, properties and individuals) in ONTO-EDU-Fedora can be annotated using DC.

In general, the following DC fields will be used in the ontology:

- **identifier** – the Fedora PID of the object
- **title** – the title/name of the object
- **description** – the description of the object (optional)

Representation of OWL classes

The structure of digital objects in Fedora for representation of OWL classes is shown on Fig. 2. All objects have content model **onto-CM:Class** which defines the datastreams in the objects.

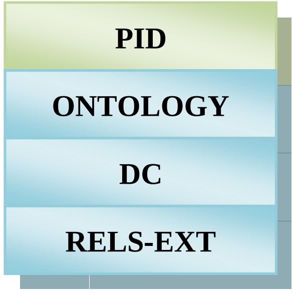


Fig. 2. Digital object structure for OWL classes

PID is the persistent identifier of the digital object in Fedora. For the OWL classes in ONTO-EDU-Fedora the PID has a prefix **onto-class**.

ONTOLOGY, **DC** and **RELS-EXT** are datastreams with internally managed XML content. The **ONTOLOGY** datastream contains the OWL description of the class. The **DC** datastream contains the Dublin Core metadata for the object. The **RELS-EXT** datastream defines the relations of the object. It should contain relations **hasModel** and **subClassOf**.

Examples:

ONTOLOGY datastream for class Context

```
<rdf:RDF xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
  <owl:Class rdf:about="info:fedora/onto-class:Context">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <dc:title xml:lang="en">Context</dc:title>
    <dc:title xml:lang="bg">Контекст</dc:title>
    <dc:description xml:lang="en">The Context includes information on
the organizations associated with the User</dc:description>
    <dc:description  xml:lang="bg">Контекстът  включва  информация  за
организациите, свързани с Потребителя.</dc:description>
    <dc:identifier>onto-class:Context</dc:identifier>
  </owl:Class>
</rdf:RDF>
```

RELS-EXT datastream for class Context

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="info:fedora/onto-class:Context">
    <subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"
       xmlns="http://www.w3.org/2000/01/rdf-schema#"/>
    <hasModel rdf:resource="info:fedora/onto-CM:Class"
       xmlns="info:fedora/fedora-system:def/model#"/>
  </rdf:Description>
</rdf:RDF>
```

Representation of OWL properties

The structure of digital objects in Fedora for representation of OWL properties is the same as for OWL classes (see Fig. 2). All objects have content model **onto-CM:Property** which defines the datastreams in the objects.

The **PID** for the OWL properties in ONTO-EDU has a prefix **onto-property**.

The **ONTOLOGY** datastream contains the OWL description of the property.

The **DC** datastream contains the Dublin Core metadata for the object.

The **RELS-EXT** datastream defines the relations of the object. It should contain relations **hasModel, definesObjectProperty** (for Object Properties), **definesDataProperty** (for Datatype properties), **domain**, **range** and **subPropertyOf** (optional).

Representation of OWL individuals

The structure of digital objects in Fedora for representation of OWL individualss is the same as for OWL classes (see Figure 2). All objects have content model **onto-CM:Object** which defines the datastreams in the objects.

The **PID** for the OWL individuals in ONTO-EDU-Fedora has a prefix **onto-object**.

The **ONTOLOGY** datastream contains the OWL description of the property.

The **DC** datastream contains the Dublin Core metadata for the object.

The **RELS-EXT** datastream defines the relations of the object. It should contain relations **hasModel, type** and the properties of the individual.

Ontology validation

The ONTO-EDU ontology is represented by a large number of digital objects in Fedora. So the consistency and validity of the ontology is very critical.

The basic assumption is that if we merge all parts of the ontology from the ONTOLOGY datastream of all digital objects (classes, properties and individuals), the resulting file should be a consistent valid ontology in OWL that can be opened and processed in Protege.

For example, if we merge the parts of the ontology from the ONTOLOGY datastream of the above examples, we will obtain the OWL ontology given in Appendix 1. This ontology can be opened and validated in Protege.

Note that the ONTOLOGY datastreams for the properties and the individuals should not contain the ontology description explicitly as described in the above sections. Since all the needed information is stored also in the DC and RELS-EXT datastreams we will develop dissemination methods for automatic generation of the content of the ONTOLOGY datastream from DC and RELS-EXT through appropriate XSLT transformations. If we restrict the ontology not to use property restrictions in the definitions of the classes, the content of the ONTOLOGY datastream for the classes can also be generated automatically.

Since the digital objects for the ONTO-EDU ontology will created, searched, edited and deleted asynchronously validation services (methods) will be developed that for each type of object (class, property or individual) will check:

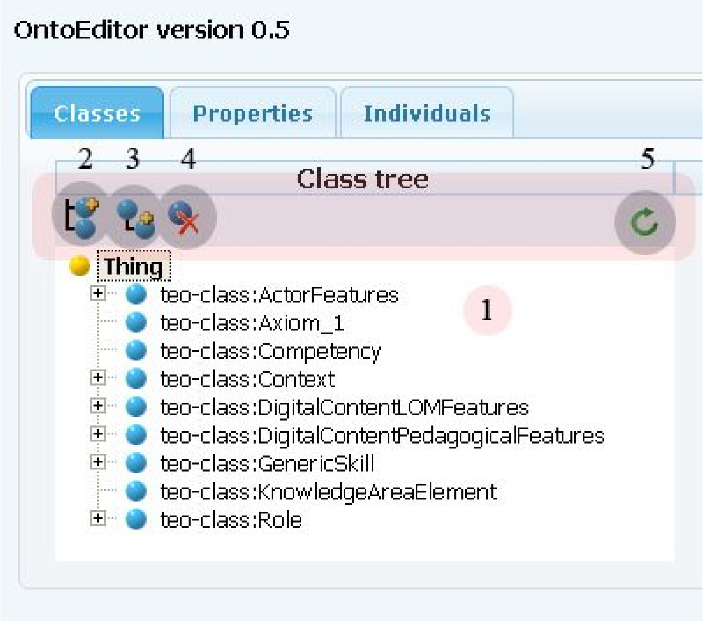
- whether a class references existing classes and properties;
- whether a property references existing classes and properties;
- whether an individual has the required properties, references existing individuals and the values of the datatype properties are valid.

Web Based Ontology Editor

We have implemented a web based ontology editor (called OntoEditor) that allows creation and modification of ontologies, represented in Fedora.

The editor is implemented in PHP and uses heavily AJAX (jQuery).

Fig. 3 shows a screenshot of the user interface of the OntoEditor.



Thing
- teo-class:ActorFeatures
- teo-class:Axiom_1
- teo-class:Competency
- teo-class:Context
- teo-class:DigitalContentLOMFeatures
- teo-class:DigitalContentPedagogicalFeatures
- teo-class:GenericSkill
- teo-class:KnowledgeAreaElement
- teo-class:Role

Fig.3. User interface of OntoEditor

We have developed the following classes in PHP that are used by the editor:

- FedoraHelper – a helper class that wraps Fedora services
- FedoraObject – for processing Fedora digital objects
- OntoClass – subclass of FedoraObject used for processing OWL classes
- OntoProperty – subclass of FedoraObject used for processing OWL properties
- OntoObject – subclass of FedoraObject used for processing OWL individuals.

Acknowledgments

The work on representing OWL ontologies in Fedora is partially supported by EC project Share.TEC - SHAring Digital REsources in the Teaching Education Community, eContentplus programme (ECP 2007 EDU 427015); http://www.sharetecproject.eu/.

The work on development of Web based Ontology editor and the use in e-learning systems is supported by Project № ВУ-ТН-204/2006 with the Ministry of Education and Science.

References

1. The Fedora Content Model Architecture http://www.fedora-commons.org/documentation/3.0/userdocs/digitalobjects/cma.html
2. The Fedora Digital Object Model http://www.fedora-commons.org/documentation/3.0/userdocs/digitalobjects/objectModel.html
3. An OWL LITE ontology for Fedora 3.0 data objects based on the CMA
http://fedora-commons.org/confluence/display/DEV/Fedora+Ontologies
4. OWL Web Ontology Language Guide
http://www.w3.org/TR/2004/REC-owl-guide-20040210/
5. OWL Web Ontology Language Reference
http://www.w3.org/TR/2004/REC-owl-ref-20040210/