**Web Services Data Access and Integration – The RDF(S) Realization (WS-DAI-RDF(S)) Ontology Specification – Profile 0, Version 0.**

Status of This Document

This document provides information regarding the specification of service-based interfaces to RDF(S) data resources. Distribution is unlimited.

Copyright Notice

Copyright © Open Grid Forum (2009). All Rights Reserved.

# Abstract

Data resources play a significant role in many applications across multiple domains. Web services provide implementation neutral facilities for describing, invoking and orchestrating collections of networked resources. The OGF (Open Grid Forum) Open Grid Services Architecture (OGSA), and its associated specifications, defines consistent interfaces through web services to components of the grid infrastructure. Both the web and grid communities stand to benefit from the provision of consistent and agreed web service interfaces for data resources and the systems that manage them.

This document presents a specification for a collection of data access interfaces for RDF(S) data resources, which extend interfaces defined in the Web Services Data Access and Integration core specification [WS-DAI]. The interfaces defined provide a set of model-based operations for accessing RDF(S) data resources at different granularities. This specification can be used together with the RDF querying access specification for RDF(S) data resources [Querying]. This specification can be applied in regular web services environments or as part of a grid fabric.

Contents

[Abstract 1](#_Toc227744532)

[1. Introduction 3](#_Toc227744533)

[1.1 Specification Scope 3](#_Toc227744534)

[1.2 Specification Organization 3](#_Toc227744535)

[1.3 Interface Composition 3](#_Toc227744536)

[2. Notational Conventions 4](#_Toc227744537)

[3. Terminology 4](#_Toc227744538)

[3.1 RDF(S) Data Resource 4](#_Toc227744539)

[4. Concepts 5](#_Toc227744540)

[4.1 Data Resources 5](#_Toc227744541)

[4.2 Interfaces 6](#_Toc227744542)

[4.3 Profiles 7](#_Toc227744543)

[4.4 Relationships with other specifications 9](#_Toc227744544)

[5. RepositoryCollection 9](#_Toc227744545)

[5.1 Static RepositoryCollection Description 9](#_Toc227744546)

[5.2 Configurable RepositoryCollection Description 10](#_Toc227744547)

[5.3 RepositoryCollectionAccess Messages 10](#_Toc227744548)

[5.4 RepositoryCollectionFactory Messages 13](#_Toc227744549)

[6. Repository 14](#_Toc227744550)

[6.1 Static Repository Description 14](#_Toc227744551)

[6.2 Configurable Repository Description 15](#_Toc227744552)

[6.3 RepositoryAccess Messages 15](#_Toc227744553)

[6.4 RepositoryFactory Messages 21](#_Toc227744554)

[7. Resource 22](#_Toc227744555)

[7.1 Static Resource Description 22](#_Toc227744556)

[7.2 Configurable Resource Description 22](#_Toc227744557)

[7.3 ResourceAccess Messages 22](#_Toc227744558)

[8. Mapping to WSDL 32](#_Toc227744559)

[9. Security Considerations 32](#_Toc227744560)

[10. Conclusion 32](#_Toc227744561)

[Editor Information 33](#_Toc227744562)

[Contributors 33](#_Toc227744563)

[Acknowledgements 33](#_Toc227744564)

[Glossary 33](#_Toc227744565)

[Intellectual Property Statement 33](#_Toc227744566)

[Disclaimer 34](#_Toc227744567)

[Full Copyright Notice 34](#_Toc227744568)

[References 34](#_Toc227744569)

[Appendix A. RepositoryCollection 36](#_Toc227744570)

[Appendix A.1 - RepositoryCollection XML Schema 36](#_Toc227744571)

[Appendix A.2 - RepositoryCollection WSDL 37](#_Toc227744572)

[Appendix B. Repository 43](#_Toc227744573)

[Appendix B.1 - Repository XML Schema 43](#_Toc227744574)

[Appendix B.2 - Repository WSDL 45](#_Toc227744575)

[Appendix C. Resource 55](#_Toc227744576)

[Appendix C.1 - Resource XML Schema 55](#_Toc227744577)

[Appendix C.2 - Resource WSDL 57](#_Toc227744578)

# Introduction

RDF(S) data access can play a central role for many types of semantic grid applications. By data access we mean the ability to retrieve, manipulate or insert data into an RDF(S) data resource.

This document presents a specification for a collection of data access interfaces for RDF(S) data resources, which extend interfaces defined in the Web Services Data Access and Integration core specification [WS-DAI]. The interfaces defined provide a set of model-based operations for accessing RDF(S) data resources at different granularities, where RDF(S) data resources are systems based on the RDF(S) data model, together with any associated management infrastructure. Examples are RDF triple storages and RDF/XML files.

The interfaces provided are organized into different profiles, so that functionalities can be delivered incrementally, and used accordingly to the end user needs. The document first provides an insight on the complete set of interfaces that provided by the Ontology specification of the RDF(S) realization. This insight constitutes the blueprints for the complete specification. The rest of the document is devoted to presenting those interface associated to the most basic profile (Profile 0).

This document should be read in conjunction with the generic Web Services Data Access and Integration specification [WS-DAI], which defines base interfaces that are extended in this document to cater for RDF(S) data resources. Also, this document should be read in conjunction with the complementary specification for RDF(S) Querying access [Querying]. These specifications have been developed for representing data resources as web services, and form part of a broader activity within the Open Grid Forum to develop the Open Grid Services Architecture (OGSA) [OGSA].

## Specification Scope

The base interfaces and properties for data access services are described in the Web Services Data Access and Integration core specification [WS-DAI]. This specification extends these interfaces to allow access to and provide descriptions of RDF(S) data resources. RDF(S) data resources are assume to contain data defined using the RDF(S) model defined in [RDF Concepts, RDF Schema], and this data is accessed using a set of ontology handling primitives based on that model.

## Specification Organization

The specification is described using the notational conventions and terminology defined in Sections 2 and 3. Sections 4 presents the concepts underlying the specification itself. Sections 5 to 7 present the different interfaces provided by the specification. The functionalities of these interfaces are mapped to WSDL in Section 8 and the appendices. Section 9 dicusses security issues. Section 10 draws the conclusions.

The WSDL and XML Schema include in the document appendices should be taken as the normative interfaces and property descriptions of this DAIS specification.

## Interface Composition

This specification does not mandate how interfaces are composed into services; the proposed interfaces may be used in isolation or in conjunction with others. Viable compositions of interfaces will, initially, follow established patterns for data access.

# Notational Conventions

The key words “MUST,” “MUST NOT,” “REQUIRED,” “SHALL,” “SHALL NOT,” “SHOULD,” “SHOULD NOT,” “RECOMMENDED,” “MAY,” and “OPTIONAL” are to be interpreted as described in RFC-2119 [RFC2119].

When describing concrete XML Schemas and XML instance fragments, this specification uses the notational convention of [WS-Security]. Specifically, each member of an element’s children or attributes property is described using an XPath-like notation (e.g., /x:MyHeader/x:SomeProperty/@value1 indicates that namespace *x* is being used, the root element *MyHeader* and a child element *SomeProperty* with an attribute *value1*). The use of {any} indicates the presence of an element wildcard (<xsd:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xsd:anyAttribute/>).

In the body of the specification, when patterns of messages are described, the layout of the XML of each message is presented, as opposed to the XML Schema. The following notation is used to indicate cardinality of XML elements in the XML fragments:

\* zero or more

+ one or more

? zero or one

Where no notation is added to an element, one instance of the element is expected.

This specification generally adopts the terminology defined in the Open Grid Services Architecture Glossary of Terms [OGSA Glossary]. This terminology is extended in Section 3.

This specification uses namespace prefixes throughout; these are listed in the table below. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

|  |  |
| --- | --- |
| **Prefix** | **Namespace** |
| wsa | http://www.w3.org/2005/08/addressing |
| wsdai | http://www.ggf.org/namespaces/2005/12/WS-DAI |
| wro | http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology |
| xs | http://www.w3.org/2001/XMLSchema |

# Terminology

Model-independent terminology, i.e. data resource, data access service, etc., is given in the WS-DAI core specification [WS-DAI]. Further specification dependent terminology is covered in the Glossary of terms of the WS-DAI-RDF(S) realization [Glossary], from which the following definition is taken.

## RDF(S) Data Resource

An RDF(S) data resource is taken to mean any system that can act as a source or sink of RDF data, together with its associated management infrastructure that exhibits capabilities that are characteristic of RDF repositories, e.g. RDF and RDF Schema entailment support.

We assume that RDF data is structured into different types of resources, which have associated values by means of properties – a particular type of resource. The types of resource, properties and values depends on the vocabulary (RDF schema) used for defining the RDF data. These resources, properties, and values, are organized into repositories. Repositories are uniquely identified by a repository name, which happens to be an IRI (see [RFC3987]). Resources are identified by its URI [RDF Concepts] together with the repository name of the repository in which they are held.

# Concepts

## Data Resources

Given the aforementioned notion of RDF(S) data resource, we can further differentiate them in two classes:

1. **Placeholders of actual built-in RDF(S) classes** (*Resource*, *Class*, *Property*, *Statement*, *Container* and *List*): these data resources provide ad-hoc views to a RDF(S) data resource, focusing in concrete parts of the underlying data and providing specialized data access methods according to the concrete nature of the element of the RDF(S) data model through which we are looking at the data resource.

These set of data resources are organized hierarchically: the *Resource* data resource defines the minimum common data that defines this kind of data resources, and the rest of the data resources specialize the view provided by this data resource.

This set of data resources can be also divided in two subsets: one which contains data resources placholders of elements that define the schema of the ontology (*Class* and *Property*), and rest, which are more focused in the data defined according to the schema. The *Resource* data resource lies in the middle of them, it can also be used for modifying the schema, but also for modifying the data defined according to the schema.

1. **Convenience abstractions** (RepositoryCollection and Repository): these data resources allow the provision of additional methods that enable an easy management of the previous data resources, with different granularity levels.

This way, a *repository* is an aggregation of resources, where the key issue is that the *resources* **belong** to the *repository* (this could be understood as if the *resource* data resource was a derived data resource from the *repository*).

Similarly, a *repository collection* data resource is an aggregation of *repositories*, the repositories managed by the implementation. Nevertheless, and opposite to the aggregation of *resources* in *repositories*, the repositories are not owned by the *repository collection*, just managed by it, so that these repositories can also be managed in other way by other realizations.



Fig. WS-DAI-RDF(S) Ontology Data resource model

The diagram depicted in Fig. 1 shows which are the data resources defined in the realization and the relationships existing between them using a UML notation.

## Interfaces

The word interface refers to the collections of messages and XML structures that describe the ways in which a consumer can validly interact, through this and the WS-DAI specifications, with a data access service. It is not intended to refer specifically to the proposed use of the word interface found in the current working draft of the WSDL 2.0 specification, although this may be an appropriate mapping in the future.

In order to interact with the data resources described in the previous section, several interfaces are specified in the WS-DAI-RDF(S) Ontology specification. Fig. 2 depicts the interfaces defined and also presents a possible composition of them into data access services. The dashed blue lines represent the intended navigability from factory interfaces to access interfaces.



Fig. Data access services and interfaces

Direct access to a data access service allows the results of a request to be delivered to the consumer directly in the response message. To cater for this mode of operation the following interfaces are defined for accessing RDF(S) data resources:

* RepositoryCollectionAccess: provides access to repositories of a collection.
* RepositoryAccess: provides access to the inside of repositories, offering functionalities for managing the repository at RDF(S) resource level.
* ResourceAccess: provides access to a particular RDF(S) resource, centered in those aspects common to every resource: property value management, resource description, etc.
* ClassAccess: provides access to particular RDF(S) resources that are RDF(S) class, focusing in the data that is specific to RDF(S) classes: class hierarchy traversal, instance retrieval, etc.
* PropertyAccess: provides access to particular RDF(S) resources that are RDF(S) properties, focusing in the data that is specific to RDF(S) properties: range and domain management, property hierarchy traversal, etc.
* StatementAccess: provides access to particular RDF(S) resources that are RDF(S) statements –reified triples, not the triples themselves – focusing the management of the components that set up the reification.
* ListAccess: provides access to particular RDF(S) resources that are RDF collections (List), focusing in the management of the members of the collection, as well as, the structure of the collection.
* ListIteratorAccess: provides access to RDF collections following the iterator pattern [Gamma et al, 1994], allowing an easy retrieval of the members of the collection without requiring the identification of the position in it.
* ContainerAccess: provides access to particular RDF(S) resources that are RDF(S) containers, focusing in the management of the members of the container, as well as, the structure of the container, regardless the its specific type[[1]](#footnote-2).
* ContainerIteratorAccess: provides access to RDF(S) containers following the iterator pattern [Gamma et al, 1994], allowing an easy retrieval of the members of the container without requiring the identification of the position in it.
* AltAccess: provides access to particular RDF(S) containers that are of the particular *alt* type.

Indirect access is supported through the use of the factory pattern. This allows data, usually a particular view of the whole data set, to be accessed by way of a new service-managed data resource, and thus the data is not returned directly to the consumer. To cater for this mode of operation the following interfaces are defined:

* RepositoryCollectionFactory: provides access to repositories of a collection.
* RepositoryFactory: provides access to the inside of repositories.
* ListFactory: provides access to the contents of an RDF collection.
* ContainerFactory: provides access to the contents of a container.

The usage of the factory patterns provides a basic navigation mechanism that lets the user browse RDF(S) data resources with different granularities and exploiting the particular semantics of the RDF(S) data represented by the RDF(S) data resource.

In addition to these interfaces, multiple description interfaces are provided (RepositoryCollectionDescription, RepositoryDescription, ResourceDescription, ClassDescription, PropertyDescription, StatementDescription, ListDescription, and ContainerDescription), which extend the properties enumerated in the WS-DAI core specification to provide information about the relationships of the RDF(S) data services and resources with the RDF(S) data to which they provide access.

## Profiles

The WS-DAI-RDF(S) Ontology specification is aimed at providing a means for managing RDF(S) data resources in an integral fashion, offering mechanisms for browsing, adding, updating and deleting contents. The specification defines these mechanisms following the RDF(S) model and semantics, providing to the user with different granularity levels in order to use available data resources with the required detail. To achieve all of this, the specification provides multiple different data resources and interfaces. As a result, the full specification is bigger that that of other realizations, such as the relational one [WS-DAIR].

From a user/developer point of view, the usefulness of the specification will depend on his requirements, and especially on the necessities he has when dealing with the RDF(S) data sources, that is, what he needs to do, and how he expects to do it. Thus, depending on his needs, he shall only use the subset of the specification that completely fulfils them.

Consequently, in order to facilitate the adoption and implementation of the specification from the community, it is necessary to allow using what is needed, without enforcing the adoption of the elements not needed.

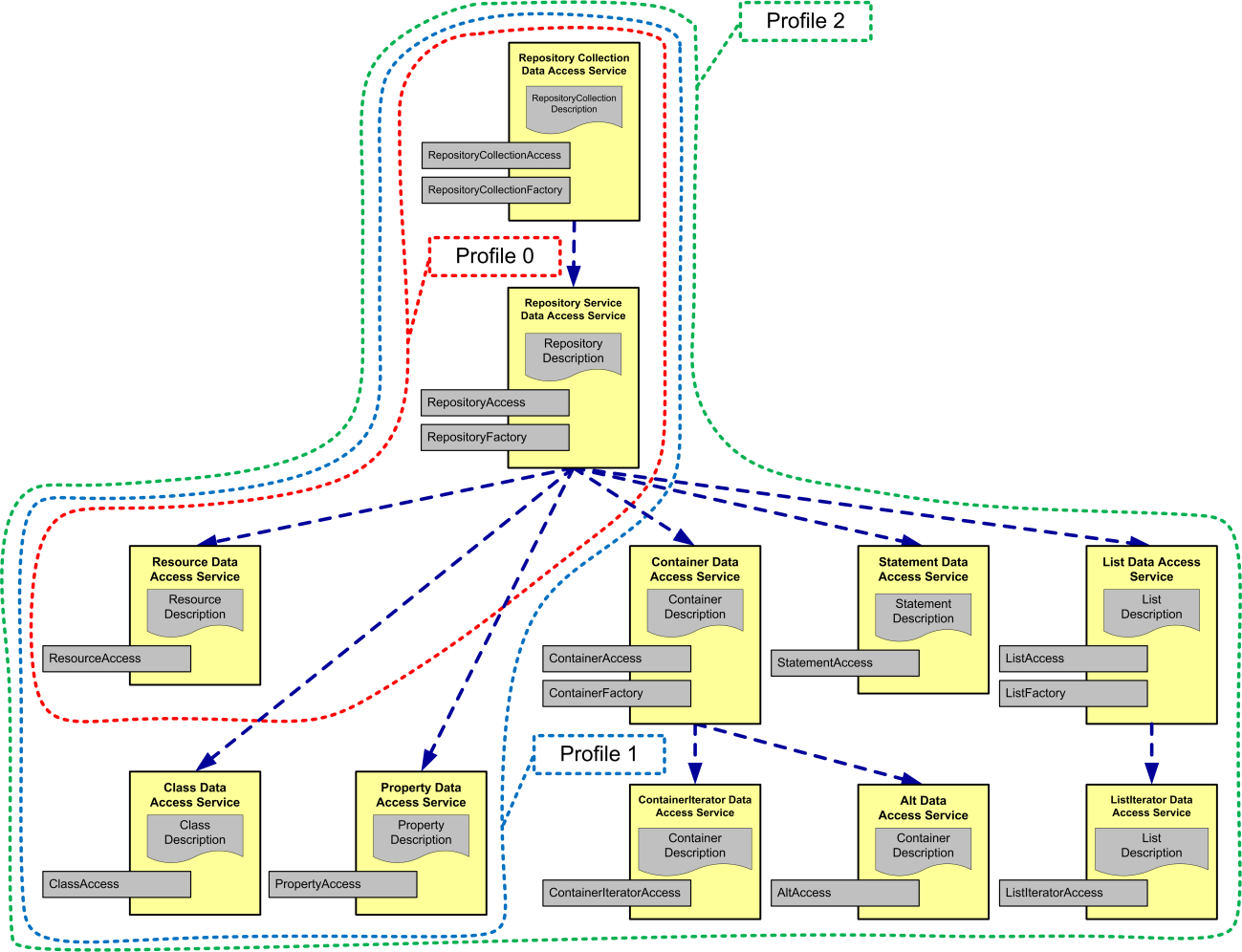


Fig. WS-DAI-RDF(S) Ontology profiles

To achieve this we divide the specification in three different *profiles* (see Fig. 3), each one including an increasing number of functionalities that shall enable the user to deal with RDF(S) data resources with finer grain of detail, while ensuring interoperability among implementations:

* **Profile 0: *Basic RDF support*.** This profile includes the minimum set of functionalities needed for dealing with RDF data. The functionalities of these profile are those defined in the following interfaces:
  + RepositoryCollectionAccess
  + RepositoryCollectionFactory
  + RepositoryAccess
  + RepositoryFactory, limited to Resource EPR retrieval.
  + ResourceAccess
* **Profile 1: *RDF Schema support*.** This profile includes the functionalities described in the Profile 0, and extends them with those required for dealing with RDF vocabularies. The new functionalities are defined in the following interfaces:
  + RepositoryFactory, augmented to retrieve Class and Property EPRs.
  + ClassAccess
  + PropertyAccess
* **Profile 2: *Full RDF(S) support*.** This profile includes the functionalities described in Profile 1, and extends them with those required for dealing with the rest of the built-in RDF vocabulary (containers, RDF collections and reifications). These functionalities are defined in the following interfaces:
  + RepositoryFactory, augmented to support the rest of the data resource EPRs.
  + StatementAccess
  + ListAccess
  + ListFactory
  + ListIteratorAccess
  + ContainerAccess
  + ContainerFactory
  + ContainerIteratorAccess
  + ContainerFactory

In this document we will cover the interfaces and capabilities that belong to the first profile (Profile 0). The other profiles will be covered in further documents, using as a basis the model and approach here defined, upgrading the interfaces and functionalities defined hereafter as required.

## Relationships with other specifications

The WS-DAI-RDF(S) Ontology specification is focused on providing access to RDF(S) data resources, by means of a set of ontology handling primitives based on the RDF(S) data model and semantics, which are defined in the W3C specification of the Resource Description Framework. This specification is made up of the following documents:

* RDF Primer [RDF Primer]: this non-normative document is intented to provide an introduction to RDF and how could it be used.
* Resource Description Framework (RDF): Concepts and Abstract Syntax [RDF Concepts]: this document defines the abstract syntax in which RDF is based, which is necessary to understand both its concrete syntax and semantics.
* RDF/XML Syntax Specification (Revised) [RDF Syntax]: this document presents the normative format for serializing RDF using XML.
* RDF Semantics [RDF Semantics]: this document defines the precise semantics (and inference rules) of both RDF and RDF Schema.
* RDF Vocabulary Description Language 1.0: RDF Schema [RDF Schema]: this document presents how to use RDF for describing new RDF vocabularies, defining a vocabulary for this purpose, as well as the other built-in RDF vocabulary presented in [RDF Concepts].
* RDF Test Cases [RDF Tests]: this document present the set of test cases.

The WS-DAI-RDF(S) realization could be extended to encompass any new or emerging RDF(S) standards by employing the patterns defined in this document.

# RepositoryCollection

## Static RepositoryCollection Description

The elements defined in this section live under a RepositoryCollectionPropertyDocument that extends the PropertyDocument defined in the WS-DAI core specification.

### NumberOfRepositories

<xs:element name="NumberOfRepositories" type="xs:unsignedLong"/>

|  |  |
| --- | --- |
| /wro:NumberOfRepositories | |
|  | The total number of repositories of the collection. |

### RepositoryCollection

<xs:element name="RepositoryCollection">

<xs:complexType>

<xs:sequence>

<xs:element name="Repository" minOccurs="0" maxOccurs="unbounded" type="wro:RepositoryDescriptorType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

|  |  |
| --- | --- |
| /wro:RepositoryCollection | |
|  | Describes the set of available repositories. |

### RepositoryCollectionPropertyDocument

<xs:complexType name="RepositoryCollectionPropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:NumberOfRepositories"/>

<xs:element ref="wro:RepositoryCollection"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="RepositoryCollectionPropertyDocument" type="wro:RepositoryCollectionPropertyDocumentType"/>

|  |  |
| --- | --- |
| /wro:RepositoryCollectionPropertyDocument | |
|  | A structure that describes a RepositoryCollection data resource, combining the properties defined in the WS-DAI core specification with those defined earlier in this section. |

## Configurable RepositoryCollection Description

No extra configurable properties are defined for RepositoryCollection.

## RepositoryCollectionAccess Messages

This interface provides the functionalities for managing the repositories that are under control of the RDF(S) storage manager.

### RepositoryCollectionAccess::GetRepositoryCollectionPropertyDocument

Allows a copy of the RepositoryCollectionPropertyDocument document to be retrieved.

Input

* GetRepositoryCollectionPropertyDocumentRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.

Output

* GetRepositoryCollectionPropertyDocumentResponse
  + GetRepositoryCollectionPropertyDocumentDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetRepositoryCollectionPropertyDocumentDatasetData – The data returned in response to the request.
      * RepositoryCollectionPropertyDocument\* – The properties described in the data description section.
    - NumberOfRepositories – The number of available repositories.

Faults

* InvalidResourceNameFault - The supplied repository resource name is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* NotAuthorizedFault - The consumer is not authorized to perform this operation at this time.
* ServiceBusyFault - The service is already processing a request and the ConcurrentAccess is false.

### RepositoryCollectionAccess::GetRepositories

Retrieve repositories from the collection, serialized in RDF/XML. An attempt SHOULD be made to retrieve all the specified repositories even if some of them do not exist in the collection.

Input

* GetRepositoriesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + RepositoryName+ – The names of the repositories to be retrieved.

Output

* GetRepositoriesResponse
  + GetRepositoriesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetRepositoriesDatasetData – The data returned in response to the request.
      * RepositoryResponse+ – For each repository:
        + RepositoryName – The name of the repository.
        + Result – The result of the repository retrieval operation for the specific repository. Possible values for this are: success, repository not retrieved as it does not exist; and repository not retrieved as the consumer is not authorized.
        + Data? – The content of the repository retrieved. This element MUST appear when the retrieval operation succeeds.
    - NumberOfRepositoriesRetrieved – The number of repositories retrieved from the collection.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryCollectionAccess::AddRepositories

Create new repositories in the collection. Any initial data MUST be passed serialized in RDF/XML. An attempt SHOULD be made to add all the specified repositories even if some additions fail. The consumer MAY specify whether the service MUST overwrite repositories or not.

Input

* AddRepositoriesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + Repository+ – For each repository:
    - RepositoryName – The desired name for the repository.
    - Data – The contents of the repository serialized in RDF/XML.
  + OverwriteRepositories? – An OPTIONAL parameter that specifies whether or not the service can overwrite an existing repository with new contents passed in the request. By default, the service SHALL not overwrite repositories. In case of repository overwrite, existing derived data resources MUST be terminated.

Output

* AddRepositoriesResponse
  + AddRepositoriesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - AddRepositoriesDatasetData – The data returned in response to the request.
      * RepositoryResponse+ – For each repository that was to be added to the collection:
        + RepositoryName – The name of the repository.
        + Result – The result of the repository addition operation for the specific repository. Possible values for this are: success; existing repository updated; repository not added as a repository with the same name already exists; and repository not added as the consumer is not authorized.
    - OverwriteRepositories – Whether or not repository overwrite was requested.
    - NumberOfRepositoriesAdded – The number of repositories added to the collection. This figure includes repository overwrites, if any takes place.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryCollectionAccess::RemoveRepositories

Remove repositories from the collection. An attempt SHOULD be made to remove all of the specified repositories even if some removals fail. Derived repository data resources SHALL be removed upon repository removal from the collection.

Input

* RemoveRepositoriesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + RepositoryName+ – The names of the repositories that are to be removed from the collection.

Output

* RemoveRepositoriesResponse
  + RemoveRepositoriesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - RemoveRepositoriesDatasetData – The data returned in response to the request.
      * RepositoryResponse+ – For each repository that was to be removed from the collection:
        + RepositoryName – The name of the repository.
        + Result – The result of the repository removal operation for the specific repository. Possible values for this are: success; repository not removed as it does not exist; and repository not removed as the consumer is not authorized.
    - NumberOfRepositoriesRemoved – The number of repositories removed from the collection.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

## RepositoryCollectionFactory Messages

This factory interface defines the mechanisms needed for providing specialized access to repositories through data services, which implement the RepositoryAccess and RepositoryFactory interfaces.

### RepositoryCollectionFactory::GetRepositoryFactory

Returns the endpoint reference of a repository data resource that represents an existing repository in the collection. This factory operation can be used if access to a repository through a different port type, such as RepositoryAccess, is required.

Input

* GetRepositoryFactoryRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + PortTypeQName? – An OPTIONAL parameter that specifies the QName of the portType through which the resulting data should be accesed. The QName value here MUST correspond to one that is specified in the ConfigurationMap property.
  + ConfigurationDocument? – An OPTIONAL parameter that includes a document that specifies the properties of the data resource that is to be used to access the resulting data. The properties contained in this document are those defined in the WS-DAI specification.
  + PreferredTargetService? – An OPTIONAL parameter that identifies a EPR of the preferred service that is to act as the host for the new data resource.
  + RepositoryName – The name of the repository which will be accesible through a new repository data resource.

Output

* GetRepositoryFactoryResponse
  + DataResourceAddress\* – A list of data resource addresses.

Faults

* InvalidResourceNameFault - The supplied repository resource name is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* InvalidPortTypeQNameFault - The PortTypeQName is not in the collection defined by the ConfigurationMap property.
* InvalidConfigurationDocumentFault - The ConfigurationDocument specified is not valid according to the ConfigurationDocumentQName when the ConfigurationMap is indexed by the specified PortTypeQName.
* NotAuthorizedFault - The consumer is not authorized to perform this operation at this time.
* ServiceBusyFault - The service is already processing a request and the ConcurrentAccess is false.
* RepositoryDoesNotExistFault - The specified repository does not exist in the collection.

# Repository

## Static Repository Description

The elements defined in this section live under a RepositoryPropertyDocument that extends the PropertyDocument defined in the WS-DAI core specification.

### RepositoryName

<xs:element name="RepositoryName" type="xs:anyURI"/>

|  |  |
| --- | --- |
| /wro:RepositoryName | |
|  | The IRI that uniquely identifies the repository within all the repositories in the collection to which this belongs to. |

### RepositoryPropertyDocument

<xs:complexType name="RepositoryPropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:RepositoryName"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="RepositoryPropertyDocument" type="wro:RepositoryPropertyDocumentType"/>

|  |  |
| --- | --- |
| /wro:RepositoryPropertyDocument | |
|  | A structure that describes a Repository data resource, combining the properties defined in the WS-DAI core specification with those defined earlier in this section. |

## Configurable Repository Description

No extra configurable properties are defined for Repository.

## RepositoryAccess Messages

The interfaces provides the basic messages needed for managing the contents of an specific repository at resource level. The messages defined provide the mechanisms needed for: retrieving resources, creating new resources, updating existing resources, and removing existing resources.

### RepositoryAccess::GetRepositoryPropertyDocument

Allows a copy of the RepositoryPropertyDocument document to be retrieved.

Input

* GetRepositoryPropertyDocumentRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.

Output

* GetRepositoryPropertyDocumentResponse
  + GetRepositoryPropertyDocumentDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetRepositoryPropertyDocumentDatasetData – The data returned in response to the request.
      * RepositoryPropertyDocument\* – The properties described in the data description section.
    - NumberOfRepositories – The number of available repositories.

Faults

* InvalidResourceNameFault - The supplied repository resource name is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* NotAuthorizedFault - The consumer is not authorized to perform this operation at this time.
* ServiceBusyFault - The service is already processing a request and the ConcurrentAccess is false.

### RepositoryAccess::GetResources

Retrieve resources from the repository. Property-based filter expressions can be used to determine the specific sets of resources that are to be retrieved. The consumer can specify whether or not associated property values will be retrieved together with the resources URI. An attempt SHOULD be made to retrieve all the specified sets of resources even if some retrievals fail.

Input

* GetResourcesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + ResourceSet+ – For each set of resources that the consumer desires to retrieve:
    - ResourceURI\* – An OPTIONAL set of URIs of candidate resources which SHALL be retrieved if they satisfy the property filters specified below. If no resource is specified, all the matching resources SHALL be retrieved.
    - PropertyFilter\* – An OPTIONAL set of property-based filters. If no filter is specified, all the existing resources SHALL be retrieved. For each filter:
      * PropertyURI – The property that must be defined for the retrieved resources.
      * PropertyValueFilter\* – An OPTIONAL set of property values that must be associated to the retrieved resources through the specified property.
      * ValueFilterSemantics? – An OPTIONAL parameter that determines the way in which the property values shall be interpreted. Possible values for this are: AND, if all the specified values must appear; OR, if one or more of the specified values must appear; XOR, if only one of them must appear; NONE, if none of the specified values must appear. This parameter should only be used when property values are specified. If property values are specified, then the default value for this parameter will be AND.
    - PropertyFilterSemantics? – An OPTIONAL parameter that determines the way in which the property value filters SHALL be interpreted. Possible values for this are: AND, if all the specified filters MUST be satisfied; OR, if one or more of the specified filters MUST be satisfied; XOR, if only one of the filters MUST be satisfied; NONE, if none of the specified filters MUST be satisfied. This OPTIONAL parameter SHOULD only be used when property filters are specified. The default value for this parameter will be AND.
  + RetrieveResourceProperties? – An OPTIONAL parameter that indicates whether or not associated propety values MUST be retrieved together with the resource URI. By default, this parameter is FALSE.

Output

* GetResourcesResponse
  + GetResourcesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetResourcesDatasetData – The data returned in response to the request.
      * ResourceSetResult+ – For each one of the resource sets requested:
        + Result – The result of the resource retrieval operation for the specified set of resources. Possible values for this are: success; no resources found; and no resources retrieved as the consumer is not authorized.
        + Resource\* – For each one of the resources that satisfied the request filters:

ResourceURI – The resource URI.

DefinedProperty\* – The properties defined for the resource. This elements SHALL only appear if resource property retrieval was requested. For each of the the properties:

PropertyURI – The property URI.

PropertyValue+ – The values defined for the property and resource.

* + - RetrieveResourceProperties – Whether or not the consumer requested retrieving the properties defined for the resources.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryAccess::GetResourceProperties

Retrieve property values for the specified resources. The consumer can determine the set of properties to be retrieved for each one of the desired resources. An attempt SHOULD be made to retrieve all the specified properties of the choosen resources even if some retrievals fail.

Input

* GetResourcePropertiesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + ResourceSet+ – For each set of resources from which the consumer desires to retrieve property values:
    - ResourceURI+ – The URIs of the resources for which the following property values will be retrieved.
    - PropertyURI\* – An OPTIONAL set of URIs of the properties whose values for the specified resources will be retrieved. If no property is specified, all the property values of the specified resources SHALL be retrieved.

Output

* GetResourcePropertiesResponse
  + GetResourcePropertiesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetResourcePropertiesDatasetData – The data returned in response to the request.
      * ResourceSetResult+ – For each one of the resources requested:
        + Resource+ – For each one of the specified resources:

ResourceURI – The resource URI.

Result – The result of the resource properties retrieval operation for the specified resource. Possible values for this are: success; no property values found; and no property values retrieved as the consumer is not authorized.

DefinedProperty\* – The properties defined for the resource. For each one of them:

PropertyURI – The property URI.

PropertyValue+ – The values defined for the property and resource.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryAccess::AddResources

Add resources to the repository. If the specified resource already exists in the repository, new property values will be added to it. An attempt SHOULD be made to add all the specified resources and property values even if some additions fail.

Input

* AddResourcesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + Resource+ – The resources that will be added to the repository. For each resource:
    - ResourceURI? – An OPTIONAL URI for the resource that is to be added to the repository. If no URI is specified a new blank node will be created.
    - Property+ – The properties defined for the resource. For each one of them:
      * PropertyURI – The property URI.
      * PropertyValue+ – The values defined for the property and resource.

Output

* AddResourcesResponse
  + AddResourcesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - AddResourcesDatasetData – The data returned in response to the request.
      * ResourceResult+ – For each one of the resources that were to be added:
        + ResourceURI – The URI of the resource. This field will contain the same URI specified on the request, or the URI of the newly created blank node in case no resource URI was specified in the request.
        + Result – The result of the resource addition operation for the specified resource. Possible values for this are: success; properties added to existing resource; and resource not added/updated as the consumer is not authorized.
    - NumberOfResourcesAdded – The number of resources added to the repository.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryAccess::RemoveResources

Remove resources from the repository. Property-based filter expressions can be used to determine the specific set of resources that are to be removed. An attempt SHOULD be made to remove all the specified sets of resources even if some removals fail. Derived resource data resources SHALL be removed upon resource removal from repository.

Input

* RemoveResourcesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + ResourceSet+ – For each set of resources the consumer desires to remove:
    - ResourceURI\* – An OPTIONAL set of URIs of candidate resources which will be removed if they satisfy the property filters. If no resource is specified, all the matching resources SHALL be removed.
    - PropertyFilter\* – An OPTIONAL set of property-based filters. If no filter is specified, all the existing resources SHALL be removed. For each property filter:
      * PropertyURI – The property that must be defined for the resources to be removed.
      * PropertyValueFilter\* – An OPTIONAL set of property values that must be associated to the resources to be removed, through the specified property.
      * ValueFilterSemantics? – An OPTIONAL parameter that determines the way in which the property values SHALL be interpreted. Possible values for this are: AND, if all the specified values MUST appear; OR, if one or more of the specified values MUST appear; XOR, if only one of them MUST appear; NONE, if none of the specified values MUST appear. This parameter SHOULD only be used when property values are specified. The default value for this parameter will be AND.
    - PropertyFilterSemantics? – An OPTIONAL parameter that determines the way in which the property filters SHALL be interpreted. Possible values for this are: AND, if all the specified filters MUST be satisfied; OR, if one or more of the specified filters MUST be satisfied; XOR, if only one of the filters MUST be satisfied; NONE, if none of the specified filters MUST be satisfied. This parameter SHOULD only be used when property filters are specified. The default value for this parameter will be AND.

Output

* RemoveResourcesResponse
  + RemoveResourcesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - RemoveResourcesDatasetData – The data returned in response to the request.
      * ResourceSetResult+ – For each one of the resource sets that were to be removed:
        + Resource\* – For each one of the resources that were to be removed:

ResourceURI – The URI of the resource.

Result – The result of the resource removal operation for the resource. Possible values for this are: success; and resource not removed as the consumer is not authorized.

* + - NumberOfRemovedResources – The number of resources removed from the repository.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### RepositoryAccess::RemoveResourceProperties

Remove property values for the specified resources. The consumer can determine the set of property values to be removed for each one of the desired resources. An sattempt SHOULD be made to remove all the specified property values of the choosen resources even if some removals fail.

Input

* RemoveResourcePropertiesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + ResourceSet+ – For each set of resources from which the consumer desires to remove property values:
    - ResourceURI+ – The URIs of the resources for which the property values will be retrieved.
    - Property\* – An OPTIONAL set of properties whose values for the specified resources will be removed. If no property is specified, all the property values of the specified resources SHALL be removed. For each property:
      * PropertyURI – The URI of the property.
      * PropertyValue\* – An OPTIONAL set of values of the property for the resource that will be removed. If no values are specified, all of them SHALL be removed.

Output

* RemoveResourcePropertiesResponse
  + RemoveResourcePropertiesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - RemoveResourcePropertiesDatasetData – The data returned in response to the request.
      * ResourceSetResult+ – For each one of the resource sets specified:
        + Resource+ – For each one of the resources whose property values were to be removed:

ResourceURI – The resource URI.

Result – The result of the resource properties removal operation for the specified resource. Possible values for this are: success; no matching property values found; all property values removed; and no property values removed as the consumer is not authorized.

Faults

* InvalidResourceNameFault - The supplied repository resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

## RepositoryFactory Messages

This factory interface defines the mechanisms needed for providing specialized access to the resources stored in the repository through data services, which implement specific data access interfaces depending on the nature of the particular resource, i.e. ResourceAccess, ClassAccess, PropertyAccess, etc.

### RepositoryFactory::GetResourceFactory

Returns the endpoint reference of a data resource that represents an existing resource in the repository. This factory operation can be used if access to a repository through a different port type, such as ResourceAccess, is required. This operation MUST be supported by every implementation, regardless of the profile they support.

Input

* GetResourceFactoryRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + PortTypeQName? – An OPTIONAL parameter that specifies the QName of the portType through which the resulting data should be accesed. The QName value here MUST correspond to one that is specified in the ConfigurationMap property.
  + ConfigurationDocument? – An OPTIONAL parameter that includes a document that specifies the properties of the data resource that is to be used to access the resulting data. The properties contained in this document are those defined in the WS-DAI specification.
  + PreferredTargetService? – An OPTIONAL parameter that identifies a EPR of the preferred service that is to act as the host for the new data resource.
  + ResourceURI – The URI of the resource for whom a new resource data access service will be created.

Output

* GetResourceFactoryResponse
  + DataResourceAddress\* – A list of data resource addresses.

Faults

* InvalidResourceNameFault - The supplied repository resource name is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* InvalidPortTypeQNameFault - The PortTypeQName is not in the collection defined by the ConfigurationMap property.
* InvalidConfigurationDocumentFault - The ConfigurationDocument specified is not valid according to the ConfigurationDocumentQName when the ConfigurationMap is indexed by the specified PortTypeQName.
* NotAuthorizedFault - The consumer is not authorized to perform this operation at this time.
* ServiceBusyFault - The service is already processing a request and the ConcurrentAccess is false.
* ResourceDoesNotExistFault - The specified resource is not defined in the repository.

# Resource

## Static Resource Description

The elements defined in this section live under a ResourcePropertyDocument that extends the PropertyDocument defined in the WS-DAI core specification.

### RepositoryName

<xs:element name="RepositoryName" type="xs:anyURI"/>

|  |  |
| --- | --- |
| /wro:RepositoryName | |
|  | The name of the repository in which the RDF(S) resource represented by this data resource is originally defined. |

### ResourceURI

<xs:element name="ResourceURI" type="xs:anyURI"/>

|  |  |
| --- | --- |
| /wro:ResourceURI | |
|  | The URI of the RDF(S) resource represented by the data resource. |

### ResourcePropertyDocument

<xs:complexType name="ResourcePropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:RepositoryName"/>

<xs:element ref="wro:ResourceURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="ResourcePropertyDocument" type="wro:ResourcePropertyDocumentType"/>

|  |  |
| --- | --- |
| /wro:ResourcePropertyDocument | |
|  | A structure that describes a Resource data resource, combining the properties defined in the WS-DAI core specification with those defined earlier in this section. |

## Configurable Resource Description

No extra configurable properties are defined for Resource.

## ResourceAccess Messages

This interface provides access methods centered in a particular RDF resource of a repository. There are two groups of methods. On the one hand there is a set of methods for managing the values associated to the main built-in properties defined for the Resource class in the RDF model (rdf:type, rdfs:label, rdfs:comment, rdfs:seeAlso, and rdfs:isDefinedBy); on the other hand, a set of general purpose methods is defined. These methods provide CRUD/BREAD functionalities for dealing with the RDF model, using the Resource class (specifically the concrete resource represented by the service) as the center of the model.

### ResourceAccess::GetResourcePropertyDocument

Allows a copy of the ResourcePropertyDocument document to be retrieved.

Input

* GetResourcePropertyDocumentRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.

Output

* GetResourcePropertyDocumentResponse
  + GetResourcePropertyDocumentDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetResourcePropertyDocumentDatasetData – The data returned in response to the request.
      * ResourcePropertyDocument\* – The properties described in the data description section.
    - NumberOfRepositories – The number of available repositories.

Faults

* InvalidResourceNameFault - The supplied repository resource name is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* NotAuthorizedFault - The consumer is not authorized to perform this operation at this time.
* ServiceBusyFault - The service is already processing a request and the ConcurrentAccess is false.

### ResourceAccess::GetResourceTypes

Retrieve the URIs of the RDF(S) classes that are parents of the resource, traversing the whole class hierarchy if required.

Input

* GetResourceTypesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + InferParents? – This OPTIONAL parameter determines whether or not the service MUST infer the parent classes.

Output

* GetResourceTypesResponse
  + GetResourceTypesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetResourceTypesDatasetData – The data returned in response to the request.
      * ParentURI\* – The URIs of the parent classes of the resource.
    - InferParents – Whether or not the service took into account inferred parent classes.
    - NumberOfParents – The number of parents of the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::GetResourceDescription

Retrieve the basic description of the resource. The description is made up of the values of the following properties: rdfs:label, rdfs:comment, rdfs:seeAlso, and rdfs:isDefinedBy. An attempt SHOULD be made to retrieve all the specified property values even if some retrievals fail.

Input

* GetResourceDescriptionRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.

Output

* GetResourceDescriptionResponse
  + GetResourceDescriptionDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetResourceDescriptionDatasetData – The data returned in response to the request.
      * Description – The basic description of the resource.
        + Label\* – The labels of the resource.
        + Comment\* – The comments of the resource.
        + SeeAlso\* – Further resources that might provide additional information about the resource.
        + IsDefinedBy\* – Further resources that identify RDF vocabularies in which the resource is described.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::GetPropertyValues

Retrieve the property values of the resource. If no properties are specified, the values of all the defined properties SHALL be retrieved. An attempt SHOULD be made to retrieve all the specified property values even if some retrievals fail.

Input

* GetPropertyValuesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + PropertyURI\* – An OPTIONAL set URIs of the properties that are to be retrieved.

Output

* GetPropertyValuesResponse
  + GetPropertyValuesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetPropertyValuesDatasetData – The data returned in response to the request.
      * PropertyResult\* – For each one of the properties whose values were to be retrieved:
        + PropertyURI – The URI of the property.
        + PropertyValue+ – The concrete values defined for the property.
    - NumberOfPropertyValuesRetrieved – The number of property values that have been retrieved.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::GetDefinedProperties

Retrieve the URIs of the properties which are defined for the resource, that is, those properties which are the predicate of a statement which has the resource as subject.

Input

* GetDefinedPropertiesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.

Output

* GetDefinedPropertiesResponse
  + GetDefinedPropertiesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetDefinedPropertiesDatasetData – The data returned in response to the request.
      * PropertyURI\* – The URIs of the specific properties defined for the resource.
    - NumberOfDefinedProperties – The number of properties that are defined for the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::GetDefinableProperties

Retrieve the URIs of the properties that might be defined for this resource, that is, those properties which have as domain a class that is a parent of the resource. The resolution mechanism to be used for retrieving the data can be specified by identifying the reasoning capabilities that SHALL be applied by using the request's parameters provided. Both parameters rely on the predefined RDF Schema entailment rules [RDF Semantics], which allow reasoning over taxonomies of both classes and properties, thus widening or narrowing the expected result set.

Input

* GetDefinablePropertiesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + InferInheritedProperties? – This OPTIONAL parameter determines whether or not the service MUST infer the properties that are inherited from parent classes of this resource. By default, this parameter is set to FALSE.
  + InferSubProperties? – This OPTIONAL parameter determines whether or not the service MUST infer the sub properties of the definable properties. By default, this parameter is set to FALSE.

Output

* GetDefinablePropertiesResponse
  + GetDefinablePropertiesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - GetDefinablePropertiesDatasetData – The data returned in response to the request.
      * PropertyURI\* – The URIs of the specific properties that are definable for the resource.
    - InferInheritedProperties – Whether or not inherited properties were taken into consideration.
    - InferSubProperties – Whether or not sub properties were taken into consideration.
    - NumberOfDefinableProperties – The number of properties that might be defined for the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::AddResourceTypes

Add type values to the resource. The RDF(S) classes that represent the types to be added are identified by their URI. An attempt SHOULD be made to add all the specified types even if some additions fail.

Input

* AddResourceTypesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + TypeURI+ – The URI of the types that are to be added to the resource.

Output

* AddResourceTypesResponse
  + AddResourceTypesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - AddResourceTypesDatasetData – The data returned in response to the request.
      * TypeResult+ – For each one of the types that were to be added:
        + TypeURI – The URI of the type.
        + Result – The result of the resource type addition operation for the specified type. Possible values for this are: success; and type not added as the consumer is not authorized.
    - NumberOfTypesAdded – The number of types that were added to the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::AddResourceDescription

Add a basic description to the resource. The description is made up of the values of the following properties: rdfs:label, rdfs:comment, rdfs:seeAlso, and rdfs:isDefinedBy. The consumer can decide how the addition will be made: append new values; replace previous values; and replace whole description. An attempt SHOULD be made to add all the specified property values even if some additions fail.

Input

* AddResourceDescriptionRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + Description – The basic description of the resource.
    - Label\* – The labels of the resource.
    - Comment\* – The comments of the resource.
    - SeeAlso\* – Further resources that might provide additional information about the resource.
    - IsDefinedBy\* – Further resources that identify RDF vocabularies in which the resource is described.
  + SubstituteDescription? – An OPTIONAL parameter that specifies how the service has to add the description. Possible values for this are: append new values to existing ones; update existing values with new ones, that is, if new values are provided for any property, these will replace existing values of this property; replace whole description with the new one, that is, existing description values SHALL be removed, and those new specified SHALL be added. By default, the operation SHALL append values.

Output

* AddResourceDescriptionResponse
  + AddResourceDescriptionDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - AddResourceDescriptionDatasetData – The data returned in response to the request.
      * Result – The result of the resource description addition operation. Possible values for this are: success; and description not added as the consumer is not authorized.
    - SubstituteDescription – How the service added the resource description.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::AddPropertyValues

Add property values to the resource. An attempt SHOULD be made to add all the specified property values even if some additions fail.

Input

* AddPropertyValuesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + Property+ – For each one of the property whose values will be added to the resource:
    - PropertyURI – The property URI.
    - PropertyValue+ – The values of the property that will be added to the resource.

Output

* AddPropertyValuesResponse
  + AddPropertyValuesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - AddPropertyValuesDatasetData – The data returned in response to the request.
      * PropertyResult+ – For each one of the properties whose values were to be added to the resource:
        + PropertyURI – The URI of the property.
        + Result – The result of the property value addition operation for the specified property. Possible values for this are: success; and property values not added as the consumer is not authorized.
    - NumberOfPropertyValuesAdded – The number of property values that were added to the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::RemoveResourceTypes

Remove resource types. If no types are specified, all the types defined for the resource SHALL be removed. Only explicit types will be removed. In no case the class resources that define the specified types are removed from the repository. An attempt SHOULD be made to remove all the specified types even if some removals fail.

Input

* RemoveResourceTypesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + TypeURI\* – An OPTIONAL set of URIs of the types that are to be removed from the resource.

Output

* RemoveResourceTypesResponse
  + RemoveResourceTypesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - RemoveResourceTypesDatasetData – The data returned in response to the request.
      * ParentResult\* – For each one of the types that were to be removed:
        + TypeURI – The URI of the type.
        + Result – The result of the resource type removal operation for the specified type. Possible values for this are: success; and type not removed as the consumer is not authorized.
    - NumberOfTypesRemoved – The number of types that were removed from the resource.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::RemovePropertyValues

Remove resource property values. If no values are specified, all the values defined for the resource and property SHALL be removed. If no property is specified, all the specified values associated to the resource regardless of the property SHALL be removed. It is important to mention that the values that are resources themselves will not be removed from the repository. Only the values that are literals will be physically removed from the repository. An attempt SHOULD be made to remove all of the specified property values even if some removals fail.

Input

* RemovePropertyValuesRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + Property+ – The properties whose values for the specified resources will be removed. If no property is specified, all the property values of the specified resources will be removed. For each of these properties:
    - PropertyURI – The URI of the property.
    - PropertyValue\* – The values that will be removed. If no values are specified, all of them will be removed.

Output

* RemovePropertyValuesResponse
  + RemovePropertyValuesDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - RemovePropertyValuesDatasetData – The data returned in response to the request.
      * PropertyResult+ – For each one of the properties whose values were to be removed from the resource:
        + PropertyURI – The resource URI.
        + Result – The result of the resource properties removal operation. Possible values for this are: success; no matching property values found; all property values removed; and no property values removed as the consumer is not authorized.
    - NumberOfValuesRemoved – The number of values removed for the property.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::ArePropertiesDefined

Check if the specified properties are defined for this resource, that is, there exists at least a value associated to the resource by means of the specified properties. The resolution mechanism to be used for providing the response can be specified by identifying the reasoning capabilities that SHALL be applied by using the request's parameter provided. The parameter relies in the predefined RDF Schema entailment rules [RDF Semantics], which allow reasoning over taxonomies of properties, thus widening or narrowing the expected result set. An attempt SHOULD be made to check all of the specified properties even if some checks fail.

Input

* ArePropertiesDefinedRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + PropertyURI+ – The URI of the specific properties that are going to be checked.
  + InferSuperProperties? – An OPTIONAL parameter that determines whether or not the service MUST infer the super properties of the defined properties. By default, this parameter is set to false.

Output

* ArePropertiesDefinedResponse
  + ArePropertiesDefinedDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - ArePropertiesDefinedDatasetData – The data returned in response to the request.
      * PropertyResult+ – For each one of the properties that were to be checked:
        + PropertyURI – The URI of the property.
        + Result – The result of the resource type defined property check operation for the specified property. Possible values for this are: true, false, and property could not be checked as the consumer is not authorized.
    - InferSuperProperties – Whether or not super properties were taken into consideration.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

### ResourceAccess::ArePropertiesDefinable

Check if the specified properties are definable for this resource, that is, if any class in the domain of the properties is parent of the resource. The resolution mechanism to be used for providing the response can be specified by identifying the reasoning capabilities that SHALL be applied by using the request's parameters provided. Both parameters rely in the predefined RDF Schema entailment rules [RDF Semantics], which allow reasoning over taxonomies of both classes and properties, thus widening or narrowing the expected result set. An attempt SHOULD be made to check all of the specified properties even if some checks fail.

Input

* ArePropertiesDefinableRequest
  + DataResourceAbstractName – The abstract name of the resource the message is directed to.
  + DatasetFormatURI – The format of the response dataset data.
  + PropertyURI+ – The URI of the specific properties that are going to be checked.
  + InferInheritedProperties? – An OPTIONAL parameter that determines whether or not the service MUST infer the properties that are inherited from parent classes of the resource. By default this paramter is set to false.
  + InferSubProperties? – An OPTIONAL parameter that determines whether or not the service MUST infer the sub properties of the definable properties. By default this parameter is set to false.

Output

* ArePropertiesDefinableResponse
  + ArePropertiesDefinableDataset
    - DatasetFormatURI – The URI of the dataset format being returned as defined in the DatasetMap.
    - ArePropertiesDefinableDatasetData – The data returned in response to the request.
      * PropertyResult+ – For each one of the properties that were to be checked:
        + PropertyURI – The URI of the property.
        + Result – The result of the resource type defined property check operation for the specified property. Possible values for this are: true, false, and property could not be checked as the consumer is not authorized.
    - InferInheritedProperties – Whether or not inherited properties were taken into consideration.
    - InferSubProperties – Whether or not sub properties were taken into consideration.

Faults

* InvalidResourceNameFault - The supplied resource resource is not known to the service.
* DataResourceUnavailableFault - The specified data resource is unavailable.
* ServiceBusyFault - The service is already processing a request and Concurrent Access is false.

# Mapping to WSDL

For a mapping to the WSDL proposal see the following sections:

* RepositoryCollection
  + XML Schema – Appendix A.1
  + WSDL – Appendix A.2
* Repository
  + XML Schema – Appendix B.1
  + WSDL – Appendix B.2
* Resource
  + XML Schema – Appendix C.1
  + WSDL – Appendix C.2

# Security Considerations

The realizations of an RDF(S) grid data access service will use standard web service security mechanisms as specified by other standards bodies. The assumption is that these standards will also indicate how to make information related to authentication, authorization security, etc., available.

# Conclusion

This document has discussed a specialization of the interfaces defined in the Web Services Data Access and Integration document [WS-DAI] and the additional capabilities required to properly address RDF(S) data resources by means of a set of ontology handling primitives, based on the RDF(S) model defined in [RDF Concepts, RDF Schema].

To facilitate the adoption of the proposal, the realization is organized as a set of profiles, which provide different granularity degrees for accessing RDF(S) data resources, so that users and developers can decide – according to their requirements – which parts of the specification to use or implement, without compromising interoperability. In particular, this document has covered the first and most basic profile (Profile 0), which provides ground over which the other profiles shall be developed in the future.

# Editor Information

Miguel Esteban Gutiérrez

Ontology Engineering Group

Facultad de Informática

Universidad Politécnica de Madrid

Campus de Montegancedo s/n

28660 – Boadilla del Monte – Madrid

Spain

Asunción Gómez Pérez

Ontology Engineering Group

Facultad de Informática

Universidad Politécnica de Madrid

Campus de Montegancedo s/n

28660 – Boadilla del Monte – Madrid

Spain

# Contributors

Isao Kojima, AIST

Said Mirza, AIST

Oscar Corcho, UPM

# Acknowledgements

This work has been funded by the OntoGrid Project (FP-511513)[[2]](#footnote-3), and many members have collaborated during the early stages of development. Thanks go to Wei Xing, Paolo Missier, and Óscar Muñoz.

The Database Access and Integration Services (DAIS) Working Group of the Open Grid Forum has been active over several years, and many people have contributed to discussions within the group, including but not limited to: Mario Antonioletti, Amy Krause, Norman Paton, and Dave Pearson.

# Glossary

Recommended but not required.

# Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

# Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

# Full Copyright Notice

Copyright (C) Open Grid Forum (2009). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the OGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the OGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

# References

[OGSA]

I. Foster, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, H. Kishimoto, F. Maciel, A. Savva, F. Siebenlist, R. Subramanian, J. Treadwell, and J.J. Von Reich. *The Open Grid Services Architecture, Version 1.5*. Open Grid Forum, OGSA Working Group. GFD-I.030. 24 July 2006. http://www.ogf.org/documents/GFD.80.pdf

[OGSA Glossary]

J. Treadwell (ed). *Open Grid Services Architecture Glossary of Terms, Version 1.6*. Open Grid Forum, OGSA Working Group. GFD-I.120, 12 December 2007. http://www.ogf.org/documents/GFD.44.pdf

[RFC2119]

S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, Internet Engineering Task Force, RFC 2119, http://www.ietf.org/rfc/rfc2119.txt, March 1997.

[WS-DAI]

M. Antonioletti, M. Atkinson, A. Krause, S. Malaika, S. Laws, D. Pearson, N.W. Paton, and G. Riccardi, *Web Services Data Access and Integration - The Core (WS-DAI) Specification, Version 1.0*. Open Grid Forum, DAIS Working Group. GFD-R-P.074. 20 July 2006

http://www.ogf.org/documents/GFD.74.pdf

[WS-DAIR]

M. Antonioletti, B. Collins, A. Krause, S. Laws, J. Magowan, S. Malaika, and N.W. Paton. *Web Services Data Access and Integration - The Relational Realisation (WS-DAIR) Specification, 1.0*. Open Grid Forum, DAIS Working Group. GFD-R-P.076. 20 July 2006

http://www.ogf.org/documents/GFD.76.pdf

[Querying]

I. Kojima, S. Mirza Pahlevi. *Web Services Data Access and Integration – The RDF(S) Realization (WS-DAIRDFS) RDF(S) Querying Specification, Version 0.2*. Draft. Open Grid Forum, DAIS Working Group. 31 July 2007

https://forge.gridforum.org/sf/go/doc14074?nav=1

[Glossary]

M. Esteban Gutiérrez, I. Kojima, Ó. Muñoz García, S. Mirza Pahlevi. *Data Access and Integration Services – RDFS Acces, Glossary of Terms, Version 0.6*. Draft. Open Grid Forum, DAIS Working Group. 24 January 2008

https://forge.gridforum.org/sf/go/doc14820?nav=1

[RDF Syntax]

D. Beckett, RDF/XML Syntax Specification (Revised), W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/

[RDF Schema]

D. Brickley and R.V. Guha, RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

[RDF Concepts]

J.J. Carroll and G. Klyne, Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[RDF Tests]

J. Grant and D. Beckett, RDF Test Cases, W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-testcases-20040210/

[RDF Semantics]

P. Hayes, RDF Semantics, W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

[RDF Primer]

F. Manola and E. Miller, RDF Primer, W3C Recommendation, B. McBride, World Wide Web Consortium, 2004.

http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

[RFC3987]

M. Duerst and M. Suignard, *Internationalized Resource Identifiers (IRIs)*, Internet Engineering Task Force, RFC 3987, http://www.ietf.org/rfc/rfc3987.txt, January 2005.

[Gamma et al., 1994]

E. Gamma, R. Helm, R. Johnson, J.M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1994.

# RepositoryCollection

# RepositoryCollection XML Schema

xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDFS Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!-- The static properties that describe a RepositoryCollection data service -->

<xs:element name="NumberOfRepositories" type="xs:unsignedLong"/>

<xs:element name="RepositoryCollection">

<xs:complexType>

<xs:sequence>

<xs:element name="Repository" minOccurs="0" maxOccurs="unbounded" type="wro:RepositoryDescriptorType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- The configuration properties that describe a RepositoryCollection data service -->

<!-- The property document of a RepositoryCollection data service -->

<xs:complexType name="RepositoryCollectionPropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:NumberOfRepositories"/>

<xs:element ref="wro:RepositoryCollection"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="RepositoryCollectionPropertyDocument" type="wro:RepositoryCollectionPropertyDocumentType"/>

<xs:complexType name="RepositoryDescriptorType">

<xs:attribute name="name" type="xs:anyURI" use="required"/>

</xs:complexType>

<xs:simpleType name="RepositoryRetrievalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="RepositoryDoesNotExist"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="RepositoryAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="RepositoryUpdated"/>

<xs:enumeration value="RepositoryClash"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="RepositoryRemovalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="RepositoryDoesNotExist"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

# RepositoryCollection WSDL

<wsdl:definitions xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

name="wsdai-rdfs-ontology"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!-- WSDL IMPORTS #################################################################### -->

<wsdl:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" location="..\resources\spec\wsdai\_core\_porttypes.wsdl"/>

<!-- WSDL TYPES ###################################################################### -->

<wsdl:types>

<xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDF(S) Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!--Include the external WS-DAI-RDF(S) Resource Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_repositorycollection\_types.xsd"/>

<!-- ##################################### -->

<!-- ### GetRepositories Message Types ### -->

<!-- ##################################### -->

<xs:element name="GetRepositoriesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="RepositoryName" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetRepositoriesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="RepositoryResponse" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="RepositoryName" type="xs:anyURI"/>

<xs:element name="Result" type="wro:RepositoryRetrievalResultType"/>

<xs:element name="Data">

<xs:complexType>

<xs:sequence>

<xs:any namespace="##any" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetRepositoriesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfRepositoriesRetrieved" type="xs:unisgnedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetRepositoriesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetRepositoriesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ##################################### -->

<!-- ### AddRepositories Message Types ### -->

<!-- ##################################### -->

<xs:element name="AddRepositoriesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="Repository" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="RepositoryName" type="xs:anyURI"/>

<xs:element name="Data">

<xs:complexType>

<xs:sequence>

<xs:any namespace="##any" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="OverwriteRepositories" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddRepositoriesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="RepositoryResponse" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="RepositoryName" type="xs:anyURI"/>

<xs:element name="Result" type="wro:RepositoryAdditionResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddRepositoriesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="OverwriteRepositories" type="xs:boolean"/>

<xs:element name="NumberOfRepositoriesAdded" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddRepositoriesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:AddRepositoriesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ######################################## -->

<!-- ### RemoveRepositories Message Types ### -->

<!-- ######################################## -->

<xs:element name="RemoveRepositoriesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="RepositoryName" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveRepositoriesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="RepositoryResponse" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="RepositoryName" type="xs:anyURI"/>

<xs:element name="Result" type="wro:RepositoryRemovalResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveRepositoriesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfRepositoriesRemoved" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveRepositoriesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:RemoveRepositoriesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ########################################## -->

<!-- ### GetRepositoryFactory Message Types ### -->

<!-- ########################################## -->

<xs:element name="GetRepositoryFactoryRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:FactoryRequestType">

<xs:sequence>

<xs:element name="RepositoryName" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetRepositoryFactoryResponse" type="wsdai:DataResourceAddressListType"/>

</xs:schema>

</wsdl:types>

<!-- WSDL MESSAGES ################################################################### -->

<wsdl:message name="GetRepositoryCollectionPropertyDocumentRequest">

<wsdl:part name="GetRepositoryCollectionPropertyDocumentRequest" element="wsdai:GetDataResourcePropertyDocumentRequest"/>

</wsdl:message>

<wsdl:message name="GetRepositoryCollectionPropertyDocumentResponse">

<wsdl:part name="GetRepositoryCollectionPropertyDocumentResponse" element="wro:RepositoryCollectionPropertyDocument"/>

</wsdl:message>

<wsdl:message name="GetRepositoriesRequest">

<wsdl:part name="GetRepositoriesRequest" element="wro:GetRepositoriesRequest"/>

</wsdl:message>

<wsdl:message name="GetRepositoriesResponse">

<wsdl:part name="GetRepositoriesResponse" element="wro:GetRepositoriesResponse"/>

</wsdl:message>

<wsdl:message name="AddRepositoriesRequest">

<wsdl:part name="AddRepositoriesRequest" element="wro:AddRepositoriesRequest"/>

</wsdl:message>

<wsdl:message name="AddRepositoriesResponse">

<wsdl:part name="AddRepositoriesResponse" element="wro:AddRepositoriesResponse"/>

</wsdl:message>

<wsdl:message name="RemoveRepositoriesRequest">

<wsdl:part name="RemoveRepositoriesRequest" element="wro:RemoveRepositoriesRequest"/>

</wsdl:message>

<wsdl:message name="RemoveRepositoriesResponse">

<wsdl:part name="RemoveRepositoriesResponse" element="wro:RemoveRepositoriesResponse"/>

</wsdl:message>

<wsdl:message name="GetRepositoryFactoryRequest">

<wsdl:part name="GetRepositoryFactoryRequest" element="wro:GetRepositoryFactoryRequest"/>

</wsdl:message>

<wsdl:message name="GetRepositoryFactoryResponse">

<wsdl:part name="GetRepositoryFactoryResponse" element="wro:GetRepositoryFactoryResponse"/>

</wsdl:message>

<wsdl:message name="RepositoryDoesNotExistFault">

<wsdl:part name="RepositoryDoesNotExistFault" element="wro:RepositoryDoesNotExistFault"/>

</wsdl:message>

<wsdl:message name="InvalidResourceNameFault">

<wsdl:part name="InvalidResourceNameFault" element="wsdai:InvalidResourceNameFault"/>

</wsdl:message>

<wsdl:message name="DataResourceUnavailableFault">

<wsdl:part name="DataResourceUnavailableFault" element="wsdai:DataResourceUnavailableFault"/>

</wsdl:message>

<wsdl:message name="ServiceBusyFault">

<wsdl:part name="ServiceBusyFault" element="wsdai:ServiceBusyFault"/>

</wsdl:message>

<wsdl:message name="InvalidPortTypeQNameFault">

<wsdl:part name="InvalidPortTypeQNameFault" element="wsdai:InvalidPortTypeQNameFault"/>

</wsdl:message>

<wsdl:message name="InvalidConfigurationDocumentFault">

<wsdl:part name="InvalidConfigurationDocumentFault" element="wsdai:InvalidConfigurationDocumentFault"/>

</wsdl:message>

<wsdl:message name="NotAuthorizedFault">

<wsdl:part name="NotAuthorizedFault" element="wsdai:NotAuthorizedFault"/>

</wsdl:message>

<!-- WSDL PORT TYPES ################################################################# -->

<wsdl:portType name="RepositoryCollectionAccessPT">

<wsdl:operation name="GetRepositoryCollectionPropertyDocument">

<wsdl:input name="GetRepositoryCollectionPropertyDocumentRequest" message="wro:GetRepositoryCollectionPropertyDocumentRequest"/>

<wsdl:output name="GetRepositoryCollectionPropertyDocumentResponse" message="wro:GetRepositoryCollectionPropertyDocumentResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="NotAuthorizedFault" message="wsdai:NotAuthorizedFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetRepositories">

<wsdl:input name="GetRepositoriesRequest" message="wro:GetRepositoriesRequest"/>

<wsdl:output name="GetRepositoriesResponse" message="wro:GetRepositoriesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="AddRepositories">

<wsdl:input name="AddRepositoriesRequest" message="wro:AddRepositoriesRequest"/>

<wsdl:output name="AddRepositoriesResponse" message="wro:AddRepositoriesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="RemoveRepositories">

<wsdl:input name="RemoveRepositoriesRequest" message="wro:RemoveRepositoriesRequest"/>

<wsdl:output name="RemoveRepositoriesResponse" message="wro:RemoveRepositoriesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

</wsdl:portType>

<wsdl:portType name="RepositoryCollectionFactoryPT">

<wsdl:operation name="GetRepositoryFactory">

<wsdl:input name="GetRepositoryFactoryRequest" message="wro:GetRepositoryFactoryRequest"/>

<wsdl:output name="GetRepositoryFactoryResponse" message="wro:GetRepositoryFactoryResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="InvalidPortTypeQNameFault" message="wsdai:InvalidPortTypeQNameFault"/>

<wsdl:fault name="InvalidConfigurationDocumentFault" message="wsdai:InvalidConfigurationDocumentFault"/>

<wsdl:fault name="NotAuthorizedFault" message="wsdai:NotAuthorizedFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

<wsdl:fault name="RepositoryDoesNotExistFault" message="wro:RepositoryDoesNotExistFault"/>

</wsdl:operation>

</wsdl:portType>

</wsdl:definitions>

# Repository

# Repository XML Schema

<xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDFS Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!-- The static properties that describe a Repository data service -->

<xs:element name="RepositoryName" type="xs:anyURI"/>

<!-- The configuration properties that describe a Repository data service -->

<!-- The property document of a Repository data service -->

<xs:complexType name="RepositoryPropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:RepositoryName"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="RepositoryPropertyDocument" type="wro:RepositoryPropertyDocumentType"/>

<xs:simpleType name="ValueFilterSemanticsType">

<xs:restriction base="xs:string">

<xs:enumeration value="OR"/>

<xs:enumeration value="AND"/>

<xs:enumeration value="XOR"/>

<xs:enumeration value="NONE"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="PropertyFilterSemanticsType">

<xs:restriction base="xs:string">

<xs:enumeration value="OR"/>

<xs:enumeration value="AND"/>

<xs:enumeration value="XOR"/>

<xs:enumeration value="NONE"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourceRetrievalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="NoMatchingResourcesFound"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourcePropertiesRetrievalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="NoMatchingResourcePropertiesFound"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourceAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="ResourceUpdated"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourceRemovalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourcePropertiesRemovalResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="NoMatchingResourcePropertiesFound"/>

<xs:enumeration value="AllPropertyValuesRemoved"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

# Repository WSDL

<wsdl:definitions xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

name="wsdai-rdfs-ontology"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!-- WSDL IMPORTS #################################################################### -->

<wsdl:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" location="..\resources\spec\wsdai\_core\_porttypes.wsdl"/>

<!-- WSDL TYPES ###################################################################### -->

<wsdl:types>

<xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDF(S) Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!--Include the external WS-DAI-RDF(S) Resource Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_repository\_types.xsd"/>

<!-- ################################## -->

<!-- ### GetResources Message Types ### -->

<!-- ################################## -->

<xs:element name="GetResourcesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="ResourceSet" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="PropertyFilter" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValueFilter" minOccurs="0" maxOccurs="unbounded" type="wro:PropertyValueType"/>

<xs:element name="ValueFilterSemantics" minOccurs="0" maxOccurs="1" type="wro:ValueFilterSemanticsType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="PropertyFilterSemantics" minOccurs="0" maxOccurs="1" type="wro:PropertyFilterSemanticsType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="RetrieveResourceProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ResourceSetResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="Result" type="wro:ResourceRetrievalResultType"/>

<xs:element name="Resource" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<xs:element name="DefinedProperty" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="1" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="RetrieveResourceProperties" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetResourcesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ########################################### -->

<!-- ### GetResourceProperties Message Types ### -->

<!-- ########################################### -->

<xs:element name="GetResourcePropertiesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="ResourceSet" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="PropertyURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcePropertiesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ResourceSetResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="Resource" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourcePropertiesRetrievalResultType"/>

<xs:element name="DefinedProperty" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="1" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcePropertiesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourcePropertiesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetResourcePropertiesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ################################## -->

<!-- ### AddResources Message Types ### -->

<!-- ################################## -->

<xs:element name="AddResourcesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="Resource" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" minOccurs="0" maxOccurs="1" type="xs:anyURI"/>

<xs:element name="Property" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="1" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourcesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ResourceResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourceAdditionResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourcesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfResourcesAdded" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourcesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:AddResourcesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ##################################### -->

<!-- ### RemoveResources Message Types ### -->

<!-- ##################################### -->

<xs:element name="RemoveResourcesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="ResourceSet" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="PropertyFilter" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValueFilter" minOccurs="0" maxOccurs="unbounded" type="wro:PropertyValueType"/>

<xs:element name="ValueFilterSemantics" minOccurs="0" maxOccurs="1" type="wro:ValueFilterSemanticsType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="PropertyFilterSemantics" minOccurs="0" maxOccurs="1" type="wro:PropertyFilterSemanticsType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ResourceSetResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="Resource" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourceRemovalResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfRemovedResources" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:RemoveResourcesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ############################################## -->

<!-- ### RemoveResourceProperties Message Types ### -->

<!-- ############################################## -->

<xs:element name="RemoveResourcePropertiesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="ResourceSet" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="Property" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="0" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcePropertiesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ResourceSetResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="Resource" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourcePropertiesRemovalResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcePropertiesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourcePropertiesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:RemoveResourcePropertiesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ######################################## -->

<!-- ### GetResourceFactory Message Types ### -->

<!-- ######################################## -->

<xs:element name="GetResourceFactoryRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:FactoryRequestType">

<xs:sequence>

<xs:element name="ResourceURI" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceFactoryResponse" type="wsdai:DataResourceAddressListType"/>

</xs:schema>

</wsdl:types>

<!-- WSDL MESSAGES ################################################################### -->

<wsdl:message name="GetRepositoryPropertyDocumentRequest">

<wsdl:part name="GetRepositoryPropertyDocumentRequest" element="wsdai:GetDataResourcePropertyDocumentRequest"/>

</wsdl:message>

<wsdl:message name="GetRepositoryPropertyDocumentResponse">

<wsdl:part name="GetRepositoryPropertyDocumentResponse" element="wro:RepositoryPropertyDocument"/>

</wsdl:message>

<wsdl:message name="GetResourcesRequest">

<wsdl:part name="GetResourcesRequest" element="wro:GetResourcesRequest"/>

</wsdl:message>

<wsdl:message name="GetResourcesResponse">

<wsdl:part name="GetResourcesResponse" element="wro:GetResourcesResponse"/>

</wsdl:message>

<wsdl:message name="GetResourcePropertiesRequest">

<wsdl:part name="GetResourcePropertiesRequest" element="wro:GetResourcePropertiesRequest"/>

</wsdl:message>

<wsdl:message name="GetResourcePropertiesResponse">

<wsdl:part name="GetResourcePropertiesResponse" element="wro:GetResourcePropertiesResponse"/>

</wsdl:message>

<wsdl:message name="AddResourcesRequest">

<wsdl:part name="AddResourcesRequest" element="wro:AddResourcesRequest"/>

</wsdl:message>

<wsdl:message name="AddResourcesResponse">

<wsdl:part name="AddResourcesResponse" element="wro:AddResourcesResponse"/>

</wsdl:message>

<wsdl:message name="RemoveResourcesRequest">

<wsdl:part name="RemoveResourcesRequest" element="wro:RemoveResourcesRequest"/>

</wsdl:message>

<wsdl:message name="RemoveResourcesResponse">

<wsdl:part name="RemoveResourcesResponse" element="wro:RemoveResourcesResponse"/>

</wsdl:message>

<wsdl:message name="RemoveResourcePropertiesRequest">

<wsdl:part name="RemoveResourcePropertiesRequest" element="wro:RemoveResourcePropertiesRequest"/>

</wsdl:message>

<wsdl:message name="RemoveResourcePropertiesResponse">

<wsdl:part name="RemoveResourcePropertiesResponse" element="wro:RemoveResourcePropertiesResponse"/>

</wsdl:message>

<wsdl:message name="GetResourceFactoryRequest">

<wsdl:part name="GetResourceFactoryRequest" element="wro:GetResourceFactoryRequest"/>

</wsdl:message>

<wsdl:message name="GetResourceFactoryResponse">

<wsdl:part name="GetResourceFactoryResponse" element="wro:GetResourceFactoryResponse"/>

</wsdl:message>

<wsdl:message name="ResourceDoesNotExistFault">

<wsdl:part name="ResourceDoesNotExistFault" element="wro:ResourceDoesNotExistFault"/>

</wsdl:message>

<wsdl:message name="InvalidResourceNameFault">

<wsdl:part name="InvalidResourceNameFault" element="wsdai:InvalidResourceNameFault"/>

</wsdl:message>

<wsdl:message name="DataResourceUnavailableFault">

<wsdl:part name="DataResourceUnavailableFault" element="wsdai:DataResourceUnavailableFault"/>

</wsdl:message>

<wsdl:message name="ServiceBusyFault">

<wsdl:part name="ServiceBusyFault" element="wsdai:ServiceBusyFault"/>

</wsdl:message>

<wsdl:message name="InvalidPortTypeQNameFault">

<wsdl:part name="InvalidPortTypeQNameFault" element="wsdai:InvalidPortTypeQNameFault"/>

</wsdl:message>

<wsdl:message name="InvalidConfigurationDocumentFault">

<wsdl:part name="InvalidConfigurationDocumentFault" element="wsdai:InvalidConfigurationDocumentFault"/>

</wsdl:message>

<wsdl:message name="NotAuthorizedFault">

<wsdl:part name="NotAuthorizedFault" element="wsdai:NotAuthorizedFault"/>

</wsdl:message>

<!-- WSDL PORT TYPES ################################################################# -->

<wsdl:portType name="RepositoryAccessPT">

<wsdl:operation name="GetRepositoryPropertyDocument">

<wsdl:input name="GetRepositoryPropertyDocumentRequest" message="wro:GetRepositoryPropertyDocumentRequest"/>

<wsdl:output name="GetRepositoryPropertyDocumentResponse" message="wro:GetRepositoryPropertyDocumentResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="NotAuthorizedFault" message="wsdai:NotAuthorizedFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetResources">

<wsdl:input name="GetResourcesRequest" message="wro:GetResourcesRequest"/>

<wsdl:output name="GetResourcesResponse" message="wro:GetResourcesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetResourceProperties">

<wsdl:input name="GetResourcePropertiesRequest" message="wro:GetResourcePropertiesRequest"/>

<wsdl:output name="GetResourcePropertiesResponse" message="wro:GetResourcePropertiesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="AddResources">

<wsdl:input name="AddResourcesRequest" message="wro:AddResourcesRequest"/>

<wsdl:output name="AddResourcesResponse" message="wro:AddResourcesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="RemoveResources">

<wsdl:input name="RemoveResourcesRequest" message="wro:RemoveResourcesRequest"/>

<wsdl:output name="RemoveResourcesResponse" message="wro:RemoveResourcesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="RemoveResourceProperties">

<wsdl:input name="RemoveResourcePropertiesRequest" message="wro:RemoveResourcePropertiesRequest"/>

<wsdl:output name="RemoveResourcePropertiesResponse" message="wro:RemoveResourcePropertiesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

</wsdl:portType>

<wsdl:portType name="RepositoryFactoryPT">

<wsdl:operation name="GetResourceFactory">

<wsdl:input name="GetResourceFactoryRequest" message="wro:GetResourceFactoryRequest"/>

<wsdl:output name="GetResourceFactoryResponse" message="wro:GetResourceFactoryResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="InvalidPortTypeQNameFault" message="wsdai:InvalidPortTypeQNameFault"/>

<wsdl:fault name="InvalidConfigurationDocumentFault" message="wsdai:InvalidConfigurationDocumentFault"/>

<wsdl:fault name="NotAuthorizedFault" message="wsdai:NotAuthorizedFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

<wsdl:fault name="ResourceDoesNotExistFault" message="wro:ResourceDoesNotExistFault"/>

</wsdl:operation>

</wsdl:portType>

</wsdl:definitions>

# Resource

# Resource XML Schema

<xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDFS Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!-- The static properties that describe a Resource data service -->

<xs:element name="RepositoryName" type="xs:anyURI"/>

<xs:element name="ResourceURI" type="xs:anyURI"/>

<!-- The configuration properties that describe a Resource data service -->

<!-- The property document of a Resource data service -->

<xs:complexType name="ResourcePropertyDocumentType">

<xs:complexContent>

<xs:extension base="wsdai:PropertyDocumentType">

<xs:sequence>

<xs:element ref="wro:RepositoryName"/>

<xs:element ref="wro:ResourceURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="ResourcePropertyDocument" type="wro:ResourcePropertyDocumentType"/>

<xs:simpleType name="DescriptionModificationOptionsType">

<xs:restriction base="xs:string">

<xs:enumeration value="APPEND\_VALUES"/>

<xs:enumeration value="UPDATE\_VALUES"/>

<xs:enumeration value="REPLACE\_DESCRIPTION"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourceDescriptionAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourcePropertyValuesAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourceDescriptionAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ResourcePropertyValuesAdditionResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="Success"/>

<xs:enumeration value="NoMatchingPropertyValuesFound"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="DefinedPropertyChecklResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="true"/>

<xs:enumeration value="false"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="DefinablePropertyChecklResultType">

<xs:restriction base="xs:string">

<xs:enumeration value="true"/>

<xs:enumeration value="false"/>

<xs:enumeration value="ConsumerNotAuthorized"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

# Resource WSDL

<wsdl:definitions xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

name="wsdai-rdfs-ontology"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!-- WSDL IMPORTS #################################################################### -->

<wsdl:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" location="..\resources\spec\wsdai\_core\_porttypes.wsdl"/>

<!-- WSDL TYPES ###################################################################### -->

<wsdl:types>

<xs:schema xmlns:wro="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology"

xmlns:wsdai="http://www.ggf.org/namespaces/2005/12/WS-DAI"

elementFormDefault="qualified"

targetNamespace="http://www.ogf.org/dais/2008/02/ws-dai-rdfs/ontology">

<!--Import the base WS-DAI Schema-->

<xs:import namespace="http://www.ggf.org/namespaces/2005/12/WS-DAI" schemaLocation="..\resources\spec\wsdai\_core\_types.xsd"/>

<!--Include the base WS-DAI-RDF(S) Core Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_types.xsd"/>

<!--Include the external WS-DAI-RDF(S) Resource Schema-->

<xs:include schemaLocation="wsdai\_rdfs\_resource\_types.xsd"/>

<!-- ###################################### -->

<!-- ### GetResourceTypes Message Types ### -->

<!-- ###################################### -->

<xs:element name="GetResourceTypesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="InferParents" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceTypesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ParentURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceTypesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="InferParents" type="xs:boolean"/>

<xs:element name="NumberOfParents" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceTypesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetResourceTypesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ############################################ -->

<!-- ### GetResourceDescription Message Types ### -->

<!-- ############################################ -->

<xs:element name="GetResourceDescriptionRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceDescriptionDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="Description">

<xs:complexType>

<xs:sequence>

<xs:element name="Label" minOccurs="0" maxOccurs="unbounded" type="wro:LiteralType"/>

<xs:element name="Comment" minOccurs="0" maxOccurs="unbounded" type="wro:LiteralType"/>

<xs:element name="SeeAlso" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="IsDefinedBy" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceDescriptionDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetResourceDescriptionResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetResourceDescriptionDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ####################################### -->

<!-- ### GetPropertyValues Message Types ### -->

<!-- ####################################### -->

<xs:element name="GetPropertyValuesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="PropertyURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetPropertyValuesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyResult" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="1" maxOccurs="unbounded" type="wro:ResourceType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetPropertyValuesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfPropertyValuesRetrieved" type="xs:unsgnedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetPropertyValuesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetPropertyValuesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ########################################## -->

<!-- ### GetDefinedProperties Message Types ### -->

<!-- ########################################## -->

<xs:element name="GetDefinedPropertiesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinedPropertiesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinedPropertiesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfDefinedProperties" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinedPropertiesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetDefinedPropertiesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ############################################ -->

<!-- ### GetDefinableProperties Message Types ### -->

<!-- ############################################ -->

<xs:element name="GetDefinablePropertiesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="InferInheritedProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

<xs:element name="InferSubProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinablePropertiesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinablePropertiesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="InferInheritedProperties" type="xs:boolean"/>

<xs:element name="InferSubProperties" type="xs:boolean"/>

<xs:element name="NumberOfDefinableProperties" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="GetDefinablePropertiesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:GetDefinablePropertiesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ###################################### -->

<!-- ### AddResourceTypes Message Types ### -->

<!-- ###################################### -->

<xs:element name="AddResourceTypesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="TypeURI" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceTypesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="TypeResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="TypeURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourceTypeAdditionResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceTypesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfTypesAdded" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceTypesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:AddResourceTypesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ############################################ -->

<!-- ### AddResourceDescription Message Types ### -->

<!-- ############################################ -->

<xs:element name="AddResourceDescriptionRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="Description">

<xs:complexType>

<xs:sequence>

<xs:element name="Label" minOccurs="0" maxOccurs="unbounded" type="wro:LiteralType"/>

<xs:element name="Comment" minOccurs="0" maxOccurs="unbounded" type="wro:LiteralType"/>

<xs:element name="SeeAlso" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="IsDefinedBy" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="SubstituteDescription" minOccurs="0" maxOccurs="1" type="wro:DescriptionModificationOptionsType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceDescriptionDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="Result" type="wro:ResourceDescriptionAdditionResultType"/>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceDescriptionDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="SubstituteDescription" type="wro:DescriptionModificationOptionsType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddResourceDescriptionResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:AddResourceDescriptionDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ####################################### -->

<!-- ### AddPropertyValues Message Types ### -->

<!-- ####################################### -->

<xs:element name="AddPropertyValuesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="Property" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="1" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddPropertyValuesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourcePropertyValuesAdditionResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddPropertyValuesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfPropertyValuesAdded" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="AddPropertyValuesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:AddPropertyValuesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ######################################### -->

<!-- ### RemoveResourceTypes Message Types ### -->

<!-- ######################################### -->

<xs:element name="RemoveResourceTypesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="TypeURI" minOccurs="0" maxOccurs="unbounded" type="xs:anyURI"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourceTypesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="ParentResult" minOccurs="0" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="TypeURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourceTypeRemovalResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourceTypesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfTypesRemoved" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemoveResourceTypesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:RemoveResourceTypesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ########################################## -->

<!-- ### RemovePropertyValues Message Types ### -->

<!-- ########################################## -->

<xs:element name="RemovePropertyValuesRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="Property" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="PropertyValue" minOccurs="0" maxOccurs="unbounded" type="wro:PropertyValueType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemovePropertyValuesDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:ResourcePropertiesRemovalResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemovePropertyValuesDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="NumberOfValuesRemoved" type="xs:unsignedLong"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="RemovePropertyValuesResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:RemovePropertyValuesDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ########################################## -->

<!-- ### ArePropertiesDefined Message Types ### -->

<!-- ########################################## -->

<xs:element name="ArePropertiesDefinedRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="PropertyURI" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="InferSuperProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinedDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="Result" type="wro:DefinedPropertyCheckResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinedDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="InferSuperProperties" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinedResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:ArePropertiesDefinedDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<!-- ############################################ -->

<!-- ### ArePropertiesDefinable Message Types ### -->

<!-- ############################################ -->

<xs:element name="ArePropertiesDefinableRequest">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:RequestType">

<xs:sequence>

<xs:element name="PropertyURI" minOccurs="1" maxOccurs="unbounded" type="xs:anyURI"/>

<xs:element name="InferInheritedProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

<xs:element name="InferSubProperties" minOccurs="0" maxOccurs="1" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinableDatasetData" substitutionGroup="wsdai:DatasetData">

<xs:complexType mixed="true">

<xs:complexContent>

<xs:restriction base="wsdai:DatasetDataType">

<xs:sequence>

<xs:element name="PropertyResult" minOccurs="1" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="PropertyURI" type="xs:anyURI"/>

<xs:element name="Result" type="DefinablePropertyCheckResultType"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:restriction>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinableDataset" substitutionGroup="wsdai:Dataset">

<xs:complexType>

<xs:complexContent>

<xs:extension base="wsdai:DatasetType">

<xs:sequence>

<xs:element name="InferInheritedProperties" type="xs:boolean"/>

<xs:element name="InferSubProperties" type="xs:boolean"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:element>

<xs:element name="ArePropertiesDefinableResponse">

<xs:complexType>

<xs:sequence>

<xs:element ref="wro:ArePropertiesDefinableDataset"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

</wsdl:types>

<!-- WSDL MESSAGES ################################################################### -->

<wsdl:message name="GetResourcePropertyDocumentRequest">

<wsdl:part name="GetResourcePropertyDocumentRequest" element="wsdai:GetDataResourcePropertyDocumentRequest"/>

</wsdl:message>

<wsdl:message name="GetResourcePropertyDocumentResponse">

<wsdl:part name="GetResourcePropertyDocumentResponse" element="wro:ResourcePropertyDocument"/>

</wsdl:message>

<wsdl:message name="GetResourceTypesRequest">

<wsdl:part name="GetResourceTypesRequest" element="wro:GetResourceTypesRequest"/>

</wsdl:message>

<wsdl:message name="GetResourceTypesResponse">

<wsdl:part name="GetResourceTypesResponse" element="wro:GetResourceTypesResponse"/>

</wsdl:message>

<wsdl:message name="GetResourceDescriptionRequest">

<wsdl:part name="GetResourceDescriptionRequest" element="wro:GetResourceDescriptionRequest"/>

</wsdl:message>

<wsdl:message name="GetResourceDescriptionResponse">

<wsdl:part name="GetResourceDescriptionResponse" element="wro:GetResourceDescriptionResponse"/>

</wsdl:message>

<wsdl:message name="GetPropertyValuesRequest">

<wsdl:part name="GetPropertyValuesRequest" element="wro:GetPropertyValuesRequest"/>

</wsdl:message>

<wsdl:message name="GetPropertyValuesResponse">

<wsdl:part name="GetPropertyValuesResponse" element="wro:GetPropertyValuesResponse"/>

</wsdl:message>

<wsdl:message name="GetDefinedPropertiesRequest">

<wsdl:part name="GetDefinedPropertiesRequest" element="wro:GetDefinedPropertiesRequest"/>

</wsdl:message>

<wsdl:message name="GetDefinedPropertiesResponse">

<wsdl:part name="GetDefinedPropertiesResponse" element="wro:GetDefinedPropertiesResponse"/>

</wsdl:message>

<wsdl:message name="GetDefinablePropertiesRequest">

<wsdl:part name="GetDefinablePropertiesRequest" element="wro:GetDefinablePropertiesRequest"/>

</wsdl:message>

<wsdl:message name="GetDefinablePropertiesResponse">

<wsdl:part name="GetDefinablePropertiesResponse" element="wro:GetDefinablePropertiesResponse"/>

</wsdl:message>

<wsdl:message name="AddResourceTypesRequest">

<wsdl:part name="AddResourceTypesRequest" element="wro:AddResourceTypesRequest"/>

</wsdl:message>

<wsdl:message name="AddResourceTypesResponse">

<wsdl:part name="AddResourceTypesResponse" element="wro:AddResourceTypesResponse"/>

</wsdl:message>

<wsdl:message name="AddResourceDescriptionRequest">

<wsdl:part name="AddResourceDescriptionRequest" element="wro:AddResourceDescriptionRequest"/>

</wsdl:message>

<wsdl:message name="AddResourceDescriptionResponse">

<wsdl:part name="AddResourceDescriptionResponse" element="wro:AddResourceDescriptionResponse"/>

</wsdl:message>

<wsdl:message name="AddPropertyValuesRequest">

<wsdl:part name="AddPropertyValuesRequest" element="wro:AddPropertyValuesRequest"/>

</wsdl:message>

<wsdl:message name="AddPropertyValuesResponse">

<wsdl:part name="AddPropertyValuesResponse" element="wro:AddPropertyValuesResponse"/>

</wsdl:message>

<wsdl:message name="RemoveResourceTypesRequest">

<wsdl:part name="RemoveResourceTypesRequest" element="wro:RemoveResourceTypesRequest"/>

</wsdl:message>

<wsdl:message name="RemoveResourceTypesResponse">

<wsdl:part name="RemoveResourceTypesResponse" element="wro:RemoveResourceTypesResponse"/>

</wsdl:message>

<wsdl:message name="RemovePropertyValuesRequest">

<wsdl:part name="RemovePropertyValuesRequest" element="wro:RemovePropertyValuesRequest"/>

</wsdl:message>

<wsdl:message name="RemovePropertyValuesResponse">

<wsdl:part name="RemovePropertyValuesResponse" element="wro:RemovePropertyValuesResponse"/>

</wsdl:message>

<wsdl:message name="ArePropertiesDefinedRequest">

<wsdl:part name="ArePropertiesDefinedRequest" element="wro:ArePropertiesDefinedRequest"/>

</wsdl:message>

<wsdl:message name="ArePropertiesDefinedResponse">

<wsdl:part name="ArePropertiesDefinedResponse" element="wro:ArePropertiesDefinedResponse"/>

</wsdl:message>

<wsdl:message name="ArePropertiesDefinableRequest">

<wsdl:part name="ArePropertiesDefinableRequest" element="wro:ArePropertiesDefinableRequest"/>

</wsdl:message>

<wsdl:message name="ArePropertiesDefinableResponse">

<wsdl:part name="ArePropertiesDefinableResponse" element="wro:ArePropertiesDefinableResponse"/>

</wsdl:message>

<wsdl:message name="InvalidResourceNameFault">

<wsdl:part name="InvalidResourceNameFault" element="wsdai:InvalidResourceNameFault"/>

</wsdl:message>

<wsdl:message name="DataResourceUnavailableFault">

<wsdl:part name="DataResourceUnavailableFault" element="wsdai:DataResourceUnavailableFault"/>

</wsdl:message>

<wsdl:message name="ServiceBusyFault">

<wsdl:part name="ServiceBusyFault" element="wsdai:ServiceBusyFault"/>

</wsdl:message>

<!-- WSDL PORT TYPES ################################################################# -->

<wsdl:portType name="ResourceAccessPT">

<wsdl:operation name="GetResourcePropertyDocument">

<wsdl:input name="GetResourcePropertyDocumentRequest" message="wro:GetResourcePropertyDocumentRequest"/>

<wsdl:output name="GetResourcePropertyDocumentResponse" message="wro:GetResourcePropertyDocumentResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="NotAuthorizedFault" message="wsdai:NotAuthorizedFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetResourceTypes">

<wsdl:input name="GetResourceTypesRequest" message="wro:GetResourceTypesRequest"/>

<wsdl:output name="GetResourceTypesResponse" message="wro:GetResourceTypesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetResourceDescription">

<wsdl:input name="GetResourceDescriptionRequest" message="wro:GetResourceDescriptionRequest"/>

<wsdl:output name="GetResourceDescriptionResponse" message="wro:GetResourceDescriptionResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetPropertyValues">

<wsdl:input name="GetPropertyValuesRequest" message="wro:GetPropertyValuesRequest"/>

<wsdl:output name="GetPropertyValuesResponse" message="wro:GetPropertyValuesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetDefinedProperties">

<wsdl:input name="GetDefinedPropertiesRequest" message="wro:GetDefinedPropertiesRequest"/>

<wsdl:output name="GetDefinedPropertiesResponse" message="wro:GetDefinedPropertiesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="GetDefinableProperties">

<wsdl:input name="GetDefinablePropertiesRequest" message="wro:GetDefinablePropertiesRequest"/>

<wsdl:output name="GetDefinablePropertiesResponse" message="wro:GetDefinablePropertiesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="AddResourceTypes">

<wsdl:input name="AddResourceTypesRequest" message="wro:AddResourceTypesRequest"/>

<wsdl:output name="AddResourceTypesResponse" message="wro:AddResourceTypesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="AddResourceDescription">

<wsdl:input name="AddResourceDescriptionRequest" message="wro:AddResourceDescriptionRequest"/>

<wsdl:output name="AddResourceDescriptionResponse" message="wro:AddResourceDescriptionResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="AddPropertyValues">

<wsdl:input name="AddPropertyValuesRequest" message="wro:AddPropertyValuesRequest"/>

<wsdl:output name="AddPropertyValuesResponse" message="wro:AddPropertyValuesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="RemoveResourceTypes">

<wsdl:input name="RemoveResourceTypesRequest" message="wro:RemoveResourceTypesRequest"/>

<wsdl:output name="RemoveResourceTypesResponse" message="wro:RemoveResourceTypesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="RemovePropertyValues">

<wsdl:input name="RemovePropertyValuesRequest" message="wro:RemovePropertyValuesRequest"/>

<wsdl:output name="RemovePropertyValuesResponse" message="wro:RemovePropertyValuesResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="ArePropertiesDefined">

<wsdl:input name="ArePropertiesDefinedRequest" message="wro:ArePropertiesDefinedRequest"/>

<wsdl:output name="ArePropertiesDefinedResponse" message="wro:ArePropertiesDefinedResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

<wsdl:operation name="ArePropertiesDefinable">

<wsdl:input name="ArePropertiesDefinableRequest" message="wro:ArePropertiesDefinableRequest"/>

<wsdl:output name="ArePropertiesDefinableResponse" message="wro:ArePropertiesDefinableResponse"/>

<wsdl:fault name="InvalidResourceNameFault" message="wsdai:InvalidResourceNameFault"/>

<wsdl:fault name="DataResourceUnavailableFault" message="wsdai:DataResourceUnavailableFault"/>

<wsdl:fault name="ServiceBusyFault" message="wsdai:ServiceBusyFault"/>

</wsdl:operation>

</wsdl:portType>

</wsdl:definitions>

1. RDF(S) defines 3 types of predefined containers: *seq*, *bag*, and *alt*. [↑](#footnote-ref-2)
2. http://www.ontogrid.eu [↑](#footnote-ref-3)