GWD-R
GGF DAIS Working Group

Editors:
Mario Antonioletti, University of Edinburgh
Amy Krause, University of Edinburgh
Shannon Hastings, Ohio State University
Stephen Langella, Ohio State University
Susan Malaika, IBM
Simon Laws, IBM
Norman W Paton, University of Manchester

Category: INFORMATIONAL

19 September 2003

Data Service Specification: The XML Realisation

Status of This Memo

This memo provides information to the Grid community regarding the specification of Grid Database Services. The specification is presently a draft for discussion. It does not define any standards or technical recommendations. Distribution is unlimited.

Copyright Notice

Abstract

Data management systems are central to many applications across multiple domains, and play a significant role in many others. Web services provide implementation neutral facilities for describing, invoking and orchestrating collections of networked resources. The Open Grid Services Infrastructure (OGSI) extends Web Services with consistent interfaces for creating, managing and exchanging information among Grid Services, which are dynamic computational artifacts cast as Web Services. Both Web and Grid service communities stand to benefit from the provision of consistent, agreed service interfaces to data management systems. Such interfaces must support the description and use of data management systems using Web Service standards, taking account of the design conventions and mandatory features of Grid Services. This document presents a specification for a collection of data access interfaces for XML Data Resources, which extends interfaces defined in the Grid Data Service Specification [GDSS], which in turn is based on the OGSA Data Services proposal [Data Services]. This document is presented for discussion within the Global Grid Forum (GGF) Database Access and Integration Services (DAIS) Working Group, with a view to the document evolving to become a proposed recommendation. There are several respects in which the current proposal is incomplete, but it is hoped that the material included is sufficient to allow an informed discussion to take place concerning both its form and substance.

Contents

Abstract....1
1. Introduction....2
2. Notational Conventions....2
3. Specification Overview....3
3.1 Scope of specification....3
3.2 Mapping to Data Service model....3
3.3 Relationships with other specifications....4
4. DataDescription Port Types....4
4.1 XMLCollectionDescription....4
4.2 XMLDocumentDescription....5
5. DataAccess PortTypes....6
5.1 XMLCollectionAccess....6
5.2 XPathAccess....7
5.3 XUpdateAccess....8
5.4 XQueryAccess....8
6. DataFactory PortTypes....9
6.1 XMLCollectionFactory....9
6.2 XMLDocumentFactory....10
6.3 XPathFactory....11
6.4 XQueryXFactory....13
7. Security Considerations....13
8. Conclusion....13
Editor Information....14
Contributors....14
Acknowledgements....15
Intellectual Property Statement....15
Full Copyright Notice....15
References....15

1. Introduction

This document presents a specification for a collection of data access interfaces for XML Data Resources. An XML Data Resource is taken to mean a data source/sink, together with any associated management framework, that exhibits capabilities that are characteristic of XML repositories, e.g., can be queried using XPath or updated using XUpdate or another suitable query/update XML based language. The interfaces instantiate the framework provided by the OGSA Data Services proposal [Data Services], in that interfaces are categorized according to the support they provide for data description, data access, Data Service creation and data management. As such, this document should be read in conjunction with the OGSA Data Services proposal and the generic Grid Data Service Specification [GDSS], which defines various portTypes that are extended in this specification. All of these documents assume some familiarity with the Open Grid Services Infrastructure (OGSI) [OGSI]. The specification does not mandate how the interfaces are composed into services. The proposed interfaces may be used in isolation or in conjunction with others.

2. 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].

GWSDL and XML Schema for operations and Service Data Elements described here is available from the DAIS web site.

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
dais	http://www.ggf.org/namespaces/2003/10/DAIS
gsa	WS-Agreement namespace URI
gwsdl	http://www.ggf.org/namespaces/2003/03/gridWSDLExtensions
http	http://www.w3.org/2002/06/wsdl/http
ogsi	http://www.ggf.org/namespaces/2003/03/OGSI
sd	http://www.ggf.org/namespaces/2003/02/serviceData
xqx	XQueryX namespace URI
xsd	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance
wsp	http://schemas.xmlsoap.org/ws/2002/12/policy

3. Specification Overview

3.1 Scope of specification

This document extends the interfaces presented in the *Grid Data Services Specification* [GDSS] to allow access to and describe XML Data Resources and aligns these interfaces with the base types provided in the *OGSA Data Services* [Data Services] document. The XML Data Resources are assumed to be composed of a hierarchy of collections, which store XML resources such as XML document descriptions and documents.

3.2 Mapping to Data Service model

3.2.1 DataDescription portTypes

DataDescription portTypes allow a description of data, represented by a Data Service, to be provided via Service Data Elements (SDEs). No operations are defined within these interfaces. The model independent specification for these is given in the *Grid Data Service Specification* [GDSS] document. Here they are extended to provide a description of XML based Data Resources. There are two main points of extension for XML Data Resources:

- *XMLCollectionDescription*: provides information about an XML collection that a Data Service may represent.

- *XMLDocumentDescription*: provides information about a particular instance of a document that a Data Service mayrepresent. This portType will make available information about the structure representing the XML instance document as well as other relevant data.

These interfaces are described in Section 4.

3.2.2 DataAccess portTypes

DataAccess portTypes allow XML Data Resources to be modified through insertion or updates or queried through an appropriate language. When a Data Service is created the supporting *DataAccess* interfaces may be specified using WS-Agreement [WS-Agreement].

- *XMLCollectionAccess.*
- *XMLDocumentAccess.*

- *XQueryAccess.*
- *XUpdateAccess.*
- *XPathAccess.*

These are covered in more detail in Section 5.

3.2.3 DataFactory portTypes

The *DataFactory* portTypes allow data represented in XML Data Resources, usually as the result of a query or update, to be instantiated as Data Services. The specializations in this instance thus deal with the type of expression that can be passed to a *DataFactory* to expose the results. The properties and interfaces that will be supported by these Data Services will be specified using WS-Agreement [WS-Agreement]. *DataFactory* specialisations are:

- *XMLCollectionFactory.*
- *XMLDocumentFactory.*
- *XPathFactory.*
- *XQueryFactory.*

These are covered in more detail in Section 6.

3.3 Relationships with other specifications

DAIS does not propose to provide its own query/update languages for XML based Data Resources. Instead, it acts as a conduit for existing XML based query and update languages to be conveyed to the appropriate Data Resources, in this instance XML based Data Resources or a relational Data Resource that supports XML type queries. As such DAIS relies on existing XML based query and update languages. In this document, interface support is provided for languages based on the following standards:

- **XPath**: version 1.0 is a W3C recommendation defining a language for addressing parts of an XML document [XPath]. There is work in progress to define a second version of XPath that is closely aligned with XQuery.
- **XUpdate**: is a language for updating XML documents [XUpdate]. It has not been standardised by any of the main standardisation bodies and is still in working draft. Nevertheless it is supported by several of the XML DBMS products in the market hence DAIS describes interfaces for XUpdate.
- **XQueryX**: currently a W3C working draft [XQueryX] proposes an XML representation for the XQuery language. [XQuery] proposes to provide a query language for XML Data Resources. Although XQuery is not yet a standard DAIS proposes to describe interfaces for it.

The DAIS framework could be extended to encompass any new or emerging XML query/update standards.

4. DataDescription Port Types

The DataDescription portTypes allow metadata for Data Services to be made available. DataDescription portTypes are provided for use with XML collections and for XML documents .

4.1 XMLCollectionDescription

4.1.1 : Service Data Declarations

The service data elements (SDEs) described in this section are associated with a Data Resource that has been represented as an XML collection.

- *collectionStructure***:** Describes the sub-collections of an XML database collection.

```
<sd:serviceData name=”collectionStructure”
                type=”CollectionStructureType”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”true”/>
```

- *collectionSchema***:** XML Schemas associated with a collection of XML documents. The documents in the collection described by this SDE have to conform to at least one XML Schema contained in this SDE if there are any schemas present, i.e. not null.

```
<sd:serviceData name=”collectionSchema”
                type=”CollectionSchemaType”
                minOccurs=”1” maxOccurs=”unbounded”
                mutability=”mutable”
                modifiable=”false”
                nillable=”true”/>
```

- *resource*: the names of the resources in this collection, such as the names of XML documents or records stored in the collection.

```
<sd:serviceData name=”resource”
                type=”xsd:string”
                minOccurs=”1” maxOccurs=”unbounded”
                mutability=”mutable”
                modifiable=”false”
                nillable=”true”/>
```

4.2 XMLDocumentDescription

4.2.1 Service Data Declarations

Service data here works at a finer granularity, providing information about a single or set of documents in a Data Service.

- *documentNames***:** a single or set of document names that are available through the DataService.

```
<sd:serviceData name=”documentNames”
                type=”xsd:schema”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”true”/>
```

- *documentSchema***:** XML Schema that this document conforms to if the schema name is not null.

```
<sd:serviceData name=”documentSchema”
                type=”xsd:schema”
                minOccurs=”0” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”true”/>
```

5. DataAccess PortTypes

5.1 XMLCollectionAccess

5.1.1 Service Data Declarations

No SDEs are defined in addition to those inherited from DataAccess.

5.1.2 Operations

5.1.2.1 XMLCollectionAccess::createCollection

Create a new subcollection of the current collection. This creates the named collection and returns success or fail.

Input

- *Name:* Name of the new subcollection.
- *XMLCollectionDescription: the metadata description of this collection.*

Output

- *Status:* Boolean indicating whether the collection was created.

Fault(s)

- *CollectionAlreadyExists:* a collection with the given name already exists
- *Fault:* any other fault.

5.1.2.2 XMLCollectionAccess::removeCollection

Remove a subcollection of this collection.

Input

- *Name:* the name the collection to be removed.

Output

- *Status:* Boolean indicating whether the collection could be removed.

Fault(s)

- *NoSuchCollection:* the collection could not be found.

5.1.2.3 XMLCollectionAccess::createResources

Create a new (empty) resource in this collection.

Input

- *Names:* the names of the new resources.

Output

- *Status:* Boolean indicating whether the resources could be created.

Fault(s)

- *ResourceAlreadyExists.*

5.1.2.4 XMLCollectionAccess::removeResources

Remove a resource from this collection.

Input

- *Names:* names of the resources to be removed.

Output

- *Status:* Boolean indicating whether the resources could be removed.

Fault(s)

- *NoSuchResource.*

5.1.2.5 XMLCollectionAccess::addSchema

Add an XML schema to this collection.

Input

- Schema: XML Schema document.

Output

- Status: Boolean indicating whether the description could be added.

Faults(s)

- *SchemaAlreadyExists.*

5.1.2.6 XMLCollectionAccess::removeSchema

Remove an XML schema from this collection.

Input

- XML Schema URI.

Output

- Status: Boolean indicating whether the description could be removed.

Faults(s)

- *SchemaDoesNotExist*

5.2 XPathAccess

This portType facilitates synchronous XPath queries across an XML resource or a collection of resources. The response document will hold the results of the query.

5.2.1 Service Data Declarations

- *xPathVersion:* The XPath version that is supported

```
<sd:serviceData name=”xPathVersion”
                type=”xsd:string”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

- *querySchema:* The XML schema for the query parameters

```
<sd:serviceData name=”querySchema”
                type=”xsd:schema”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

5.2.2 Operations

XPathAccess::query

Query an XML resource or a collection of resources.

Input

- *Expression:* An XPath request wrapped in XML, including namespaces.

```
<xsd:complexType name="XPathExpressionType">
    <xsd:element name="expression" type="xsd:string"/>
    <xsd:element name="collection" type="xsd:string" minOccurs="0"/>
    <xsd:element name="resourceId" type="xsd:string" minOccurs="0"/>
</xsd:complexType>
```

Output

- *Response:* An XML document, the response to the request.

Fault(s)

- *InvalidExpression:* The query is not a valid XPath expression.
- *Fault:* Any other fault.

5.3 XUpdateAccess

A service implementing XUpdateAccess will typically be associated to one or more XML resources (or a collection of XML resources) and allows updating of the resources using XUpdate.

5.3.1 Service Data Declarations

- *xUpdateVersion:* The version of XUpdate that is supported (current version based on the working draft document is 1.0)

```
<sd:serviceData name=”xUpdateVersion”
                type=”xsd:string”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

- *requestSchema:* The XML Schema for the update parameters

```
<sd:serviceData name=”requestSchema”
                type=”xsd:schema”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

5.3.2 Operations

5.3.2.1 XUpdateAccess::update

Update an XML resource using XUpdate.

Input

- *Expression:* An XUpdate request

Output

- *Response*

Fault(s)

- *InvalidStatement*

5.4 XQueryAccess

This portType provides an interface for XQuery requests across a collection of XML resources.

5.4.1 Service Data Declarations

- *xQueryVersion:* The XQuery version that is supported

```
<sd:serviceData name=”xQueryVersion”
                type=”xsd:string”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

- *querySchema:* The XML schema for the query parameters (e.g. XQueryX)

```
<sd:serviceData name=”querySchema”
                type=”xsd:schema”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

5.4.2 Operations

5.4.2.1 XQueryAccess::execute

Input

- *request:* An XML document containing the XQuery request. The request must conform to the XML schema as defined in the *querySchema*.

Output

- *results:* The results of the request.

Fault(s)

6. DataFactory PortTypes

6.1 XMLCollectionFactory

6.1.1 Service Data Declarations

No SDEs are defined aside from those inherited from DataFactory. However, the XMLCollectionFactory portType defines the following initial set of service data value elements.

```
<sd:staticServiceDataValues>
  <dais:proposedAgreement>
    <dais:XMLCollectionFactoryAgreement>
       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
             <dais:readable gsa:Negotiability=”gsa:Fixed”.../>
             <dais:updateable gsa:Negotiability=”gsa:Fixed” .../>
       <wsp:OneOrMore/>

       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
         <dais:supportsInterface dais:qname=”DataDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataManagement”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLDocumentDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLCollectionDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XUpdateAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLCollectionAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLCollectionFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLDocumentFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryXFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
       <wsp:OneOrMore/>
     </dais:XMLCollectionFactoryAgreement>
  </dais:proposedAgreement>
</sd:staticServiceDataValues>

<xsd:complexType name="XMLCollectionFactoryAgreementType">
```

```
  <xsd:complexContent>
    <xsd:extension base="dais:DAISBaseAgreementType">
      <xsd:sequence>
      <!-- name of collection to be represented as a new Data Service -->
      <xsd:element name="collectionName" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name=”XMLCollectionFactoryAgreement”
             type=”XMLCollectionFactoryAgreementType”/>
```

The createServiceExtensibility SDE of the OGSI Factory portType and the supportedAgreement SDE of the WS-Agreement AgreementFactory portType should be populated with the information required to support the use of WS-Agreement documents of type XMLCollectionFactoryAgreementType.

6.1.2 Operations

The *createService* operation inherited from the Factory portType can be used to create a subcollection and associate a Data Service with it, i.e. services implementing any of the Data Factory, Data Description or Data Access interfaces. For example, createService could create a new collection with a Data Service that implements the XMLCollectionAccess, XPathAccess and XPathFactory interfaces. The interfaces that the factory can create are published as SDEs. The client specifies the creation of the required interfaces in the parameters to createService using WS-Agreement.

6.1.2.1 XMLCollectionFactory::createService

Create a new Data Service handle, which represents an XML collection. The arguments are the same as those defined in the OGSI specification for the Factory portType. WS-Agreement terms are passed in the *CreationParameters*.

mario: if agreement part is not in the CreationParameters does this throw a fault or are services created with default interfaces? Simon thought fault … is this changing the behaviour of createService though?

6.2 XMLDocumentFactory

6.2.1 Service Data Declarations

No SDEs are defined in addition to those inherited from DataFactory. However, the XMLDocumentFactory portType defines the following initial set of service data value elements.

```
<sd:staticServiceDataValues>
  <dais:proposedAgreement>
    <dais:XMLDocumentFactoryAgreement>
       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
              <dais:readable gsa:Negotiability=”gsa:Fixed”.../>
              <dais:updateable gsa:Negotiability=”gsa:Fixed” .../>
       <wsp:OneOrMore/>

       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
         <dais:supportsInterface dais:qname=”DataDescription”
                                  sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataAccess”
                                  sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataFactory”
                                  sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLDocumentDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLCollectionDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XUpdateAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>


         <dais:supportsInterface dais:qname=”XMLDocumentFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryXFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
       <wsp:OneOrMore/>
    </dais:XMLDocumentFactoryAgreement>
  </dais:proposedAgreement>
</sd:staticServiceDataValues>

<xsd:complexType name="XMLDocumentFactoryAgreementType">
  <xsd:complexContent>
    <xsd:extension base="dais:DAISBaseAgreementType">
      <xsd:sequence>
      <!-- name of document to be represented as a new Data Service -->
      <!—- or the name of the new document to be created            -->
      <xsd:element name="documentName" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name=”XMLDocumentFactoryAgreement”
             type=”XMLDocumentFactoryAgreementType”/>
```

The createServiceExtensibility SDE of the OGSI Factory portType and the supportedAgreement SDE of the WS-Agreement AgreementFactory portType should be populated with the information required to support the use of WS-Agreement documents of type XMLDocumentFactoryAgreementType.

6.2.2 Operations

6.2.2.1 XMLDocumentFactory::createService

Create a new XML document in a collection as well the Data Service representing an existing XML document in a collection.

6.3 XPathFactory

This portType allows the result of XPath queries across one or more XML resources to be represented as a Data Service.

6.3.1 Service Data Declarations

TODO – These SDEs are a repetition of the SDEs found in the XpathAccess portType. We should consider how to define just one set.

- *xPathVersion:* The XPath version that is supported.

```
<sd:serviceData name=”xPathVersion”
                type=”xsd:string”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

- *querySchema:* The XML schema for the query parameters.

```
<sd:serviceData name=”querySchema”
                type=”xsd:schema”
                minOccurs=”1” maxOccurs=”1”
                mutability=”mutable”
                modifiable=”false”
                nillable=”false”/>
```

The XPathFactory portType also defines the following initial set of service data value elements.

```
<sd:staticServiceDataValues>
  <dais:proposedAgreement>
    <dais:XPathFactoryAgreement>
       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
             <dais:readable gsa:Negotiability=”gsa:Fixed”.../>
             <dais:updateable gsa:Negotiability=”gsa:Fixed” .../>
       <wsp:OneOrMore/>

       <wsp:OneOrMore wsp:Usage=”wsp:Required”>
         <dais:supportsInterface dais:qname=”DataDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”DataManagement”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLDocumentDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XMLCollectionDescription”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XUpdateAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryAccess”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XPathFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
         <dais:supportsInterface dais:qname=”XQueryXFactory”
                                 sa:Negotiability=”gsa:Fixed”.../>
       <wsp:OneOrMore/>
    </dais:XPathFactoryAgreement>
  </dais:proposedAgreement>
</sd:staticServiceDataValues>

<xsd:complexType name="XPathFactoryAgreementType">
  <xsd:complexContent>
    <xsd:extension base="dais:DAISBaseAgreementType">
      <xsd:sequence>
      <xsd:element name=”xpathExpression” type=”XPathExpressionType”>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="XPathExpressionType">
    <xsd:element name="expression" type="xsd:string"/>
    <xsd:element name="collection" type="xsd:string" minOccurs="0"/>
    <xsd:element name="resourceId" type="xsd:string" minOccurs="0"/>
</xsd:complexType>

<xsd:element name=”XPathFactoryAgreement”
             type=”XPathFactoryAgreementType”/>
```

The createServiceExtensibility SDE of the OGSI Factory portType and the supportedAgreement SDE of the WS-Agreement AgreementFactory portType should be populated with the information required to support the use of WS-Agreement documents of type XPathFactoryAgreementType.

6.3.2 Operations

6.3.2.1 XPathFactory::createService

Create a new Data Service instance, which represents the results of an XPath query. An agreement document holding an XPath request and the interfaces to be created is passed to Factory::createService. The factory will create a Data Service holding the result of the request.

6.4 XQueryXFactory

6.4.1 Service Data Declarations

To be determined based on completion of XQueryX 1.0 specification.

6.4.2 Operations

6.4.2.1 XQueryXFactory::createService.

Create a new Data Service that corresponds to the results of an XQueryX request.

6.4.3 Operations

To be determined based on completion of XQueryX 1.0 specification.

7. Security Considerations

The XML Realization of a Grid Data Service will use standard Grid Security mechanisms as specified by OGSA Security working group combined with standard ways of relating Grid credentials and authorities to resource access rights. The assumption is that these standards will also indicate how to make information related to authentication, authorization security etc available

8. Conclusion

This document has discussed a specialization of the portTypes defined in the *Grid Data Service Services* [GDSS] document providing the additional capabilities required to address XML based Data Resources. This is a work in progress and feedback is welcomed on this document.

Editor Information

Mario Antonioletti,
EPCC,
University of Edinburgh,
James Clerk Maxwell Building,
Mayfield Road,
Edinburgh EH9 3JZ,
United Kingdom.

Shannon Hastings,
Ohio State University,
333 W. Tenth Ave.,
Columbus OH, 43210,
USA.

Amy Krause,
EPCC,
University of Edinburgh,
James Clerk Maxwell Building,
Mayfield Road,
Edinburgh EH9 3JZ,
United Kingdom.

Stephen Langella,
Ohio State Unversity,
333 W. Tenth Ave.,
Columbus OH, 43210,
USA.

Simon Laws,
IBM United Kingdom Limited,
Hursley Park,
Winchester,
Hampshire, SO21 2JN,
United Kingdom.

Susan Malaika,
IBM Corporation,
Silicon Valley Laboratory,
555 Bailey Avenue,
San Jose, CA 95141,
USA.

Norman W. Paton,
Department of Computer Science,
University of Manchester,
Oxford Road,
Manchester M13 9PL,
United Kingdom.

Contributors

Malcolm Atkinson, NESC.
Dave Pearson, Oracle.
Greg Riccardi, Florida State University.

Acknowledgements

The DAIS Working Group of the Global Grid Forum is active, and many people have contributed to discussions within the group in recent months, including but not limited to: Bill Allcock, Vijay Dialani, Dieter Gawlick, Allen Luniewski, Sastry Malladi, Inderpal Narang, Steve Tuecke , Jay Unger, Paul Watson and Martin Westhead.

Intellectual Property Statement

The GGF 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 GGF Secretariat.

The GGF 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 GGF Executive Director.

Full Copyright Notice

References

[Data Services]

I.Foster, S.Tuecke, J.Unger. *OGSA Data Services*, August 14, 2003. See: https://forge.gridforum.org/docman2/ViewProperties.php?group_id=42&document_content_id=733.

[GDSS]

M. Antonioletti, M. Atkinson, S. Malaika, S. Laws, N. W. Paton D. Pearson and G. Riccardi. *Grid Data Service Specification*. DAIS-WG Informational Draft, 9th Global Grid Forum, 19th September 2003.

[OGSI]

S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, D. Snelling, P. Vanderpilt, Open Grid Services Infrastructure, Version 1.0, http://www.gridforum.org/ogsi-wg, March 13, 2003.

[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-Agreement]

K.Czajkowski, A.Dan, J.Rofrano, S.Tuecke, M.Xu. *Agreement-based Grid Service Management*, Version 0, June 12, 2003.

[XPath]

J. Clark and S. DeRose. *XML Path Language (XPath)*, Version 1.0
W3C Recommendation 16 November 1999. See: http://www.w3.org/TR/xpath.

[XQuery]

S. Boag, D. Chamberlin, M. F. Fernández, D. Florescu, J. Robie and J. Siméon. *XQuery 1.0: An XML Query Language*, W3C Working. See: http://www.w3.org/TR/xquery/.

[XQueryX]

A. Malhotra, J. Robie and M. Rys. *XML Syntax for XQuery 1.0 (XQueryX).* W3C Working, See: http://www.w3.org/TR/xqueryx.

[XUpdate]

A. Laux and L. Martin. *XUpdate Working Draft*, last release September 14, 2000. See: http://www.xmldb.org/xupdate/xupdate-wd.html.