GFD-R.074
DAIS Working Group

Mario Antonioletti, University of Edinburgh
Malcolm Atkinson, NESC
Amy Krause, University of Edinburgh
Simon Laws, IBM
Susan Malaika, IBM
Norman W Paton, University of Manchester
Dave Pearson, Oracle
Greg Riccardi, Florida State University
20 July 2006

Web Services Data Access and Integration – The Core (WS-DAI) Specification, Version 1.0

Status of This Memo

This memo provides information regarding the specification of service based interfaces to data resources. Distribution is unlimited.

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 GGF (Global Grid Forum) Open Grid Services Architecture (OGSA), and its associated specifications, defines consistent interfaces through web services to components of a 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 generic data interfaces developed by the Database Access and Integration Services (DAIS) Working Group that can be extended to support specific kinds of data resources, such as relational databases, XML repositories, object databases, or files. Related specifications define how specific data resources and systems can be described and manipulated through such extensions. The specifications can be applied in regular web services environments or as part of a grid fabric.

Contents

1. Introduction

Data access plays a central role for many types of grid applications. Data access generally involves the retrieval, insertion or modification of data, which may be available from a variety of infrastructures and in a range of formats. Data access in grids requires a flexible framework for handling data requests to a data resource that is to be integrated within a grid fabric as defined by the Open Grid Services Architecture (OGSA) [OGSA] of the Global Grid Forum (GGF).

This document specifies a collection of generic data access interfaces that are made available as web services. The interfaces described here are grouped into the following functional categories:

- Data description: the provision of information about a service or the data resource accessed by way of the service.
- Data access: the provision of access to data through a service interface.
- Data factory: the provision of indirect access to data through a client specified interface.

The operations that make up the data access and data factory interfaces often use established query languages to specify what data is to be retrieved, inserted or modified.

This specification provides a pattern for data service interfaces and the properties that describe or modify the behavior of these interfaces. The pattern described here can be extended in realizations to define interfaces to access particular types of data, as is done in the proposals to access relational [WS-DAIR] and XML [WS-DAIX] representations of data. Future specifications for accessing other specialized forms of data, for example, files or object databases, should also extend the core set of interfaces defined in this document. The term *WS-DAI specifications* is used in this document to refer to WS-DAI and its associated realizations.

1.1 Specification Scope

This document specifies a data service in terms of the core data access and data factory interfaces and core data description properties that a data service may implement.

This specification does not define new query languages or data models. Data access interfaces are therefore described in terms of existing language interfaces supported by the underlying data resource to which the service is providing access.

1.2 Specification Organization

This specification separates the abstract model of a data service from its operational representation. The abstract model is described using the terminology defined in Section 3, and employs the concepts elaborated upon in Section 4. Section 5 presents the data service properties and messages that form the core of this specification. Section 6 describes the role of WSRF in the representation of data resources. A mapping of the abstract model to the web services description language (WSDL) is described in Section 7. Section 8 discusses security. Section 9 draws some conclusions. The WSDL and XML Schema included in the appendices should be taken as the normative interface and property descriptions of this specification.

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].

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/>).

Italicized element names are used when an element is intended to be specified by the specification realizations.

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 XML Schema is provided in Appendix A. 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]. The terms “data access service”, “data resource” and “data set” are used as described 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
xsd	http://www.w3.org/2001/XMLSchema
wsdai	http://www.ggf.org/namespaces/2005/12/WS-DAI
wsa	http://www.w3.org/2005/08/addressing

3. Terminology

3.1 Data Access Service

In this specification a data access service is taken to mean a web service that implements one or more of the WS-DAI specified interfaces to provide access to data resources. A data access service is a kind of data service, where the latter may additionally include services for managing data resources or for data movement. The specifications provide a web service based data access framework, exposing existing data access techniques already available in a data resource or through the use of other relevant specifications as required. Many similar data resources use different query languages and data representations. In addition, different data resources, through their management system, may offer different capabilities. It is not the role of this specification to specify a uniform interface, semantics and functionality hiding these variations. Instead WS-DAI and its specializations expose the heterogeneity so that clients may be able to exploit it. We envisage that in due course higher level standards will emerge that hide this heterogeneity where this is required, or that the WS-DAI specifications will be used to provide web service access to data integration systems.

3.2 Data Resource

A data resource represents any system that can act as a source or sink of data. Examples of data resources include relational or XML databases, file systems and sensor networks.

The expectation is that data resources in a grid will still generally be managed using existing systems such as relational database management systems or file systems. In this case, an existing system will already provide consumers with mechanisms for accessing data resources,

and it is the responsibility of the existing system to manage the lifetime of a data resource. Such data resources are called *externally managed data resources*.

There will be situations where data is created in the context of a WS-DAI data access service alone, and the lifetime of that data resource, and any data it contains, is managed by the data access service. Such data resources are called *service managed data resources*.

3.3 WSRF Data Resource

A WSRF data resource provides a representation of a data resource using WS-Resource [WS-Resource]. WSRF data resources exist within the context of WS-DAI data access services, through which they are accessed. The WSRF data resource is an OPTIONAL component of a WS-DAI service.

3.4 Data Resource Abstract Name

For the purposes of the WS-DAI specifications a data resource abstract name is defined as a unique and persistent name for a data resource suitable for machine processing that does not necessarily contain any location information. An abstract name takes the form of a URI [URI].

3.5 Data Resource Address

A data resource address is a name that specifies the location of a data resource as accessed through a data access service. The address takes the form of a web service end point with enough information to distinguish the data resource at that end point.

3.6 Resource List

A resource list contains the abstract names and addresses of all the data resources known to and accessible through a data access service. The list is made available through a set of WS-DAI defined messages. The resource list is OPTIONAL.

3.7 Consumer

A consumer, in terms of a data resource, is an artifact that exploits the interface provided by a data access service in order to access a data resource.

3.8 Data Set

A data set is an encoding of data suitable for externalization outside a data access service, for example, as an XML document. The concept of a data set is introduced to describe data as it appears in the messages passing to and from data access services, i.e. between the consumer and the data access service.

4. Concepts

4.1 Data Service Model

The focus of this specification is on defining core data access service interfaces. WS-DAI considers interfaces for two main kinds of data resource:

Externally managed data resource: An externally managed data resource:

1. Normally has an existence outside the service.
2. Has its lifetime managed in ways that are not specified in the WS-DAI specifications.

An example of an externally managed data resource is a relational database established by an observatory to hold astronomy data that is subsequently made available through WS-DAI interfaces.

Service managed data resource: A service managed data resource:

1. Does not normally have an existence outside the service-oriented middleware.
2. Has its lifetime managed in ways that are specified in this WS-DAI specification.

The relationships between the principal WS-DAI constructs are illustrated in Figure 1; note that this is a conceptual view, and does not represent architectural features.

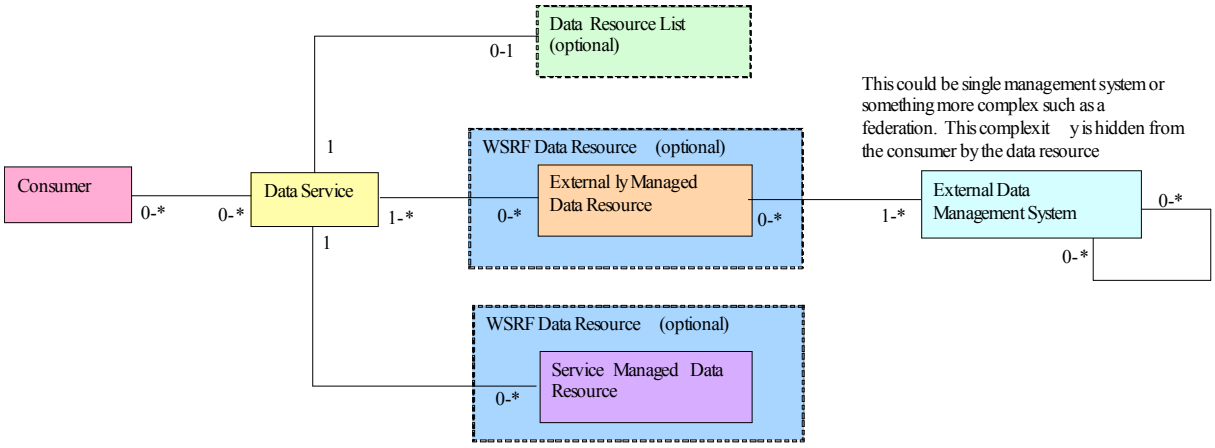

Figure 1: The relationships between WS-DAI constructs

An example of a service managed resource is a query result RowSet which is preserved for subsequent access.

Two main categories of interface information, that apply to both kinds of data resource, are described in this specification:

- Properties of the data resources being accessed and of the service by which access is being provided.
- Messages for accessing data resources. Every message MUST contain the abstract name of the data resource that it is targeting.

The above properties and messages are supported by all realizations, and may be augmented in realizations with:

- Properties specific to the kind of data resource being accessed.
- Messages specific to the kind of data resource being accessed.

The following OPTIONAL resource type, called a WSRF data resource, defines a generic WS-Resource whose purpose is to:

- Provide messages supporting soft state control of the lifetime of an individual data resource.
- Provide messages for querying and updating the properties of an individual data resource.

When present, the WSRF data resource provides a unifying facade over both externally and service managed data resources. A WSRF data resource, when present, has a one-to-one correspondence with an externally or service managed data resource, and the WS-ResourceProperties messages MUST be supported for accessing the properties of the data access service.

A data access service may provide a data resource list interface that lists the names and addresses of data resources known to and accessible through the data access service. Such a service also supports:

- A message for retrieving this data resource list.
- A message for translating from a data resource abstract name to a data resource address. This address MUST be an EPR.

The resource list provides a mechanism for a consumer to find out which data resources are know to and available to a data access service, and provides a mechanism for mapping between the name of a data resource and its resource address.

A data access service ultimately presents a consumer with an interface to a data resource. A data resource can have arbitrary complexity, for example, a federation of relational databases. A consumer is not typically exposed to this complexity, and operates within the bounds and semantics of the interface provided by the data access service.

A data access service implementation:

- MAY include a resource list.
- MAY include externally managed data resources (with or without an associated WSRF data resource).
- MAY include service managed data resources (with or without associated WSRF data resources).

If a data access service implementation supports a WSRF data resource for any one data resource, then it MUST support:

- A WSRF data resource for each individual data resource.
- A protocol for translating from a data resource abstract name to a data resource address.

A data access service implementation that does not support a WSRF data resource is not dependent on WSRF [WS-Resource].

The WS-DAI specifications include operations for creating service managed resources through data factory messages.

4.2 Interface

The word interface refers to the collections of messages that a service offers its clients. In the WS-DAI specifications, interfaces are often associated with properties that provide information about the service. The word interface is not intended to refer specifically to the proposed use of the word interface found in the candidate recommendation of the WSDL 2.0 specifications although this may be an appropriate mapping in the future. The messages defined in the WS-DAI specifications are collected into a number of portTypes in the associated WSDL; these portTypes may be composed in different ways to provide services.

4.3 Interface Composition

This specification does not mandate how interfaces are composed into data access 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, as illustrated in Figure 2.

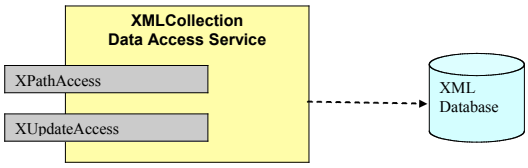

Figure 2: An example of interface composition

Here, a data access service provides XPathAccess and XUpdateAccess interfaces for an XMLCollection data access service that, in this case, is associated with an XML Database.

4.4 Naming

The OGSA naming scheme is a three-tiered system that allows arbitrary web and grid resources to be identified in a global context. These resources are identified through hierarchically arranged human-oriented names, globally unique abstract names or low-level addresses.

For the purposes of this specification, the data resource abstract name is a URI [URI].

```
<xsd:simpleType name="DataResourceAbstractNameType">
  <xsd:restriction base="xsd:anyURI" />
</xsd:simpleType>

<xsd:element name="DataResourceAbstractName"
             type="wsdai:DataResourceAbstractNameType"/>
```

The data resource address is a WS-Addressing [WS-Addressing] end point reference (EPR).

When a data resource address is returned by a WS-DAI data service, for example, in the case of factory messages, the EPR ReferenceParameter element MUST contain the DataResorceAbstractName element that identifies the data resource to which the address refers.

When passing a message to a WS-DAI data resource the consumer MUST provide the information required to discriminate the data resource at that address. An example of the information required to discriminate a data resource is the data resource abstract name passed in with WS-DAI messages. When a WSRF data resource is used, the EPR of the WSRF data resource itself identifies the data resource using reference parameters. The abstract name MUST still appear in WS-DAI messages but MAY be ignored by the implementation.

The structure and use of human-oriented names, as they may be used to identify data resources, is not considered by this specification.

A property, through which the data resource abstract name can be obtained, is provided as part of the core data description.

4.5 Properties

Properties describe the characteristics of a data resource as well as the data access service's relationship with that data resource. Properties are represented by XML elements. They SHOULD be made available through the data access service associated with a data resource.

Properties MAY be readable through the data access service interface.

> This specification describes messages that return the entire property document associated with a data resource. The WS-ResourceProperties specificaiton [WS-ResourcePropeties] describes messages for performing queries on individual data resource properties that are supported if a WSRF Resource is implemented.

Properties MAY be settable through the data access service interface.

> WSRF [WS-ResourceProperties] describes messages for changing the values of data resource properties.

Properties may be required to be set when establishing a data access service / data resource relationship.

> WS-DAI factory messages allows a configuration document to be passed which provides initial values for the data resources configuration properties. If no configuration document is passed then default values are assumed as described in the `ConfigurationMap` for the factory message in question, as described in Section 5.1.6.

The value of each property MAY depend on the credentials of the consumer.

The structure and valid states of any particular property are dependent on the data access service and the data resource that the property describes.

Properties are associated with each interface in the WS-DAI specifications. They are collected in the data description section for each interface type.

Configuration properties MAY be set when the data access service / data resource relationship is established and MAY be passed in as part of the message exchanges that implement the factory pattern. These properties appear in each data description section under the configurable data description heading.

4.6 Direct Data Access

In this specification “direct data access” means that a consumer receives a direct response, containing the requested data, following a request made to a data access service. For example, passing an XPathQuery message to a data access service will result in a response message containing a set of XML fragments – this is considered to be direct data access.

An operation that directly inserts, updates or deletes data through a data access service is also involved in direct data access. For example, passing a SQL insert statement to a data access service will result in a response message indicating how many tuples have been inserted.

4.7 Indirect Data Access

In this specification “indirect data access” means that a consumer does not receive the results in the response to a request made to a data access service. The request to access data will be processed by the data access service and data resource, with the results being made available to the consumer indirectly as a new data resource, often through a different data access service that may support a different set of interfaces. The type and behavior of the new data resource are determined by the data access service and the configuration parameters passed in with the original request.

For example, passing a SQL query message to a data access service in this mode can result in a reference to another data service, and data resource, being produced that allows access to the result of the original query via a RowSet interface.

Holding intermediate results at the service side can minimize unnecessary data movement.

The indirect data access model can also be used when data is being inserted, updated or deleted. For example, with a data access service representing a directory in a file system, indirect data access could be used to represent a new file into which new data can be inserted. However, it is not appropriate to apply the indirect model of operation in all situations. For example, in the case of a relational database, indirect data access does not naturally model an empty table into

which data is to be added, as SQL statements are traditionally sent to the database data resource and not to a table data resource.

4.8 Subscription Based Data Access

The WS-DAI specifications do not consider subscription based data access, where the consumer supplies a profile describing the data of interest and the conditions under which it will be delivered. Work is ongoing in the GGF Information Dissemination working group to specify this model.

4.9 Lifetime

When a data resource is removed from a data access service, via its corresponding WSRF data resource using the WS-ResourceLifetime messages [WS-ResourceLifetime] or via the DestroyDataResource message, the underlying data ceases to be accessible to the consumer via the data access service. This does not necessarily require that the underlying data be removed. The semantics of the data resource lifetime with respect to the lifetime of the underlying data depend on whether the data resource is externally or internally managed, as indicated by the `DataResourceManagement` property described in Section 5.1.3.

For externally managed data resources the lifetime of the data resource has no effect on the lifetime of the data management system or the data it contains. As the data is managed by an external management system any data will remain until the external management system decides to remove it. For example, a relational data management system's databases will not be removed when the data resources that represent them are removed from a data access service.

For service managed data resources the lifetime of the data resource is related directly to the lifetime of the data itself. When the data resource, or its corresponding WSRF data resource, is removed the data SHOULD also be removed. For example, the RowSet resulting from a SQL query should be removed when the consumer has finished with it. This can be achieved by removing the WSRF data resource that provides access to the RowSet.

4.10 Sessions

The WS-DAI specifications do not describe how multiple requests to a data access service are correlated either for single or multiple consumers, or for single or multiple requests. This is left to other proposed web service specifications, for example, WS Coordination [WS-Coordination] or WS Context [WS-Context].

4.11 Access Control

The possibility of many client processes accessing a data access service interface, possibly concurrently, is assumed to be the default situation. The `ConcurrentAccess` property, described in Section 5.1.4, indicates whether the service is able to successfully process messages concurrently. When a service that is only able to process messages sequentially receives messages concurrently then it MUST return a ServiceBusyFault.

Access to a data access service and the data it represents may be granted or denied, where appropriate, using suitable access mechanisms and interfaces either at the service or the underlying data resource. In particular, a data access service implementation is responsible for mapping grid level credentials to credentials that are acceptable to the underlying data resource. WS-DAI does not specify these mechanisms or interfaces.

Where new data resources are generated during indirect access requests they SHOULD adopt the access control policies of the parent data resource.

The requirement to pass and control security related information is common with many other specifications. It is anticipated that this requirement will be satisfied using other specifications such as WS Security [WS-Security]. Agreement protocols such as those proposed by WS-Agreement [WS-Agreement] could in future be supported to allow client and service to negotiate the access control environment.

4.12 Operation Validity

The WS-DAI specifications describe messages and properties in accordance with the type of interface being presented. The appearance of an operation in a portType does not guarantee that it may be called validly in any particular situation. Faults are provided to notify the caller that an operation could not be completed successfully.

4.13 Faults

This core specification defines general fault patterns. The message patterns and properties included in this document imply that any message in a WS-DAI realization SHOULD implement a minimum set of faults:

ServiceBusyFault	The service is already processing a message and concurrent operations are not supported.
NotAuthorizedFault	The consumer is not authorized to perform the requested operation or is not authorized to perform the requested operation at this time.
InvalidResourceNameFault	The data resource specified is unknown to the service.
DataResourceUnavailableFault	The data resource that is the target of the message is currently not available. This could be caused by a temporary fault or could indicate that the data resource has stopped operating permanently.
InvalidExpressionFault	The expression given as part of the data access request contains errors.
InvalidLanguageFault	The input dataset (usually the expression component of an incoming request) has an unrecognized Language element. Languages are discussed further in Section 5.1.7.

For messages implementing direct data access further faults SHOULD be implemented:

InvalidDatasetFormatFault	The DatasetFomatURI specified is not in the collection defined by the DatasetMap property.

For messages implementing indirect data access further faults SHOULD be implemented:

InvalidPortTypeQNameFault	The PortTypeQNname specified is not in the collection defined by ConfigurationMap property.
InvalidConfigurationDocumentFault	The ConfigurationDocument specified is not valid according to the ConfigurationDocumentQName when the

ConfigurationMap is indexed by the specified PortTypeQName.

5. Core

The core interface describes those properties and messages required to access data resources. These properties and messages MUST be supported by all realizations. Where no WSRF data resource is present the entire properties document may be retrieved using the realization specific messages defined for this purpose. Where a WSRF data resource is present, the WS-ResourceProperties messages MUST also be supported.

5.1 Static Data Description

The data description contains XML structures that describe the properties of a data resource. These properties are static in as much that they cannot be set by the consumer. Hence, properties described here are not required to be set as part of a factory pattern. These properties MUST all appear in a data access service that implements the data description. Realizations based on this specification can extend the list of properties as required.

The mapping section describes how these elements are made available in WSDL.

5.1.1 DataResourceAbstractName

```
<xsd:simpleType name="DataResourceAbstractNameType">
  <xsd:restriction base="xsd:anyURI" />
</xsd:simpleType>

<xsd:element name="DataResourceAbstractName"
             type="wsdai:DataResourceAbstractNameType"/>
```

/wsdai:DataResourceAbstractName

The abstract name associated with the data resource(s) as represented by the data access service.

5.1.2 ParentDataResource

```
<xsd:complexType name="DataResourceAddressType">
  <xsd:complexContent>
    <xsd:extension base="wsa:EndpointReferenceType"/>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name="ParentDataResource" type="wsdai:DataResourceAddressType"
             minOccurs="0" maxOccurs="1"/>
```

/wsdai:ParentDataResource

If this data resource is a service managed data resource this property SHOULD hold the address of the data resource from which it was generated. If the data resource is an externally managed data resource this property MUST be omitted.

5.1.3 DataResourceManagement

```
<xsd:element name="DataResourceManagement">
  <xsd:simpleType>
    <xsd:restriction base="xsd:token">
      <xsd:enumeration value="ExternallyManaged"/>
      <xsd:enumeration value="ServiceManaged"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

/wsdai:DataResourceManagement
An enumeration indicating the type of data resource for lifetime management purposes. It takes the values:

ExternallyManaged	The data resource is managed by an external data management system. The lifetime of the data is not related to the lifetime of the data resource and cannot be controlled through the data access service interface.
ServiceManaged	The data resource is managed by the data access service. The lifetime of the data within the data resource is directly related to the lifetime of the data resource which in turn is controlled through the data access service interface.

5.1.4 ConcurrentAccess

```
<xsd:element name=”ConcurrentAccess” type=”xsd:boolean” />
```

/wsdai:ConcurrentAccess
Has the value true if a data access service is able to process more than one message concurrently otherwise it has the value false. If a service is unable to support concurrent operations a ServiceBusyFault will result from a message submitted while another is already being processed.

5.1.5 DatasetMap

```
<xsd:element name="DatasetMap" minOccurs=”0” maxOccurs=”unbounded”>
  <xsd:complexType name="DatasetMapType" >
    <xsd:sequence>
      <xsd:element name="MessageQName" type=”xsd:QName”/>
      <xsd:element name="DatasetFormatURI" type=”xsd:anyURI”/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

/wsdai:DatasetMap
A mapping between the QName of a message and the URI of a dataset format. For direct access operations the dataset format refers to the format of the dataset returned. Multiple instances of this property provide a complete map for a data resource.

/wsdai:DatasetMap/wsdai:MessageQName
The QName of a message.

/wsdai:DatasetMap/wsdai:DatasetFormatURI
The URI of a dataset format that this message can return. Initially this is intended to be a URL that can be used to retrieve an XML Schema but also allows for other types of URIs to be used.

When messages are defined with flexible result types, following the message pattern in Section 5.3.1, the result types supported MUST be identified using DatasetMap.

5.1.6 ConfigurationMap

```
<xsd:element name="ConfigurationMap" minOccurs=”0” maxOccurs=”unbounded”>
  <xsd:complexType name="ConfigurationMapType" >
    <xsd:sequence>
      <xsd:element name="MessageQName" type=”xsd:QName”/>
      <xsd:element name="PortTypeQName" type=”xsd:QName”/>
```

```
      <xsd:element name="wsdai:ConfigurationDocumentQName" type=”xsd:QName”
                   minOccurs="0"/>
       <xsd:element name="DefaultConfigurationDocument">
         <xsd:complexType mixed="true">
           <xsd:sequence>
             <xsd:element ref="wsdai:ConfigurationDocument"/>
           </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

/wsdai:ConfigurationMap
: A mapping between the QName of a factory message and the QName of a port type that can be used to access the data resource resulting from a factory message. The QName of the configuration document required to initialize the new data resource is also included in the map. The DefaultConfigurationDocument describes default properties in the case that no configuration document is provided with a factory message. Multiple instances of this property provide the complete map for a data resource.

/wsdai:ConfigurationMap/wsdai:MessageQName
: The QName of a message.

/wsdai:ConfigurationMap/wsdai:PortTypeQName
: The QName of a port type that is to be used to access data resources resulting from the factory message.

/wsdai:ConfigurationMap/wsdai:ConfigurationDocumentQName
: The QName of the XML schema of a configuration document that MAY be provided when using the factory message with the expectation of using the associated port type.

/wsdai:ConfigurationMap/wsdai:DefaultConfigurationDocument
: The configuration document that will be used if no configuration document is provided with the factory message.

When the factory message pattern is supported, as described in Section 5.4.1, an associated ConfigurationMap MUST be provided.

5.1.7 LanguageMap

```
<xsd:element name="LanguageMap" minOccurs=”0” maxOccurs=”unbounded”>
  <xsd:complexType name="LanguageMapType" >
    <xsd:sequence>
      <xsd:element name="MessageQName" type=”xsd:QName”/>
      <xsd:element name="LanguageURI" type=”xsd:anyURI”/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

/wsdai:LanguageMap
: A mapping between the QName of a message and the URI of an expression language. For example, a message that supports SQL queries may allow such queries as SQL-1999 expressions or SQL-2003 expressions.

/wsdai:DatasetMap/wsdai:MessageQName
: The QName of a message.

/wsdai:DatasetMap/wsdai:LanguageURI

The URI of a language that controls the format of the expression in the message identified by MessageQName. These URIs are not defined by WS-DAI specifications, and initially will be defined by the service implementers. When messages are defined, as in Section 5.3.1, that include language statements as parameters, the language(s) supported MUST be described using LanguageMap.

5.1.8 PropertyDocument

```
<xsd:element name=”PropertyDocument”>
  <xsd:complexType name="PropertyDocumentType">
    <xsd:sequence>
      <xsd:element ref="wsdai:DataResourceAbstractName" />
      <xsd:element ref="wsdai:DataResourceManagement" />
      <xsd:element ref="wsdai:ParentDataResource" minOccurs="0"
                                                  maxOccurs="1"/>
      <xsd:element ref="wsdai:DatasetMap" minOccurs="0"
                                          maxOccurs="unbounded"/>
      <xsd:element ref="wsdai:ConfigurationMap" minOccurs="0"
                                                maxOccurs="unbounded"/>
      <xsd:element ref="wsdai:LanguageMap" minOccurs="0"
                                           maxOccurs="unbounded"/>
      <xsd:element ref="wsdai:DataResourceDescription" />
      <xsd:element ref="wsdai:Readable" />
      <xsd:element ref="wsdai:Writeable" />
      <xsd:element ref="wsdai:ConcurrentAccess" />
      <xsd:element ref="wsdai:TransactionInitiation" />
      <xsd:element ref="wsdai:TransactionIsolation" />
      <xsd:element ref="wsdai:ChildSensitiveToParent" />
      <xsd:element ref="wsdai:ParentSensitiveToChild" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

/wsdai:PropertyDocument

All of the properties described in the static and configurable data description sections collected under a single structure. This structure is returned by the GetDataResourcePropertyDocument operation. See Section 5.3.2.

5.2 Configurable Data Description

Properties defined here MUST appear in all data access services that implement data description. They SHOULD appear in configuration documents passed to factory operations as they describe data access service and data resource behavior.

5.2.1 DataResourceDescription

```
<xsd:element name="DataResourceDescription" >
    <xsd:complexType mixed="true">
      <xsd:sequence>
        <xsd:any minOccurs="0" maxOccurs="unbounded" processContent=”lax”/>
      </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

/wsdai:DataResourceDescription

A free format description of the data resource represented by a data access service, to provide a human-readable description.

5.2.2 Readable

```
<xsd:element name=”Readable” type=”xsd:boolean” />
```

/wsdai:Readable
: Has the value true if a data access service is able to return data in response to query operations otherwise it has the value false.

5.2.3 Writeable

```
<xsd:element name=”Writeable” type=”xsd:boolean” />
```

/wsdai:Writeable
: Has the value true if a data access service is able to update data represented by the data access service in response to update, insert or delete operations. Otherwise it has the value false.

5.2.4 TransactionInitiation

```
<xsd:element name="TransactionInitiation">
    <xsd:simpleType>
      <xsd:restriction base="xsd:token">
         <xsd:enumeration value="NotSupported"/>
         <xsd:enumeration value="Automatic"/>
         <xsd:enumeration value="Manual"/>
      </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

/wsdai:TransactionInitiation
: Describes under what circumstances a transaction is initiated in response to messages. It takes the values:

NotSupported	Does not support Transactions.
Automatic	An atomic transaction is initiated for each message.
Manual	Transaction context under control of the consumer. WS-DAI does not define interfaces for controlling transactions manually. It is expected that other specifications, such as WS-AtomicTransaction [WS-AtomicTransaction], will be used alongside the WS-DAI specifications.

5.2.5 TransactionIsolation

```
<xsd:element name="TransactionIsolation">
    <xsd:simpleType>
      <xsd:restriction base="xsd:token">
           <xsd:enumeration value="NotSupported"/>
           <xsd:enumeration value="ReadUncommitted"/>
           <xsd:enumeration value="ReadCommitted"/>
           <xsd:enumeration value="RepeatableRead"/>
           <xsd:enumeration value="Serialisable"/>
      </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

/wsdai:TransactionIsolation
: Describes how transactions behave with respect to other ongoing transactions on the same data resource. It takes one of the values:

NotSupported	Does not support transactions.
ReadUncommitted	Accesses uncommitted changes made by other transactions.
ReadCommitted	Accesses only committed changes made by other

transactions.

RepeatableRead	Accesses only committed changes made by other transactions and ensures that no records read during the transaction are changed by other transactions.
Serialisable	Accesses only committed changes made by other transactions, ensures that no records read during the transaction are changed by other transactions and ensures that result sets read during the transaction are not extended by other transactions.

5.2.6 ChildSensitiveToParent

```
<xsd:element name="ChildSensitiveToParent">
    <xsd:simpleType>
      <xsd:restriction base="xsd:token">
           <xsd:enumeration value="Insensitive"/>
           <xsd:enumeration value="Sensitive"/>
      </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

/wsdai:ChildSensitiveToParent

If data resource B is created from data resource A then this property describes the sensitivity to change of data resource B with respect to changes in data resource A. It takes the values:

Insensitive	Changes to the parent data resource do not affect the data presented by this data access service/data resource.
Sensitive	Changes to the parent data resource may be reflected in this data access service/data resource. The property ParentDataResource gives the name of the parent data resource

For example, when reading forwards and backwards in a RowSet, “Insensitive” means that you will read the same results when you read forwards and then backwards regardless of any changes in the data resource that created the RowSet. “Sensitive” means that any changes in the data resource that created the RowSet will be observed.

5.2.7 ParentSensitiveToChild

This is the converse of the previous property whereby changes in the derived data resource are reflected in the content of the parent data resource.

```
<xsd:element name="ParentSensitiveToChild">
    <xsd:simpleType>
      <xsd:restriction base="xsd:token">
           <xsd:enumeration value="Insensitive"/>
           <xsd:enumeration value="Sensitive"/>
      </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

/wsdai:ParentSensitiveToChild

If data resource B is created from data resource A then this property describes the sensitivity to change of data resource A with respect to changes in data resource B. It takes the values:

Insensitive	Changes to the derived data resource do not affect the data presented in the parent data resource.

Sensitive	Changes to the derived data resource may be reflected in the parent data resource as specified in the ParentDataResource property.

5.2.8 Example PropertyDocument

This is a non-normative example of a PropertyDocument.

```
<wsdai:PropertyDocument xmlns:...>

  <wsdai:DataResourceAbstractName>urn:dais:ds2</wsdai:DataResourceAbstractName>
  <wsdai:DataResourceManagement>ExternallyManaged</wsdai:DataResourceManagement>
  <wsdai:ParentDataResource>
    <wsa:Address>http://www.ggf.org/services/daisservice</wsa:Address>
    <wsa:ReferenceParameters>
      <DataResourceAbstractName>urn:dais:ds1</DataResourceAbstractName>
    </wsa:ReferenceParameters>
  </wsdai:ParentDataResource>
  <wsdai:DatasetMap>
    <wsdai:MessageQName>myService:MessageQName1</wsdai:MessageQName>
    <wsdai:DatasetFormatURI>
      http://www.ggf.org/dataforma1
    </wsdai:DatasetFormatURI>
  </wsdai:DatasetMap>
  <wsdai:DatasetMap>
    <wsdai:MessageQName>myService:MessageQName2</wsdai:MessageQName>
    <wsdai:DatasetFormatURI>
      http://www.ggf.org/dataforma2
    </wsdai:DatasetFormatURI>
  </wsdai:DatasetMap>
  <wsdai:ConfigurationMap>
    <wsdai:MessageQName>MessageQName</wsdai:MessageQName>
    <wsdai:PortTypeQName>PortTypeQName</wsdai:PortTypeQName>
    <wsdai:ConfigurationDocumentQName>
       myService:ConfigurationDocumentQName
    </wsdai:ConfigurationDocumentQName>
    <DefaultConfigurationDocument>
      <wsdai:ConfigurationDocument>
        <wsdai:DataResourceDescription>
         A description of the data resource would go here.
        </wsdai:DataResourceDescription>
        <wsdai:Readable>true</wsdai:Readable>
        <wsdai:Writeable>true</wsdai:Writeable>
        <wsdai:TransactionInitiation>NotSupported</wsdai:TransactionInitiation>
        <wsdai:TransactionIsolation>NotSupported</wsdai:TransactionIsolation>
        <wsdai:ChildSensitiveToParent>Insensitive</wsdai:ChildSensitiveToParent>
       <wsdai:ParentSensitiveToChild>Insensitive</wsdai:ParentSensitiveToChild>
      </wsdai:ConfigurationDocument>
    </DefaultConfigurationDocument>
  </wsdai:ConfigurationMap>
  <wsdai:LanguageMap>
    <wsdai:MessageQName>myService:MessageQName1</wsdai:MessageQName>
    <wsdai:LanguageURI>http://www.ggf.com/querylangugage1</wsdai:LanguageURI>
  </wsdai:LanguageMap>
  <wsdai:DataResourceDescription/>
  <wsdai:Readable>true</wsdai:Readable>
  <wsdai:Writeable>true</wsdai:Writeable>
  <wsdai:ConcurrentAccess>true</wsdai:ConcurrentAccess>
  <wsdai:TransactionInitiation>NotSupported</wsdai:TransactionInitiation>
  <wsdai:TransactionIsolation>NotSupported</wsdai:TransactionIsolation>
  <wsdai:ParentSensitiveChild>Insensitive</wsdai:ParentSensitiveChild>
  <wsdai:ChildSensitiveParent>Insensitive</wsdai:ChildSensitiveParent>
</wsdai:PropertyDocument>
```

5.3 Core Data Access Messages

Data access collects together messages that directly access and modify the data represented by a data access service along with the properties that describe the behavior of these access messages, as illustrated in Figure 3.

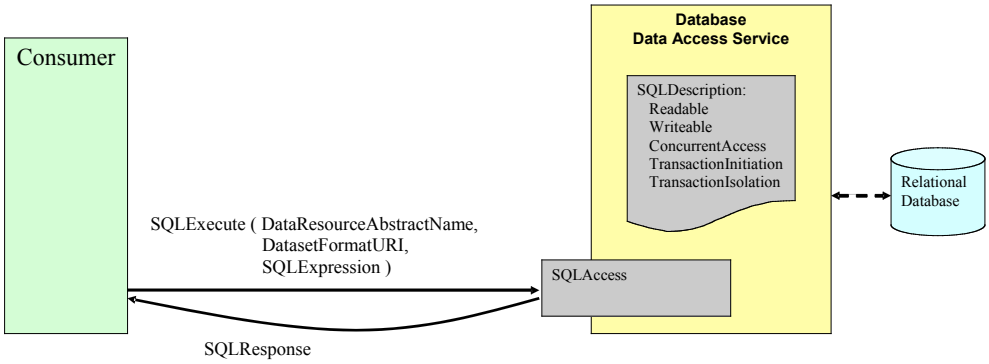

Figure 3: Data access example

The database data access service, in the diagram above, implements the SQLAccess messages and exposes the SQLDescription properties; more details about these properties can be found in [WS-DAIR]. In this example, a consumer uses the SQLExecute message to submit a SQL expression. The associated response message will contain the results of the SQL execute request. When the SQL expression used is a SELECT statement, the SQL response will contain a RowSet.

The behavior of the database data access service during data access is controlled, in part, by its data description properties. These properties include the properties defined in this specification. For example, “Writeable” appears in the SQLDescription of this example and will be set to “true” if the database data access service is able to process messages that update the data resource.

All data description properties defined in this specification SHOULD appear in all data access services. Realizations based on this specification MAY extend the list of properties and messages as required and MUST define one or more access messages.

5.3.1 Message Patterns

WS-DAI data access interfaces support messages that allow data sets to be passed into or retrieved from a data access service, and describe the messages that provide direct data access from a data access service.

The structure of a direct data access request message XML instance is:

```
<RequestMessage>
      <wsdai:DataResourceAbstractName/>
      <wsdai:DatasetFormatURI/>?
      <RequestDocument/>
</RequestMessage>
```

/*RequestMessage*

> This is the root element for a request message. The type of this element is specific to each message.

/*RequestMessage*/wsdai:DataResourceAbstractName

The abstract name of the data resource to which the message is directed.

/*RequestMessage*/wsdai:DatasetFormatURI

An OPTIONAL element that MAY be used to define the format of the response dataset data. This element, when present, MUST contain a URI from the set that appears as DatasetMap property elements. When this element is omitted the format of the response message will follow the format referenced by the first DatasetMap property.

/*RequestMessage*/*RequestDocument*

This element contains the request expression.

The request document itself has a structure which is specific to the realization that defines it and the expression language being used. The document MAY contain a URI indicating the language used to form the expression. The URI of this attribute, if present, MUST be one of those specified by the LanguageMap property. Where no language attribute is provided the first LanguageMap entry for the request message is assumed. The structure of a request document is:

```
<RequestDocument Language="a uri">
  The realization specific expression
</RequestDocument>
```

The structure of a direct access response message is:

```
<ResponseMessage>
  <wsdai:Dataset>
    <wsdai:DatasetFormat/>
    <wsdai:DatasetData>
       Data goes here formatted according to the DatasetFormat uri in
       the request message.
    </wsdai:DatasetData>
  </wsdai:Dataset>
</ResponseMessage>
```

The structure of the response message is determined by the DatasetFormat element in the request message. This element contains the URI of a data format supported by the data access service. Valid response data formats are exposed by the data access service using the DatasetMap property.

Realizations MAY choose to define interfaces containing statically typed response messages. Realizations MAY choose to return metadata associated with the results of a direct access response message dataset by extending the Dataset element.

5.3.2 CoreDataAccess::GetDataResourcePropertyDocument

Returns the core property document values associated with the service implementing this message.

Input

- GetDataResourcePropertyDocumentRequest
  - DataResourceAbstractName – the abstract name of the resource to which the message is directed.

Output

- GetDataResourcePropertyDocumentResponse
  - PropertyDocument – the properties described in the data description section. See Section 5.1.8.

Faults
- InvalidResourceName – the supplied 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 ConcurrentAccess is false.

5.3.3 CoreDataAccess::DestroyDataResource

Destroy the named data resource; future messages directed at the resource MUST yield an InvalidResourceNameFault. The semantics of data resource destruction depends on whether the data resource is externally managed (DataResourceManagement has the value ExternallyManaged) or internally managed (DataResourceManagement has the value InternallyManaged).

When an internally managed data resource is destroyed all of its associated data MAY be destroyed, and the resources used MAY be reclaimed. The internal behavior depends upon the data access service implementation but from the consumers perspective the data resource is no longer available following this operation.

When an externally managed data resource is destroyed none of its data is destroyed (as it is managed by an eternal data management system) but the knowledge of this data resource is removed from the service to which the DestroyDataResource message is directed. From the consumers perspective the data resource is no longer available following this operation

Input
- DestroyDataResourceRequest
  - DataResourceAbstractName – abstract name of the resource to which the message is directed.

Output
- DestroyDataResourceResponse
  - No useful content is contained in this response. The operation will succeed or a fault will be thrown.

Faults
- InvalidResourceName – the supplied 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 ConcurrentAccess is false.

5.3.4 CoreDataAccess::GenericQuery

A general message for passing query documents to a data resource. The URIs of valid GenericExpression languages are provided using the LanguageMap property described in Section 5.1.7.

Input
- GenericQueryRequest
  - DataResourceAbstractName – abstract name of the resource to which the message is directed.
  - DatasetFormatURI? – An OPTIONAL element that MAY be used to define the format of the response dataset data. This element, when present, MUST contain a URI from the set that appears as DatasetMap property elements described in Section 5.1.5.

When this element is omitted the format of the response message will follow the format referenced by the first DatasetMap property.

- *GenericExpression* – the query expression document. The document MAY contain URI attribute indicating the language used to form the expression. The URI of this attribute, if present, MUST be one of those specified by the LanguageMap property described in Section 5.1.7. Where no language attribute is provided the first LanguageMap entry for GenericQuery is assumed.

Output

- GenericQueryResponse – the response document formatted according to DatasetFormatURI with the data in a DatasetData element.

Faults

- InvalidResourceNameFault – the supplied resource name is not known to the service.
- DataResourceUnavailableFault – the specified data resource is unavailable.
- InvalidDatasetFormatFault – the supplied dataset format is not known to the service.
- InvalidExpressionFault – the supplied expression is not of a form known to the service.
- InvalidLanguageFault – the supplied expression language is not known to the service.
- NotAuthorizedFault – the consumer is not authorized to perform this operation at this time.
- ServiceBusyFault – the service is already processing a request and ConcurrentAccess is false.

5.4 Core Data Factory Messages

This section specifies the message patterns that are to be used by any WS-DAI realization to implement the factory pattern. Such messages create a new relationship between a data resource and a data access service. In this way, a data resource may be used to represent the results of a query or act as a place holder for data to be inserted. A data factory describes properties that dictate how a data access service must behave on receiving factory messages.

The factory pattern MAY involve the creation of a new data resource. The factory pattern MAY involve the deployment of a web service.

The factory pattern allows a new relationship between a data access service and a data resource to be established as the consequence of a message exchange with a data access service. This ability to derive one data resource from another, or to provide alternative views of the same data resources, leads to a collection of notionally related data resources, as illustrated in Figure 4.

The database data access service in this example presents a SQLFactory interface. The SQLExecuteFactory operation is used to construct the SQL response data access service. This service provides access to the RowSet resulting from a SQL expression against the Relational Database, assuming that the expression contains a SELECT statement. The RowSet could be stored as a table in a relational database or decoupled from the database. The RowSet is represented as a collection of rows via a data access service that does not implement the SQLAccess portType. Instead the SQL response data access service presents the SQLResponseAccess collection of operations that allows the RowSet to be retrieved but does not provide facilities for submitting SQL expressions.

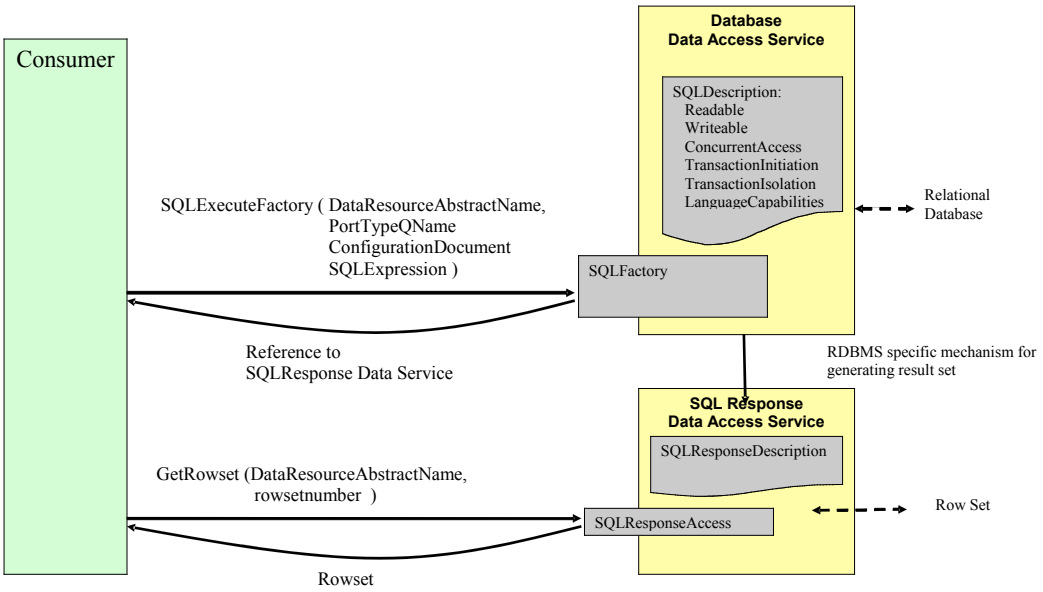

Figure 4: Data factory example

5.4.1 Message Patterns

The structure of a message implementing the factory pattern is:

```
<RequestMessage>
      <wsdai:DataResourceAbstractName/>
      <wsdai:PortTypeQName/>?
      <wsdai:ConfigurationDocument/>?
      <wsdai:PreferredTargetService/>?
      <RequestDocument/>
</RequestMessage>
```

/*RequestMessage*

This is the root element for a request message. The type of this element is specific to each message.

/*RequestMessage*/wsdai:DataResourceAbstractName

The abstract name of the resource to which the message is directed.

/*RequestMessage*/wsdai:PortTypeQName

The OPTIONAL QName of the port type that the resulting resource will be accessed through. If no PortTypeQName is specified the port type specified by the first ConfigurationMap property, described in Section 5.1.6, is assumed.

/*RequestMessage*/wsdai:ConfigurationDocument

The OPTIONAL initial values for the properties of the new data resource. These are provided by the consumer. The type of this XML document is specific to the request message and, in particular, the type of data resource that is expected to result from the processing of this message. Valid types are advertised by the ConfigurationMap property elements, as described in Section 5.1.6. If no ConfigurationDocument element is

provided or elements are missing from the provided ConfigurationDocument then values from the ConfigurationMap/DefaultConfigurationDocument are used as defaults.

/*RequestMessage*/wsdai:PreferredTargetService

The OPTIONAL end point reference of the service in which the new data resource should be instantiated. This is a hint to the implementation and MAY be ignored if the EPR is invalid, the referenced service does not support the required port type or for other implementation specific reasons.

/*RequestMessage*/*RequestDocument*

This element contains the request expression.

The request document itself has a structure which is specific to the WS-DAI realization that defines it and the expression language being used. The document may contain an OPTIONAL URI indicating the language used to form the expression. The URI of this attribute, if present, MUST be one of those specified by the LanguageMap property. Where no language attribute is provided the first LanguageMap entry for the request message is assumed.

The structure of a request document is:

```
<RequestDocument Language="a uri">
  The realization specific expression
</RequestDocument>
```

This specification does not adopt a mechanism for negotiating the initial values of a data resource's properties. It is expected that these will be defined in other specifications, for example, WS Agreement [WS-Agreement].

Configuration properties are not universally applicable and will make sense only in the context of a data access service implementing a particular interface. For example, it does not make sense to discuss the forward or backward nature of an iterator for a service that only accepts the SQLExecute message. The document containing configuration properties MUST also specify a type of interface that allows access to the result from the factory message.

A valid set of configuration property document schemas MUST be advertised using the ConfigurationMap property described in Section 5.1.6.

The structure of the factory pattern response message is:

```
<wsdai:DataResourceAddressList>
  <wsa:EndPointReference>
    <wsa:ReferenceParameters>
      <wsdai:DataResourceAbstractName/>?
    </wsa:ReferenceParameters>
  </wsa:EndPointReference>+
</wsdai:DataResourceAddressList>
```

/wsdai:DataResourceAddressList

A list containing oneor more WS-Addressing End Point References.

/wsdai:DataResourceAddressList/wsa:EndPointReference

This is the root element of a WS-Addressing End Point Reference.

/wsdai:DataResourceAddressList/wsa:EndPointReference /wsa:ReferenceParameters

The parameters associated with the End Point Reference.

/wsdai:DataResourceAddressList/wsa:EndPointReference /wsa:ReferenceParameters /wsdai:DataResourceAbstractName

The abstract name of the newly created resource.

5.5 Core Resource List Messages

The resource list is an OPTIONAL part of data access. It defines a message for retrieving a list of names and addresses of data resources accessible via the data access service implementing the message. It also provides a message which translates between a data resource name and the address of a data access service that is able to provide access to that named data resource.

5.5.1 CoreResourceList::GetDataResourceList

Return a list of abstract names and addresses of data resources that are accessible via the data access service that implements the message.

Input
- GetDataResourceListRequest

Output
- GetDataResourceListResponse
  - DataResourceAddress* – a structure containing the data resource abstract name and the data resource address.

Faults
- NotAuthorizedFault – the consumer is not authorized to perform this operation at this time.
- ServiceBusyFault – the service is already processing a request and ConcurrentAccess is false.

5.5.2 CoreResourceList::Resolve

Translate a data resource abstract name into a list of data resource addresses.

Input
- ResolveRequest
  - DataResourceAbstractName – the abstract name of the data resource.

Output
- ResolveResponse
  - DataResourceAddress+ – this structure MUST represent the data resource address and MAY contain the data resource abstract name.

Faults
- InvalidResourceNameFault – the supplied resource name is not known to the service.
- NotAuthorizedFault – the consumer is not authorized to perform this operation at this time.
- ServiceBusyFault – the service is already processing a request and ConcurrentAccess is false.

6. WSRF Data Resource

The WSRF data resource describes those properties and messages required to access the properties of and control the lifetime of data resources.

6.1 Data Description

6.1.1 DataResourceProperties

```
<xsd:element name="DataResourceProperties">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="wsdai:PropertyDocumentType">
        <xsd:sequence>
          <xsd:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
```

/wsdai:DataResourceProperties
: A structure that describes the data resource that the WSRF data resource refers to.

/wsdai:DataResourceProperties/*
: The core set of properties for the data resource inherited from the PropertyDocument (see Section 5.1.8) plus any further information that describes the data resource, such as the properties defined by a realization.

6.2 Data Access Messages

Messages supported by the WSRF Data Resource are defined in the WS-ResourceProperties [WS-ResourceProperties] and WS-ResourceLifetime [WS-ResourceLifetime] specifications.

6.3 Data Factory Messages

No data factory messages are defined by the WSRF data resource.

7. Mapping the Abstract Model to WDSL

7.1 Related Specifications

The WS-DAI WSDL makes use of the following specifications:

- The specifications referenced by WS-I Basic Profile 1.0 [WS-I]
  - SOAP 1.1 [SOAP]
  - WSDL 1.1 [WSDL]
- WS-Addressing 1.0 [WS-Addressing]
- WS-Resource 1.2 [WS-Resource]
- WS-ResourceProperties 1.2 [WS-ResourceProperties]
- WS-ResourceLifetime 1.2 [WS-ResourceLifetime]
- WS-Resource 1.2 [WS-Resource]

The use of WSDL 1.1 forces the manual aggregation of messages and properties from the WS-DAI core specification and from WS-DAI realizations into the port type for each data access service.

The Web Services Resource Framework (WSRF) is a set of web services specifications that describe a WS-Resource construct as a means of expressing the relationship between stateful resources and web services. The reader is referred to the WSRF documentation for more information, for example “The WS-Resource Framework” [WS-Resource]. These specifications have been standardized by the OASIS WSRF technical committee.

7.1.1 Mapping the Model

The WS-DAI model is mapped to a set of WSDL defined messages in the usual way. WSRF is employed in mapping the WSRF data resource to WSDL thus providing full access to data resource properties and allowing the lifetime of a data resource to be controlled.

For data access services that do not implement the OPTIONAL WSRF data resource the services rely on the DataResourceAbstractName passed with each WS-DAI message in order to identify the target data resource.

Regardless of the approach adopted the destruction of a data resource adopts the semantics described for the wsdai:DestroyDataResource message, as described in Section 5.3.3.

The text of the specification documents does not provide a complete description of the WSDL and XML Schema contained in the appendices (for example, changes to faults associated with WSRF data resources are not discussed in the text). As a result, the WSDL and XML Schema included in the appendices should be taken as the definitive interface descriptions.

7.1.2 WSRF Data Resource

WS-Addressing End Point References (EPR) are adopted and the resource access pattern [WS-Resource] is used to identify each WSRF data resource within a data access service.

The properties of each WSRF data resource are accessed using the approach described by the WS Resource Properties specification [WS-ResourceProperties].

The lifetime of the WSRF data resource is controlled using the messages and properties described in the WS Resource Lifetime specification [WS-ResourceLifetime].

Data resource properties may be accessed using the WS Resource Properties messages as well as via messages described by each specialization.

7.1.3 Data Resource (Without WSRF)

When a WSRF data resource is used, data resources are identified using the information contained in the EPR of the WSRF data resource. The DataResourceAbstractName passed in each WS-DAI message simply duplicates this information and MAY be ignored by an implementation.

When no WSRF data resource is used the DataResourceAbstractName passed with each WS-DAI message provides the information required to discriminate a data resource at a service end point.

Each WS-DAI specialization describes the set of messages required to access the data resource in question. This includes a message that returns the properties document for each data resource. The full functionality of WS Resource Properties, including property query and update functionality, is not supported by WS-DAI unless the WSRF data resource interfaces are being used.

The lifetime of the data resource is controlled using DestroyDataResource. Soft state lifetime management of data resources is not supported by WS-DAI unless the WSRF data resource interfaces are being used.

7.1.4 Port Type Aggregation

The core specification defines a single generic access message. Data access messages defined in the realizations will appear in WSDL 1.1 definitions as operations on the port type of the data access service. Due to the restrictions of WSDL 1.1, messages must be composed from the

WSDL in the specification into the WSDL for the specific data access service by the service implementer.

Properties from realization specifications are grouped together into an XML schema type which is particular to the data access service being developed. An element of this type is returned by a specific message for the interface in question.

Further grouping of properties into configuration documents allows them to be passed easily with any factory messages.

8. Security Considerations

The realizations of a 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.

9. Conclusions

This document describes a collection of top level interfaces for access to data resources as services, which are extended in companion documents to provide support for multiple data storage paradigms. The interfaces proposed are intended to be compatible with the architecture to be proposed by the GGF Open Grid Services Architecture working group.

Editor Information

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

Malcolm Atkinson,
e-Science Institute,
15 South College Street,
Edinburgh EH8 9AA,
UK.

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

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,
School of Computer Science,
University of Manchester,
Oxford Road,
Manchester M13 9PL,
United Kingdom.

Dave Pearson,
Oracle Corporation Ltd,
Thames Valley Park,
Reading,
Berkshire RG6 1RA,
United Kingdom.

Greg Riccardi,
Department of Computer Science,
Florida State University,
Tallahassee, FL 32306-4530,
USA.

Contributors

Vijay Dialani, IBM.
Steven Lynden, University of Manchester.
Allen Luniewski, IBM.
Inderpal Narang, IBM.
Savas Parastatidis, University of Newcastle upon Tyne.
Steve Tuecke, Univa.
Jay Unger, IBM.

Acknowledgements

The DAIS Working Group of the Global Grid Forum has benefited from many people contributing to discussions within the group, including but not limited to: Bill Allcock, Miguel Esteban Gutierrez, Dieter Gawlick, Shannon Hastings, Stephen Langella, Sastry Malladi, Paul Watson and Martin Westhead.

Intellectual Property Statement

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.

Full Copyright Notice

References

[OGSA]
I. Foster (Ed), H. Kishimoto (Ed), A. Savva (Ed), D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, R. Subramaniam, J. Treadwell, J. Von Reich. *The Open Grid Services Architecture, Version 1.0.* Global Grid Forum. GFD-I.030. 29 January 2005. http://www.ggf.org/documents/GFD.30.pdf.

[OGSA Glossary]
J. Treadwell, *Open Grid Services Architecture Glossary of Terms*, GFD-I.044, January 25th 2005. http://www.ggf.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.

[SOAP]
D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, D. Winer. *Simple Object Access Protocol (SOAP) 1.1*. W3C Note 08 May 2000. Available from: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[WS-Addressing]
D. Box and F. Cubera (Ed). *Web Services Addressing 1.0 - Core*. W3C Proposed Recommendation, 21 March 2006. http://www.w3.org/TR/2006/PR-ws-addr-core-20060321.

[WS-Agreement]
A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke and M. Xu. *Web Services Agreement Specification (WS-Agreement).*

[WS-Context]
D. Bunting, M. Chapman, O. Hurley, M. Little, J. Mischkinsky, E. Newcomer, J. Webber, K. Swenson. *Web Services Context (WS-Context)* , http://www.oasis-open.org/committees/download.php/4344/WSCTX.pdf.

[WS-Coordination]
F. Cabrera, G. Copeland, T. Freund, J. Klein, D. Langworthy, D. Orchard, J. Schwchuk and T. Storey, *Web Services Coordination (WS-Coordination)*, http://www-106.ibm.com/developerworks/library/ws-coor/, 2002-a.

[WS-DAIR]
M. Antonioletti, B. Collins, A. Krause, S. Malaika, J. Magowan, S. Laws, N. W. Paton. *Web Services Data Access and Integration – The Relational Realization (WS-DAIR) Specification Version 1.0*, Global Grid Forum. 2006.

[WS-DAIX]
M. Antonioletti, A. Krause, S. Hastings, S. Langella, S. Malaika, S. Laws, N. W. Paton. *Web Service Data Access and Integration – The XML Realization (WS-DAIX) Specification Version 1.0.* Global Grid Forum. 2006.

[WSDL]
E. Christensen, F. Curbera, G. Meredith and S. Weerawarana. *Web Services Description Language 1.1*. World Wide Web Consortium. W3C Note 15 March 2001. http://www.w3.org/TR/wsdl.

[WS-I]
K. Ballinger, D. Ehnebuske, M. Gudgin, M. Nottingham, P. Yendluri. *Basic Profile 1.0*. 16 April 2004. Available from: http://www.ws-i.org/Profiles/BasicProfile-1.0.html.

[WS-Resource]
S. Graham (Ed), A. Karmarkar (Ed), J. Mischkinsky (Ed), I. Robinson (Ed), I. Sedukhin (Ed). Web Services Resource 1.2 (WS-Resource), Version 1.2, OASIS Standard, 1 April 2006. http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf.

[WS-ResourceProperties]
S. Graham (Ed), J. Treadwell (Ed). Web Services Resource Properties 1.2 (WS-ResourceProperties), Version 1.2, OASIS Standard, 1 April 2006. http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf.

[WS-ResourceLifetime]
L. Srinivasan (Ed), T. Banks (Ed). Web Services Resource Lifetime 1.2 (WS-ResourceLifetime), Version 1.2, OASIS Standard, 1 April 2006. http://docs.oasis-open.org/wsrf/wsrf-ws_resource_lifetime-1.2-spec-os.pdf.

[WS-Security]
OASIS Web Services Security 1.0 (WS-Security 2004) standard as of April 6th 2004, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

[WS-AtomicTransaction]
F. Cabrera, et. al., *Web Services Transaction (WS-AtomicTransaction)*, V1.0 August 2005, ftp://www6.software.ibm.com/software/developer/library/WS-AtomicTransaction.pdf.

[URI]

T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax*, January 2005, http://www.ietf.org/rfc/rfc3986.txt.