Scenarios for Mapping DAIS Concepts

<u>Status of This Memo</u>

This memo provides information to the Grid community regarding the mapping of Data Service concepts, as defined by the GGF DAIS Working Group, to various infrastructure patterns. It does not define any standards or technical recommendations. Distribution is unlimited.

Abstract

The specifications produced by the Database Access and Integration Services (DAIS) working group to date describe mappings onto the Open Grid Services Infrastructure (OGSI) v1.0. The group of specifications collectively referred as the Web Services Resource Framework (WSRF) has recently been proposed as a replacement for OGSI. Furthermore, the Web Services Grid Application Framework (WS-GAF) is a proposal of patterns that utilise existing Web Services technologies for building Grid applications. WS-GAF also considers the emerging WS-Context specification. In light of these the DAIS working group is considering which mappings should be presented in future versions of its specifications. This document documents the result of some scenario based investigations into a number of possible mappings. It is intended to inform the reader with a view to revising on the working group's specifications.

This document is presented for discussion within the Global Grid Forum (GGF) Database Access and Integration Services (DAIS) Working Group, with a view to informing the ongoing specification activity.

Contents

List Of Figures

1. Introduction

The Database Access and Integration Services (DAIS) [1] working group is defining the concepts and design patterns necessary for working with data in a Grid environment. Until recently, the Open Grid Services Infrastructure (OGSI) v1.0 [2], which defined fundamental concepts and characteristics for Grid Services, was considered as the infrastructure of choice for DAIS.

In August 2003, a group of authors from the University of Newcastle and Arjuna presented a "Grid Application Framework based on Web Service Specifications and Practices" (WS-GAF) [3]. This proposes an infrastructure for grid services based on the specifications that are part of the Web Services Interoperability (WS-I) Profile 1.1a [4] and WS-Context [5]. In more recent work WS-GAF proposes the use of URNs [6] for the identification of resources.

In January 2004, a group of authors from IBM and the Globus Alliance announced the Web Services Resource Framework (WS-RF) [7] as "the convergence of Web and Grid Services" and the evolution of OGSI. Many of the features that OGSI introduced, like Grid Service Instances, Grid Service Handles and Service Data Elements were reworked, so addressing many of the concerns expressed in [3] and the Web Services community in general.

Given that there are different possible mappings of the DAIS concepts to underlying infrastructure solutions, the DAIS working group decided to investigate the alternatives and identify the issues that may arise in a number of representative cases. This paper documents the finding of this investigation.

2. Goals of the Investigation

The investigation into the potential choices of infrastructure for mapping the DAIS concepts aims to identify possible issues that relate to:

1. The nature of the interactions with data services (the message formats)

2. The representation of data resources through service interfaces (realisations)

3. Accessing data resources from

    a. Different portTypes offered by the same service

    b. portTypes offered by different services

4. Providing names for data resources

5. Creation/generation of data resources through interactions (factory patterns)

6. Access to metadata about data resources and/or data services

7. Stateful interactions with data resources (e.g. interacting with a result set)

8. Stateful interactions with data services (e.g. setting a logging context)

9. Locating data resources (registries)

10. Third-party delivery of data resources

Other issues that are important but are not considered by this investigation:

1. Orthogonal functionality (security, policies, transactions, coordination, reliable messaging, orchestration, etc.)

2. Notification

3. Representation of data resources outside the boundaries of services (data formats)

3. Approach

The investigation concentrates on a number of scenarios in which consumers and services interact:

1. Stateful interactions with data resources

2. Sessions with data services

3. Discovery of data resources

4. Access to metadata about data resources and/or services

5. Third-party delivery

Three different underlying infrastructures are assumed for mapping the DAIS concepts:

- Web Services-Interoperability (WS-I) only specifications

- WS-I plus WS-Context

- WS-I plus WS-RF

The way in which each of the above infrastructures can support each of the scenarios is examined and the differences or commonalities are identified.

4. Terminology

Data Resource

> An identifiable target or source of data, e.g., a DB2 or Oracle database, a result set, a file in a file system. This document assumes a service oriented architecture (SOA) where data resources are exposed to interested parties through data services.

Data Service

> A web service providing an interface for access to a Data Resource

Dataset

> Formatted data external to the Data Resource, e.g. WebRowSet that is the result of a query

Context

Context in the general sense, as opposed to WS-Context, is any information required by a web service in order to determine the scope in which to consider a message exchange, e.g. the formal parameters, information passed in the SOAP header, information appended to the URL of the service, information held by the service and not passed in.

"what" – Context which identifies a Data Resource

"how" – Context describing how operations should be performed against a Data Resource

Contextualisation refers to the process of providing a Context for a message.

Consumer

The caller of a web service. In the interactions illustrated in the diagrams in this document a consumer could be another web service or some other application that is able to call the web service.

5. Scenarios

5.1. Scenario 1 - Stateful interactions with data resources

A consumer discovers a data service from a registry (out of scope). This service provides access to some data resource of interest.

The consumer submits a query to the service.

1. The result (dataset) may be returned as part of the response.

2. The result may be kept at the service, as a data resource, and only a name of the data resource is returned. The name can be the actual name of the data resource or, most likely, a logical name.

In case 2, the consumer may wish to correlate subsequent messages to the service with the identified data resource (e.g, a sort operation). The three approaches considered are:

- Use of the resource name (the resource name is carried as part of the message's body)

- Contextualisation using WS-Context (using a context managed outside the consumer and data service)

- Using WSRF

5.1.1. Contextualisation using WS-I stack

Logical names (e.g., URIs, URNs, or other application-domain specific identifiers) are used to identify resources outside the boundaries of a service. The messages are contextualised (correlated) through the inclusion of logical resource names in the body of each message.
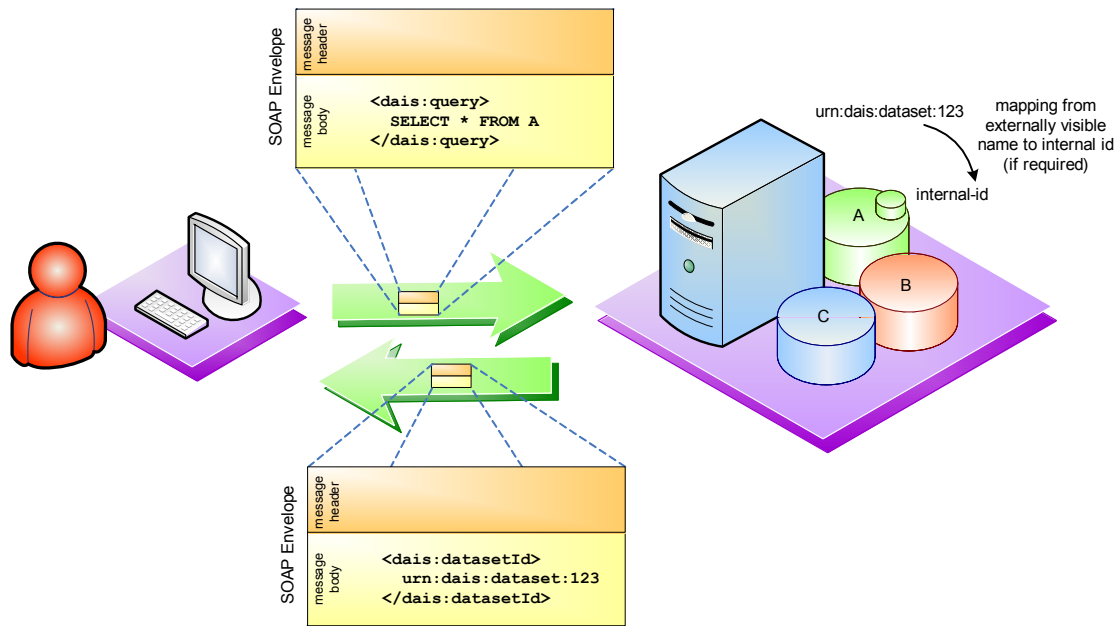
Figure 1: A query is sent to a Web Service and the (logical) name of a data resource is returned

In Figure 1, a query is sent to a data service. The query results in a data resource being generated. A name of the data resource is returned to the consumer. That name may be the actual name used by the data service for that data resource or a logical name, in which case the data service has to maintain a mapping (this is true for all the examples in this document).
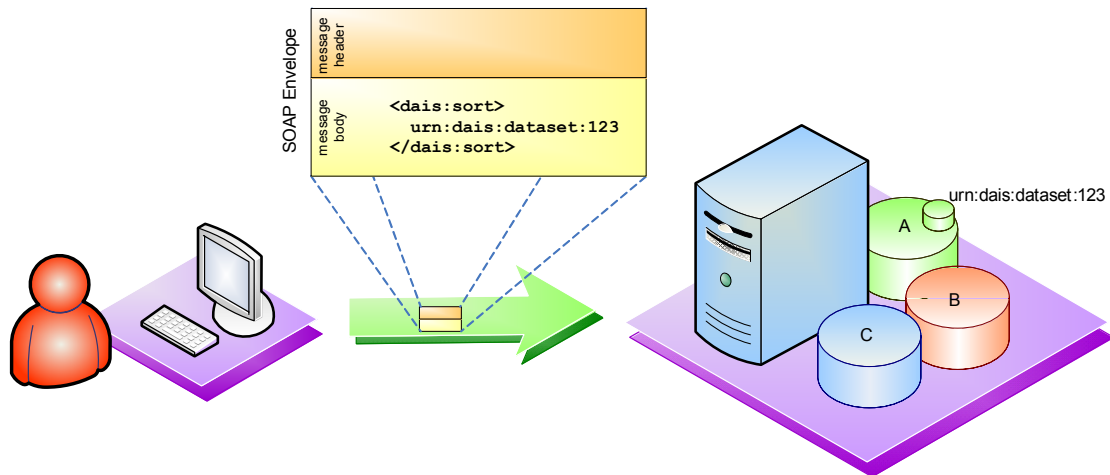
Figure 2: A message carrying the name of the data resource in the body of the message

Figure 2 shows another interaction with the data service of Figure 1. The message sent to the service is logically correlated with the data resource produced by the previous message by including the name of the data resource.
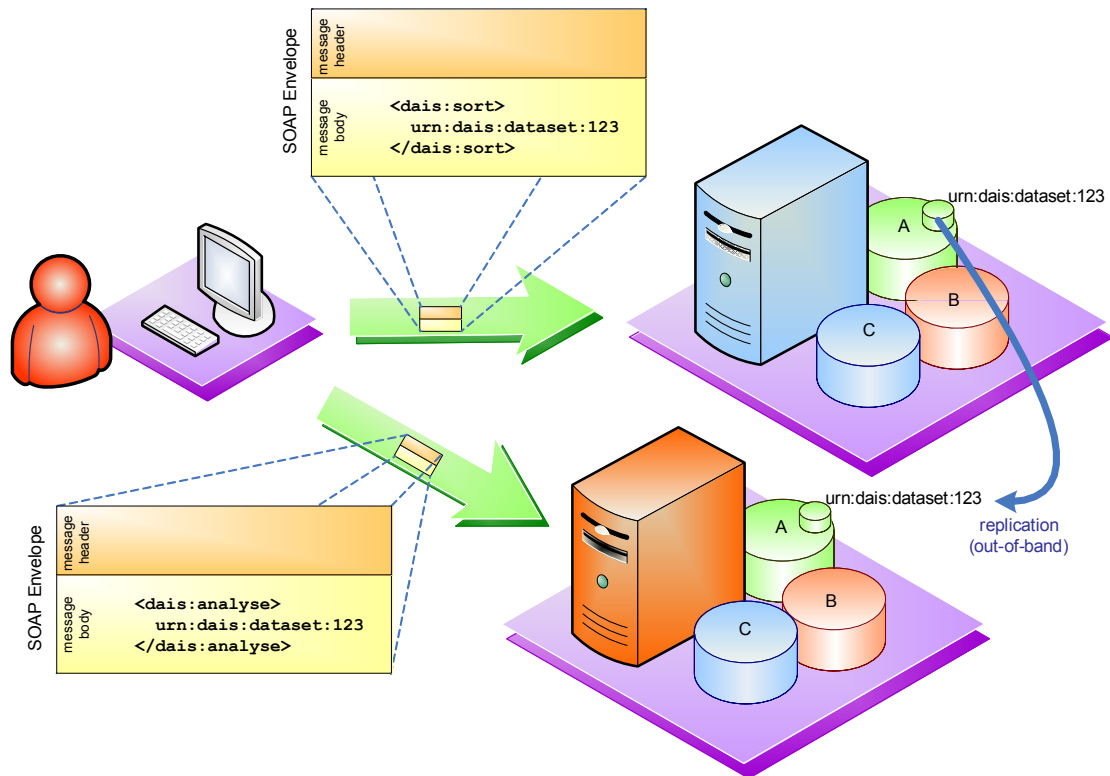
Figure 3: Interactions with multiple services using the same (logical) name for the data resource

If the data resource is subsequently replicated, for example, for quality of service reasons (through some infrastructure that is out of scope here), and access to it is offered by another service, then explicit contextualisation can still be used assuming that the same (logical) name is maintained (Figure 3).

The identity can also be passed off to a third party which can use it to access the data in the same way as the original consumer.

5.1.2. Contextualisation using WS-Context

In this example, an infrastructure supporting contextualisation is assumed (e.g., WS-Context) which models stateful interactions between services or between consumers and services.

A context structure is created (out of scope for this example) and is used in all the related message exchanges. A message sent to the service results in the identified data resource being associated with the stateful interaction. The context structure could be maintained by the participants in the interaction or by a separate service, according to the semantics of the specification used.
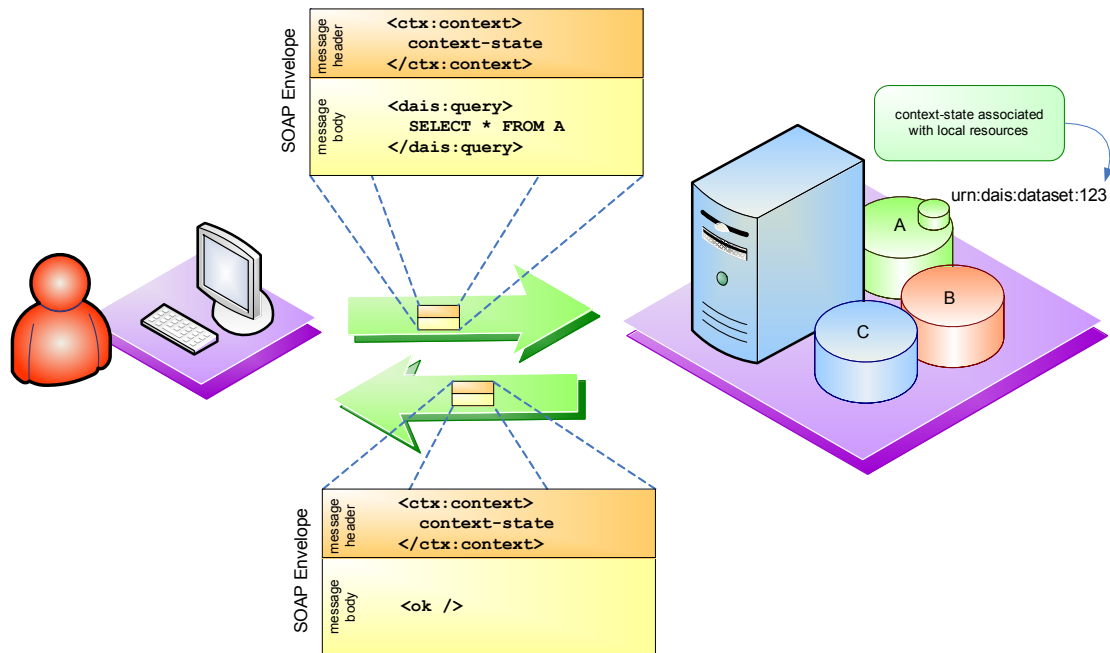
Figure 4: A data resource is generated within the scope of a contextualised interaction

A consumer sends a query to a service within the scope of a contextualised interaction (Figure 4). The service makes an association between the context and the resulting data resource. No name needs to be exposed and returned as part of the message interaction. It is assumed that the consumer somehow indicates to the runtime what context to use (how this is done is out of scope for this document) in subsequent messages.
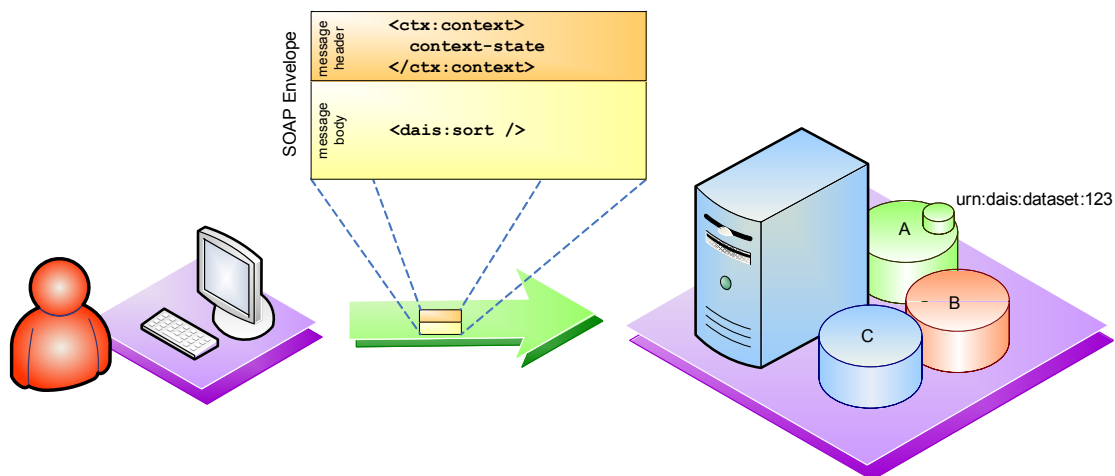
Figure 5: A contextualised message that is associated with the resulting data resource

In Figure 5, the consumer sends a contextualised message to the service. The service interprets that message within the context of the identified interaction.
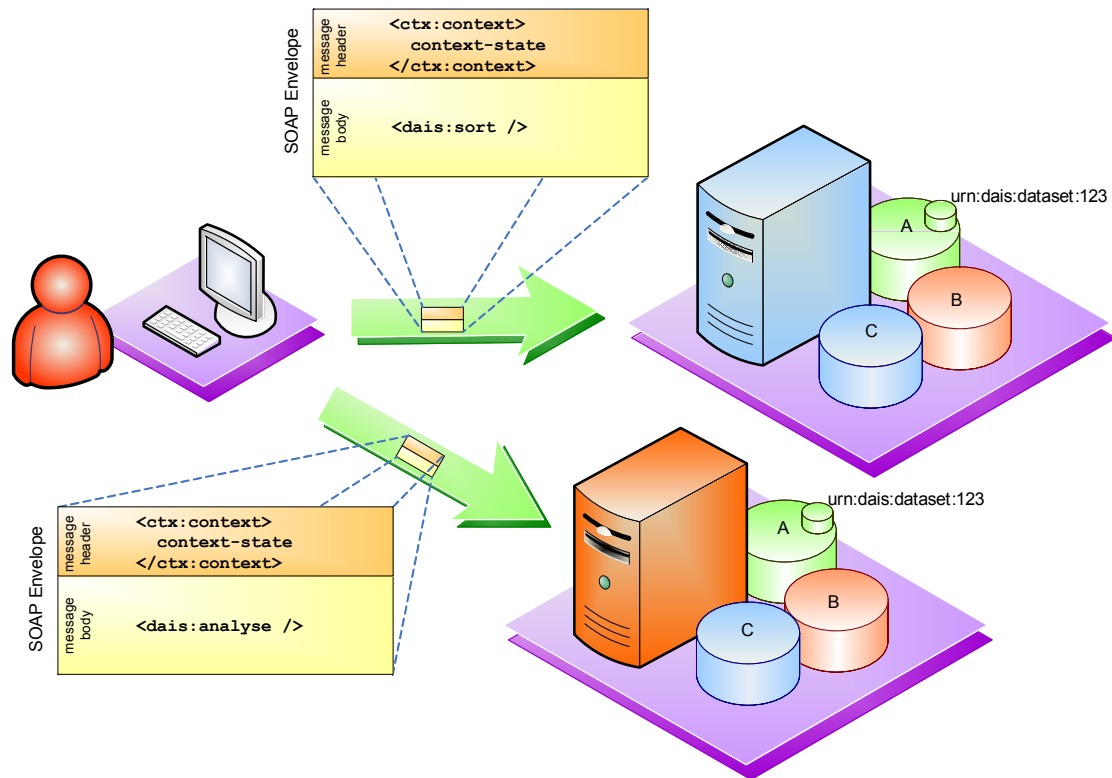
Figure 6: Messages sent to multiple services within the context of the same interaction

As shown in Figure 6, the same context can be used to send messages to multiple services that have correlated the same dataset with the context that identifies the interaction.

Another pattern of using context is shown in Figure 7 where a (logical) name for a dataset is known to the consumer (e.g., it was returned to the consumer as shown in Figure 1) . The consumer associates the identified dataset with the context of the interaction by sending an appropriate message to the service. Subsequent messages within the scope of the same interaction are interpreted by the service as being correlated with the identified dataset, because the dataset ID/name is implicitly sent across in the context.
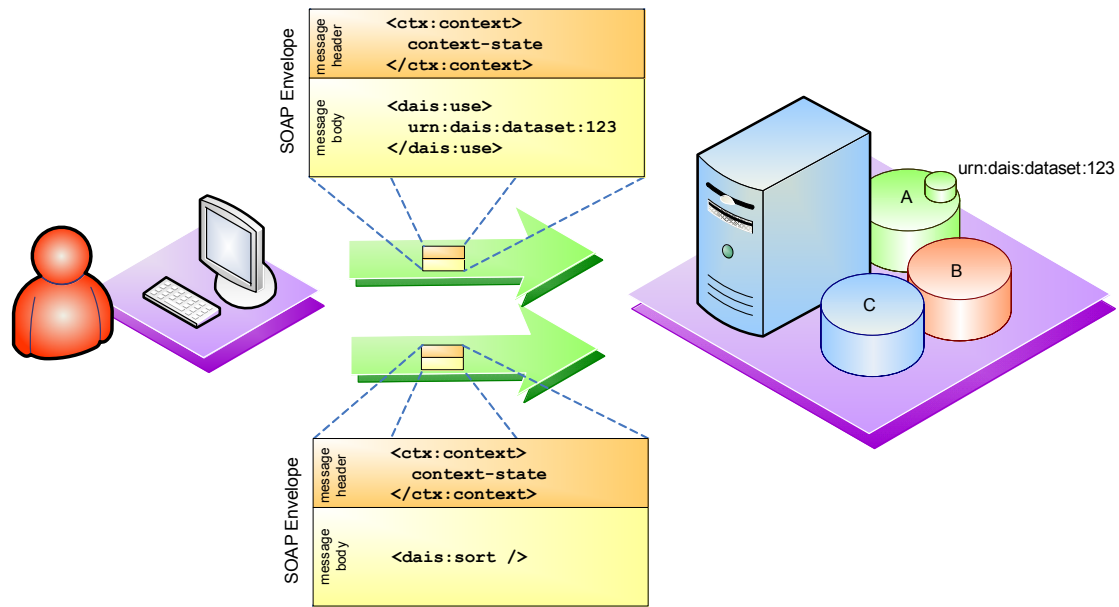
Figure 7: A dataset is associated with the context of an interaction

5.1.3. Contextualisation using WS-RF

In WS-RF, a resource, like a data resource, is identified through a WS-Addressing [8] Endpoint Reference (EPR) construct. The EPR contains endpoint-related information about a service and data specific to the service about the identified resource.

In Figure 8, a message with a query is sent to a service and an EPR identifying the resulting data resource is returned.
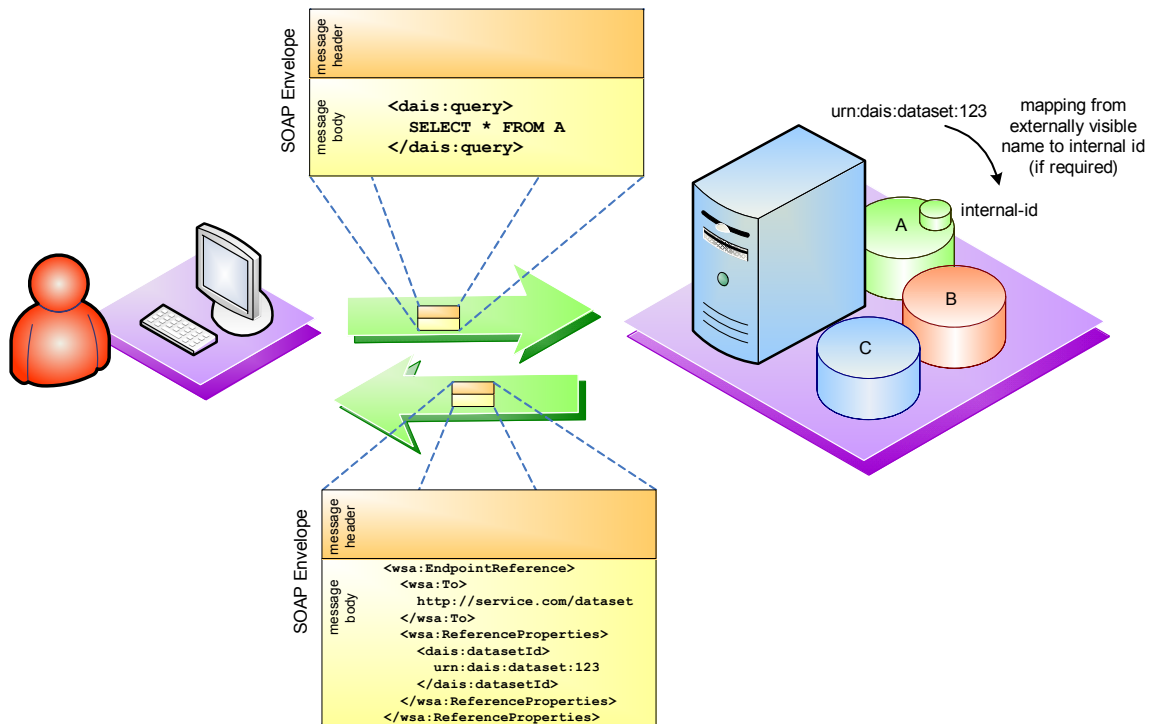
Figure 8: A query is sent to a Web Service and a WS-Addressing construct is returned

WS-Addressing ensures that the `ReferenceProperties` element of the WS-Addressing construct is included in every subsequent interaction with the service that relates to the generated data resource, as shown in Figure 9.
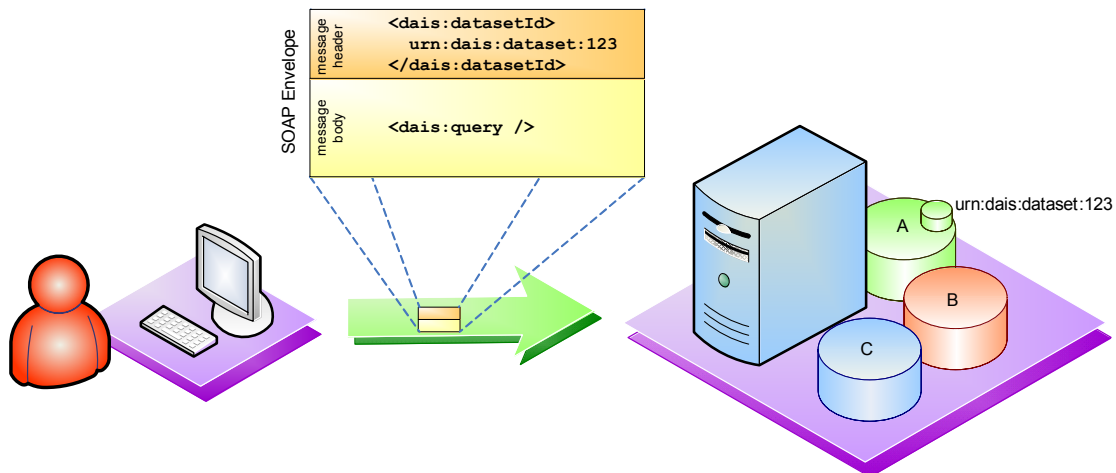
Figure 9: Implicit contextualisation using the contents of the WS-Addressing ReferneceProperties element

The WS-RF specification defines that the contents of the `ReferenceProperties` element are opaque to the consumer and they should not be interpreted in any way in order to reason about the identity of the dataset.

If a copy of the same data resource is to be accessed via a different service, a new EPR must be obtained.

If the data resource is to take part in a stateful interaction that involves multiple participants, then an approach is to explicitly associate the identity of the data resource with an externally managed context (as in the example of Figure 7).

5.2. Scenario 2 - Sessions

Consumers may want to begin a session with a data resource/service and participate in a potentially stateful conversation. In the context of the same session, the consumer may actually interact with multiple different services, instead of just one. Conversely, multiple different consumers may want to communicate in the context of a same session, with any given service.

It's important to understand first what a session is. A session is a way to group a set of messages together so the recipient of the messages has some understanding of what messages have previously been received that are somehow "related" and do the appropriate thing.

The example of sessions with a database service is not really a good example, since pretty much the only "state" information from the session is security credentials and any cursor state, which can always be recovered by establishing a new session. The real state "committed or uncommitted" of a database session is covered by the transactional semantics. But none the less, since this exercise is also to cover how the session scenario is mapped to different approaches, A DAIS example is used to illustrate this scenario.

A consumer discovers a service from a registry (out of scope). The consumer wishes to create a session with the data-oriented service in order to exchange a number of messages within the scope of that session. A consumer may wish to create a session with a data resource to establish specific quality of service characteristics, security considerations, logging purposes, etc. Context is used to identify what data resource to interact with and how that interaction should be performed.

5.2.1. Identifying sessions using WS-I stack

In this scenario, there is an explicit message sent to create a session. The message is typically sent to a session manager that is distributed. But how that session manager works or whether or not it is distributed is out-of-scope for this document. The successful response to that request is a session ID, which is then explicitly passed when sending subsequent messages, so they can be correlated. The following sub-sections discuss how this is done with different examples.

5.2.1.1 One consumer interacting with one service
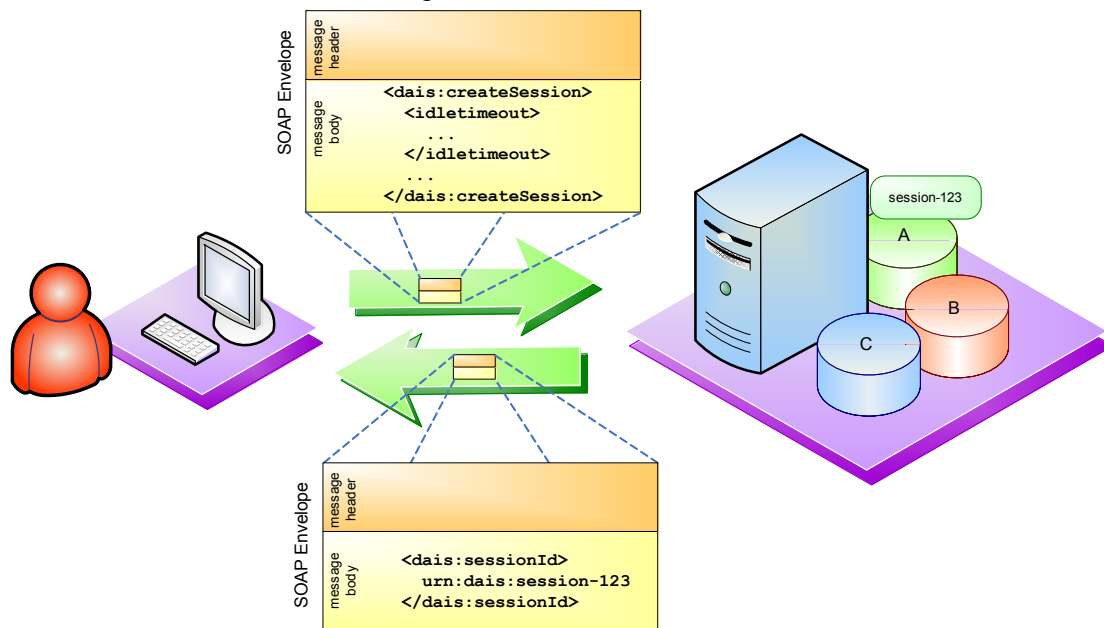
Figure 10: Create a session

In Figure 10 a message is sent to a service (or session manager) and a new session is created. A session identifier is returned to the consumer. The consumer will have to include the received session identifier in every message that is sent and which must be dealt with within the scope of the established session. In the following picture, the response message is shown to have the session ID as well, but it is not necessary for a synchronous request-response type of exchange.
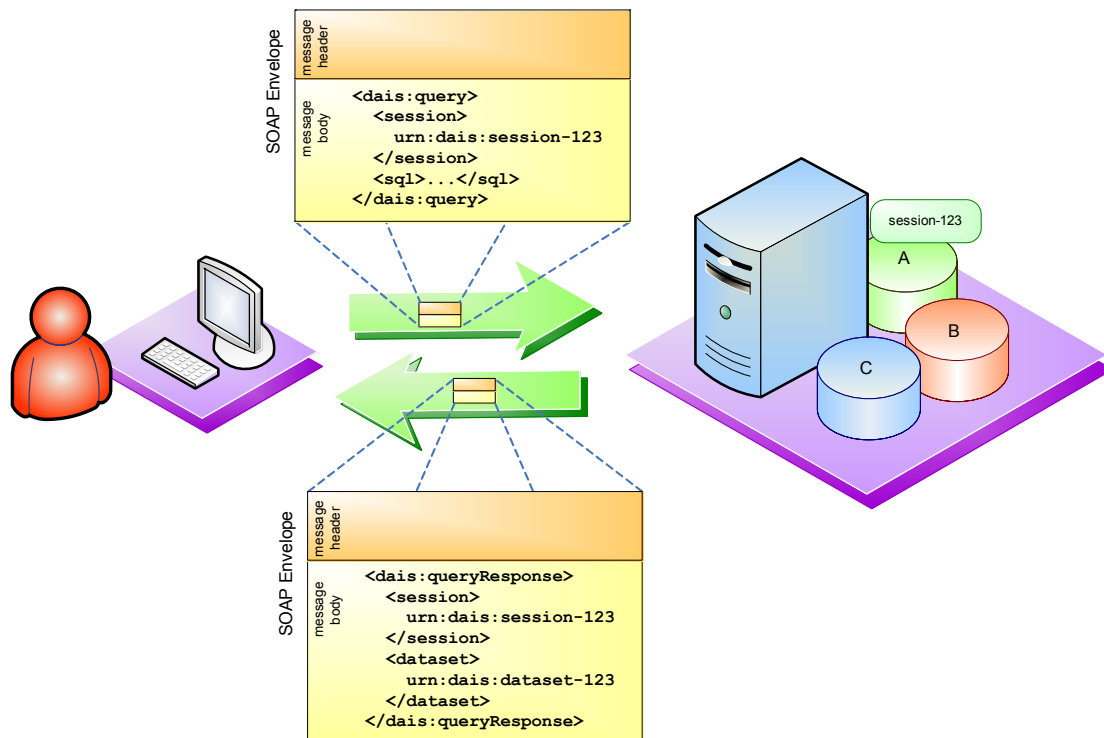
Figure 11: A message is sent within the scope of a session

Figure 11 shows a message being sent to the service with the session identifier being carried as part of the body of the message.
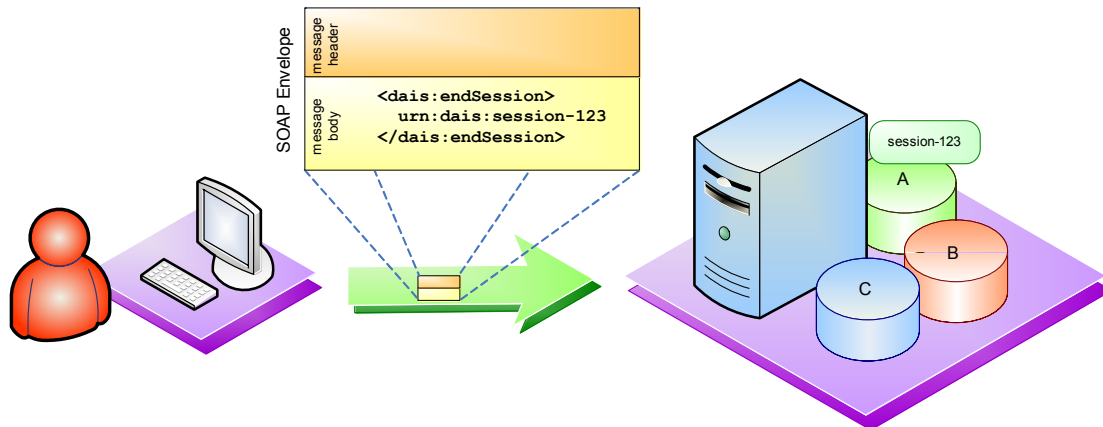
Figure 12: A message is sent to end the session

Basically, the sessionID that is obtained from the createSession message is explicitly passed around in the body of each subsequent message. When the consumer wishes to indicate that the session should be terminated, an appropriate message may be sent (Figure 12).

5.2.1.2 One consumer interacting with multiple services

In this case, the interface contract for all the services that the consumer is interacting with should be taking a session ID as parameter (message part equivalent). These multiple services will supposedly co-ordinate with a distributed session manager to correlate and retrieve any necessary conversational state, corresponding to the session ID. How that is done is beyond the scope of this document. Other than that, the messages to all the involved services, with respect to explicit session ID context will be similar to the ones listed in the previous section.

5.2.1.3 Multiple consumers interacting with a service in the context of a same session

In this scenario, one consumer establishes/creates a session and passes that sessionID to another consumer that potentially talks to the same or a different service. How these consumers exchange the sessionID information is beyond the scope of this document. Once the other consumer receives the sessionID, the messages it sends to the service are similar to the ones sent by the first consumer that created the session, except that there is no creation part for the second consumer.

Of course, the security and access issues need to be well defined for this model.

5.2.2. Identifying a Session using WS-Context

Session based communication can also be done using contexualization supported by infrastructure, if the service supports it, as indicated through a service policy document for example. In this scenario, the session interaction is modelled using WS-Context and recently published WS-MessageDelivery [9]. The service is responsible for associating the received context with any local, internal-

to-the-service resources or state. This approach scales well with multiple participants since the session is modelled as an external entity, as opposed to directly incorporating it in the service interface.
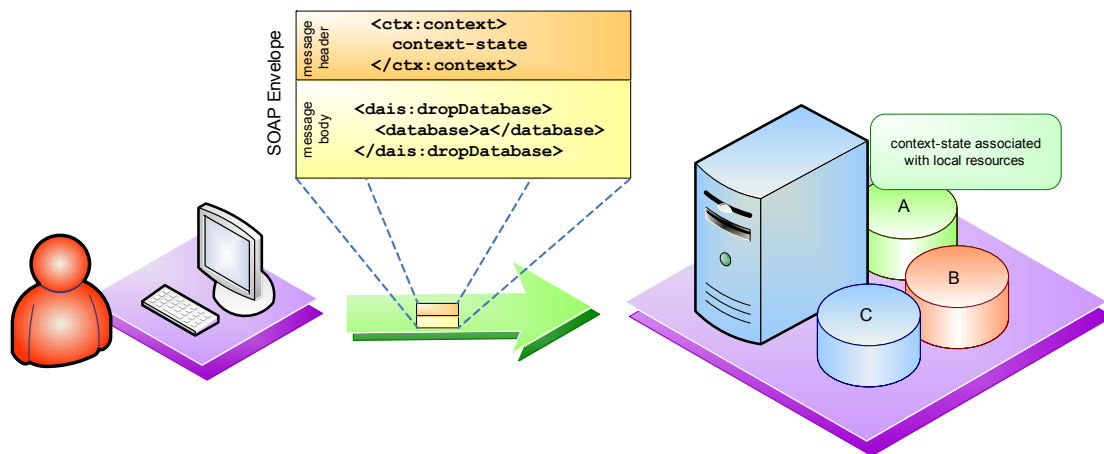
Figure 13: Using context to model a session

In this scenario, the client still initiates to create a session (either initiated by the user code or the runtime code under the covers by default on the first message) and obtains the session ID. However, after that, the sessionID is sent using a WS-Context construct in the header of subsequent messages. The service operations don't have the session ID coded in them (Figure 13).

It's presumed that the client runtime knows that it needs to insert the session context in the header because it obtained that information from the "Policy" document of the service. This sort of information (i.e, whether a session is supported, if so, how the session ID is expected back as part of each message) somehow needs to be expressed in the policy document for service discovery and interoperability.

It's similar to a transaction context, where a consumer sends a "beginTransaction" message and until an "endTransaction" message is explicitly sent, all messages include a transaction context that identifies the current transaction. Exactly how the transaction context is going to look like is defined in the respective transaction service specification. But the policy document for the service indicates that the service supports/expects certain information in the message either directly or indirectly. A similar thing applies for the sessions. We don't get into the API business here, but primarily focus on the message exchanges.

Next, the same scenarios described using the WS-I explicit model are now illustrated using the WS-Context model in the following sections.

5.2.2.1   One Consumer interacting with one service
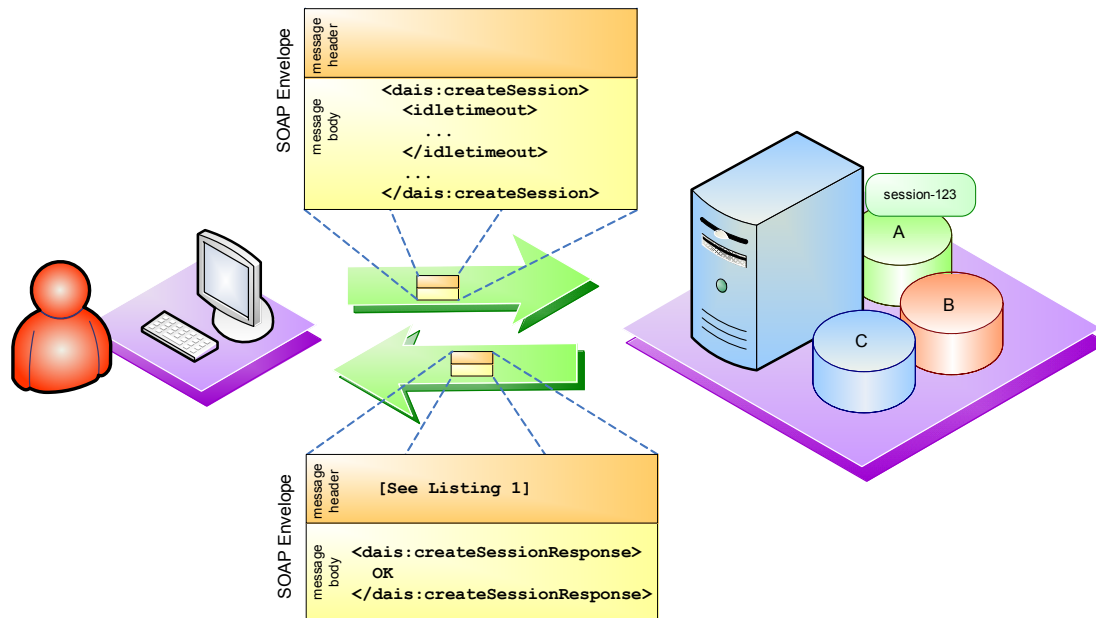
Figure 14: A new session is created and a WS-Context structure is returned
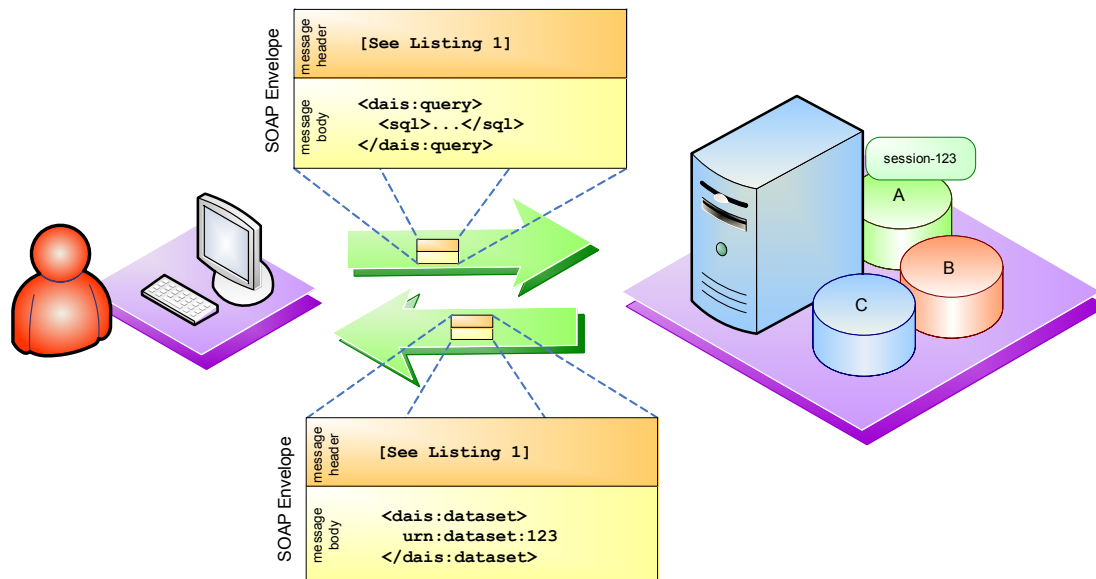
Figure 15: WS-Context is used to model session-based interactions
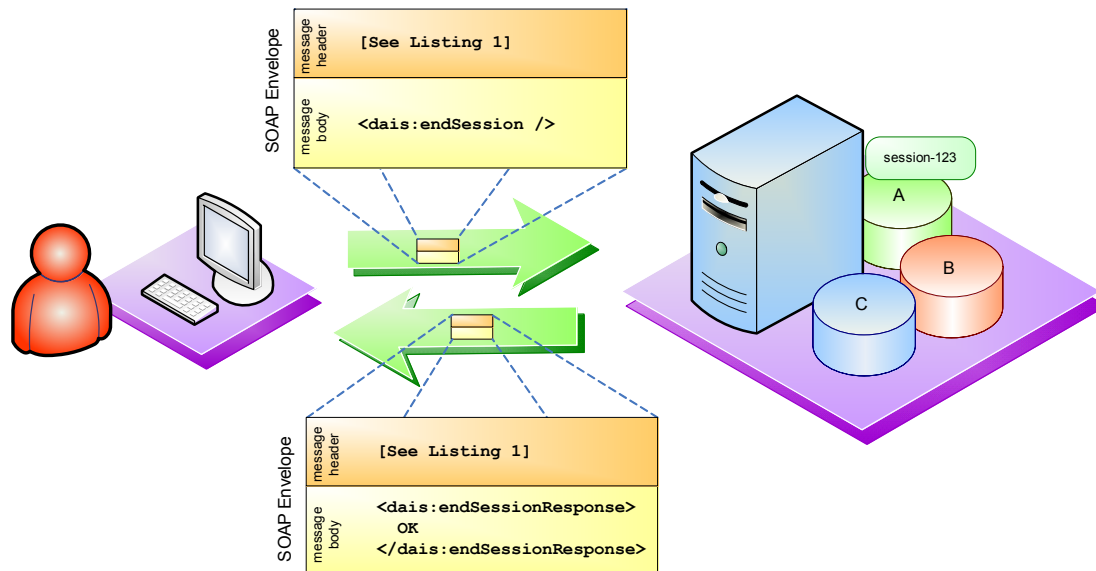
Figure 16: A session modelled using WS-Context is ended

```
<context
  xmlns="http://www.webservicestransactions.org/schemas/wsctx/2003/03"
  timeout=idle_timeout_value>
  <context-identifier>
    urn:dais:session-123
  </context-identifier>
  <!-- Optional elements -->
  <activity-list>
    <service>    <!-- wsmd:WSRef of participating service -->
      <wsdl11:service name="MyService"
                      wsmd:portType="ex:myServiceType">
        <wsdl11:port name="myPort"
                     binding="ex:myServiceBinding">
          <soapbind:address
                     location="http://example.com/wsdl-example1/impl"/>
        </wsdl11:port>
      </wsdl11:service>
    </service>
  </activity-list>
</context>
```

Listing 1: WS-Context structure in figures

5.2.2.2 *One consumer interacting with multiple services*

Once you establish a session and invoke any service in the context of this session, all the messages that are exchanged will contain the context with the session ID in the message header. The services themselves will co-ordinate with a distributed session manager, when needed.

5.2.2.3 *Multiple clients interacting with a service in the context of the same session*

Again this case is same as the explicit case except that the sessionID is sent in the headers. It is conceivable that these clients also exchange the sessionID information by including it in the body rather than headers it to pass to one another.

5.2.3. Sessions and WSRF

WSRF can be used to represent a session and sessions can be initiated across resource represented using WSRF

5.2.3.1   Sessions Represented Using WSRF

In section 5.2.1 a session is identified using some session ID. I section 5.2.2 a session is identified using a WS-Context described <context-identifier> and associated <context> structure. In the same way ws resource can be constructed to represent the properties of a session. The session is identified using and EPR. The session properties are accessed using the WS-ResourceProperties interfaces, the session lifetime is controlled using the WS-ResourceLifetime interfaces. And the session is controlled using specialised interfaces such as those defined by WS-Context.

5.2.3.2   Sessions Across WS Resources

WSRF may be being used to represent resources across which a session is required. In this cases all the approaches outlines in this section can be employed. The approach chosen depends on which approach is preferred by the developers of services and which approach is advertised as supported by the deployed service.

5.3. Scenario 3 - Discovery - Registries of data resources

Once data resources are named outside the boundaries of a service, a registry of those names may be necessary. Such registries may give information about the endpoints of the services providing access to a particular named resource or even offer richer metadata about the resource, such as lifetime, policies, semantics, etc (Figure 17).
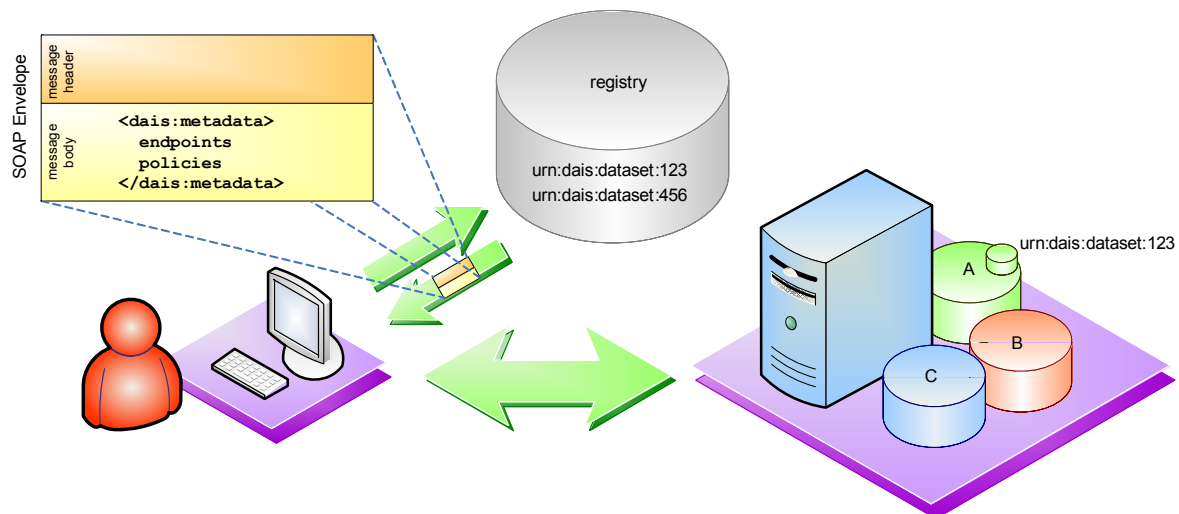
Figure 17: Looking into a registry for metadata about advertised data resources

5.3.1. Resources identified via URNs

If the names of the data resources are URNs, then those URNs could act as keys in a registry. A consumer requests the metadata information about a data resource as in the example of Figure 18.
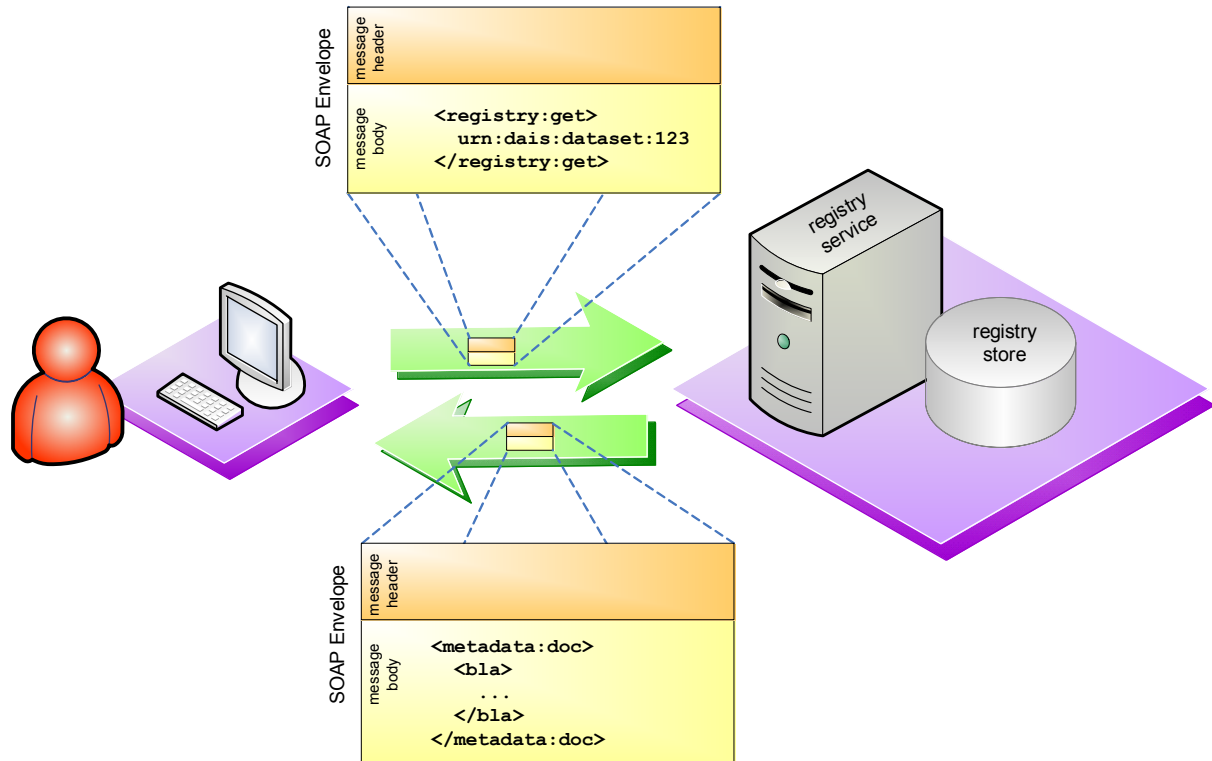
Figure 18: A query to a metadata registry using a URN

The returned metadata may include the endpoint information about the service(s) that may be offering access to the identified data resource, policy related information, WS-RF EPRs, etc.

5.3.2. Resources identified using WS-RF EPRs

WS-RF EPRs could also be used as keys in a registry for metadata information. A consumer asks the registry service for metadata information about an identified data resource, as shown in Figure 19.
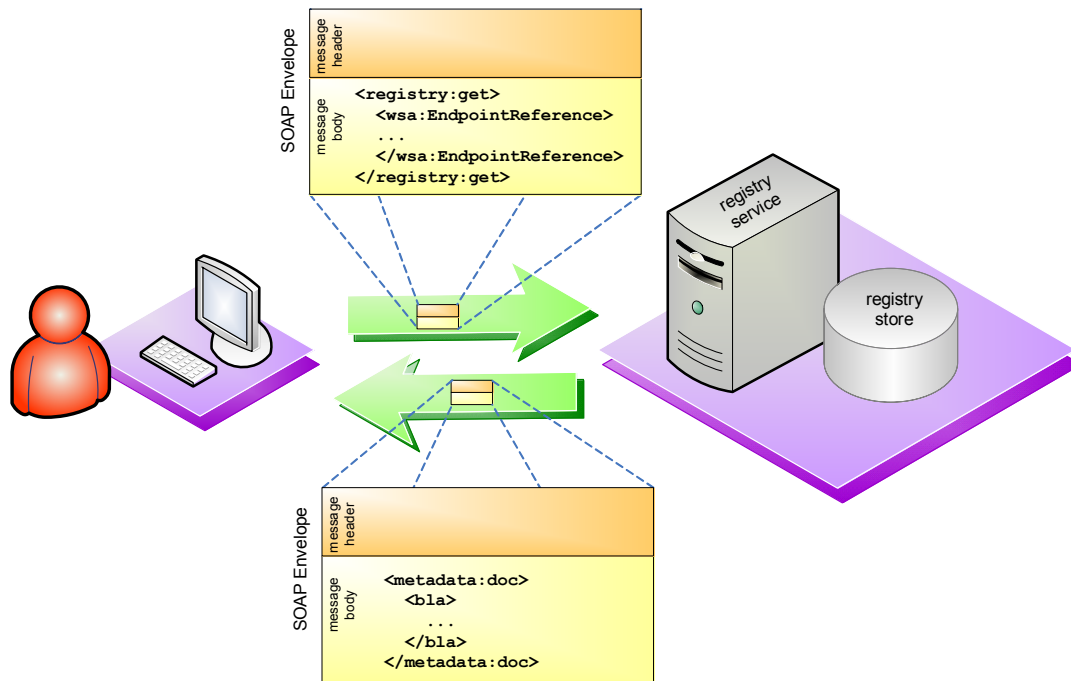
Figure 19: A query to a metadata registry using a WS-RF EPR

5.4. Scenario 4 – Access to metadata

A consumer discovers a service from a registry and wishes to obtain metadata that is not provided by the registry. Metadata has different meanings to different people. Examples include,

- Information about the service itself, e.g. interfaces supported (WSDL), active status, available endpoints, number of messages received.

- Information about the expected behaviour of the service, e.g. statements of policy.

- Information about data resources that a service interacts with, e.g database name, database schema.

Where a service exposes metadata which has relevance to a particular data resource we assume that the mechanisms for correlating requests for metadata with a data resource are as described in Scenario 1.

5.4.1. Application Defined Specific Access

With no specification to guide developers, access to metadata is typically provided by application specific operations. Information about available metadata and its structure is exposed by a service's WSDL.

When metadata is requested for an identified data resource, there is no implication that the services providing the metadata and services providing access to the data resource are one and the same.
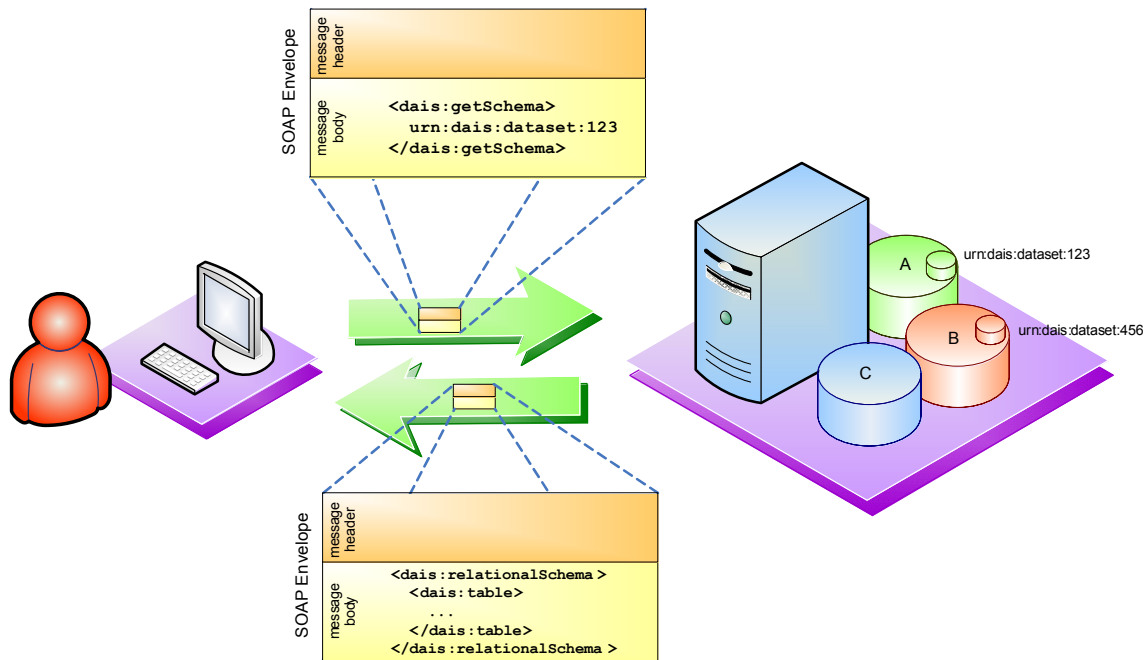
Figure 20: Application defined messages (explicit contextualisation)

5.4.2. Application Defined Generic Access

Applications may define general mechanisms for discovering and accessing metadata.
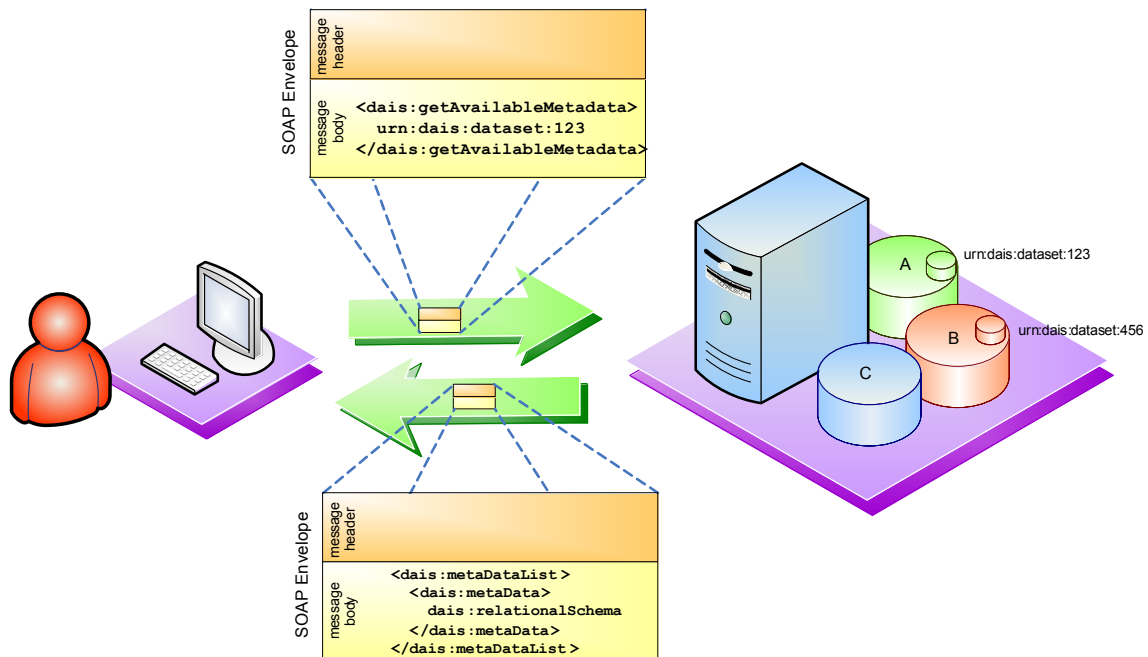
Figure 21: Discovering metadata (explicit contextualisation)
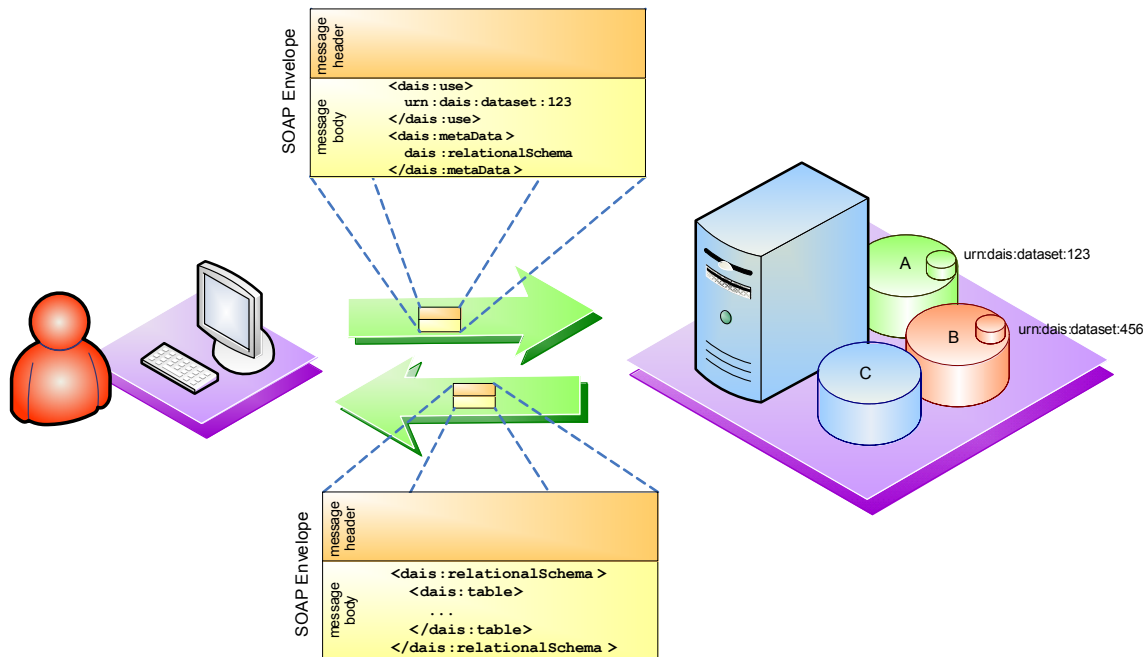
Figure 22: Retrieving metadata (explicit contextualisation)

5.4.3. WS-ResourceProperties

A specification that presents a mechanism for describing and accessing metadata as it applies to an identified WS resource. WS-ResourceProperties is a combination of the previous two approaches. Available metadata is described in the service's WSDL. Metadata is retrieved using generic message structures which are also described in the services WSDL.
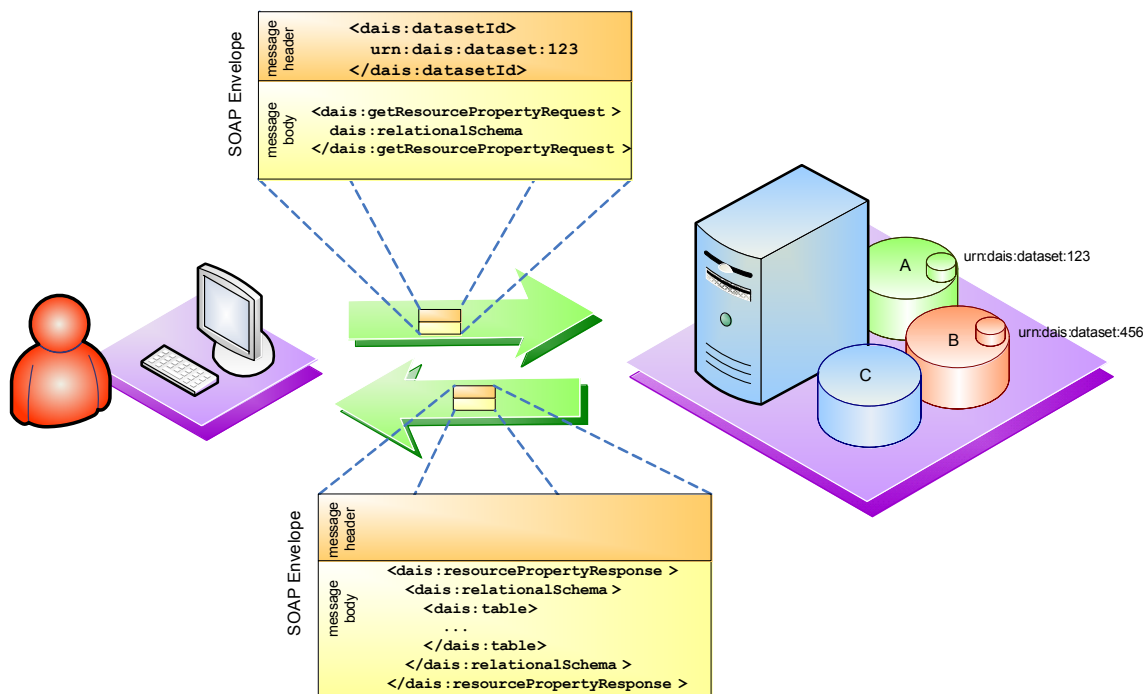
Figure 23: Retrieving metadata (WSRF contextualisation)

5.5. Scenario 5 – Third party delivery

In many cases it is useful to indicate to a service the data resource to which results of some request is to be delivered..

5.5.1. When URNs are used

If the names of the data resources are just URNs, service endpoint information must somehow be propagated to the service. This may happen in a number of ways:

- The name of the data resource (e.g., a URN) is sent together with the endpoint of the service that provides access to the named data resource (e.g., a URL).

- The processing service is expected to look into a registry for the endpoint information of the identified data resource.

5.5.2. When WS-RF WS-Addressing constructs are used

If the data resources are identified through WS-RF and WS-Addressing constructs, the service endpoint information is part of the data resource's identity and, hence, no additional mechanism is required to get access to the identified data. If for some reason the identified service no longer provides access to the identified data resource (e.g., the data resource has been moved to a different service) an additional mechanism for retrieving a new WS-Addressing construct would have to be in place. (This kind of functionality is provided by the WS-Renewable References specification). Alternatively you could repeat the search in the registry using either the id you started with or by extracting the id from the service you have been given.

6. Summary

This section refers back to the points listed in section "Goals of the Investigation" and notes observations made while considering the various approaches.

	WS-I	WS-I + WS-Context	WS-RF
The nature of the interactions with data services	If a resource must be identified, its identity must be included in the body of a message	Identity can be sent using WS-Context in the headers, without having to declare it as a parameter.	The reference property element in an EPR identifies a ws resource in a web service. Knowledge of format of reference properties not required by consumer

The representation of data resources through service interfaces	Services with appropriate interfaces for a data resource are located via registry or out of band mechanism. Data resource identified using application-domain defined conventions and identifier carried in application payload of a message.	Services with appropriate interfaces for a data resource located via registry or out of band mechanism. The Data resource is identified using a WS-context that is passed as part of the message header. The identity is assumed to be declared, as part of the metadata of the service, for example.	Services with appropriate interfaces for a data resource located via registry or out of band mechanism. Data resource identified as part of the EPR obtained. EPR resolves to information passed as part of the message header.
Accessing data resources via different portTypes offered by the same service	No difference across approaches. Port types are aggregated into the one service and resource is identified in accordance with the approach in question.	No difference across approaches. Port types are aggregated into the one service and resource is identified in accordance with the approach in question.	No difference across approaches. Port types are aggregated into the one service and resource is identified in accordance with the approach in question.
Accessing data resources via different portTypes offered by different services	Resource identity must be available and valid across services. Resource identity used with one service is passed in with message bodies to other services that are able to interpret it. Suitable services must be located via a registry or via out of band mechanisms.	Resource identity must be available and valid across services. Context used for one service is passed in with message headers to other services that are able to interpret it. Suitable services must be located via a registry of via out of band mechanisms.	Requires that the data resource identity is available across services. In the implied resource pattern an EPR identifies a specific ws resource. To access that resource through another web service would involved retrieving a new EPR from the existing service, from a registry or via an

			out of band mechanism
Providing names for data resources	The name and the information used to identify the data resource can be one and the same. Other meta data can be associated with this name in a registry.	The name and the information in the context used to identify the data resource can be one and the same. Other meta data can be associated with this name and/or context in a registry.	Reference properties are designed to be opaque and are distinct from the name (or other meta data) that a consumer would use to locate a data resource via a registry.
Creation/generati on of data resources through interactions (factory patterns)	Such a process may result in a new resource identifier. To use the new identifier a suitable service endpoint must also be returned by the factory operation or must be located via a registry or via out of band mechanisms.	Such a process may result in a new resource identifier. To use the new identifier a suitable service endpoint must also be returned by the factory operation or a suitable service must be located via a registry or via out of band mechanisms. The identifier is be passed in the message header via a WS-Context	Such a process would result in a new EPR. The new EPR can be used directly to access the new resource.
Access to metadata about data resources	A service must be designed and implemented to provide access to meta data	A service must be designed and implemented to provide access to meta data	WS-ResourceProperties is a proposal for providing access to meta data using the EPR to identify the data resource.

Access to metadata about data services	No difference across approaches. A service must be designed and implemented to provide access to meta data	No difference across approaches. A service must be designed and implemented to provide access to meta data	No difference across approaches. A service must be designed and implemented to provide access to meta data
Stateful interactions with data resources (e.g. interacting with a result set)	Not explicitly part of the model. Identity of the resource can be passed in the body.	Not explicitly part of the model. Identity of the resource can be passed using a context in the header or the body.	The implied resource pattern is used identify the resource.
Stateful interactions with data services (e.g. setting a logging context or establishing a session)	Not explicitly part of the model. Identity can be passed in the body.	Not explicitly part of the model. Identity can be passed using a context in the header or the body.	Can use either a service specific mechanism or something like WS-Context.
Locating data resources (registries)	No major difference across approaches. Meta data held by a registry allows a consumer to identify the URL of service able to act on a named resource.	No major difference across approaches. Meta data held by a registry allows a consumer to identify the URL of a service able to act on a named resource	No major difference across approaches. Meta data held by a registry allows a consumer to identify the EPR of a ws resource able to act on a named resource
Third-party delivery of data resources	Assumes that data to be delivered is identified by a resource identifier. The resource identifier along with the service endpoint is passed to the third party which arranges delivery. Third party could look up service in a registry if necessary but not required.	Assumes that data to be delivered is identified by a resource identifier. The resource identifier along with the service URL is passed to the third party, using a WS-Context, which arranges delivery. Third party could look up service in a registry.	Assumes that data to be delivered is identified by an EPR. The EPR is passed to the third part which arranges delivery

Deployment	Any web service container, e.g. Tomcat/axis, WebSphere/Oracle Application Server 10g	Any web service container + WS_context implementation	Any web service container + code to manage addressing, properties, lifetime. E.g GT4 (when available)
Lifetime management of resources	Interfaces must be defined to manage the lifetime of resource. Specifically the artifacts associated with resource identities.	Interfaces must be defined to manage the lifetime of resource. Specifically the artifacts associated with resource identities.	WS-ResourceLifetime is a proposal for the management of resource lifetitmes.
Security	Not specifically addressed but no issues specific to approaches anticipated	Not specifically addressed but no issues specific to approaches anticipated	Not specifically addressed but no issues specific to approaches anticipated

7. Observations

In this section we attempt to identify any possible implementation issues with adopting one approach over the other with regards to specific issues.

7.1. Stability of supporting infrastructure

7.1.1. WS-I

The specifications that are part of the WS-I Profile 1.0a (SOAP, WSDL, UDDI) are considered to be stable.

7.1.2. WS-I plus WS-Context

The WS-Context specification is part of the WS-CAF [10] suite of specifications which has been in the OASIS standardisation process since August 2003.

7.1.3. WS-RF

The WS-RF suite of specifications has been in the OASIS standardisation process since March 2004.

7.2. Interoperability and adoption

7.2.1. WS-I only

All major Web Services vendors support the specifications in the WS-I profile and there interoperability tests from the WS-I organisation.

7.2.2. WS-Context

WS-CAF is supported by Oracle/Sun/IONA and other companies but not by IBM and Microsoft. There are no available implementations to test interoperability.

7.2.3. WS-RF

The WS-RF suite of specifications is supported by IBM/HP/Globus and others but not by Microsoft/Oracle/Sun. Private interoperability tests between implementations have taken place.

7.3. Composability

7.3.1. WS-I only

There are no issues with using other, existing WS specifications.

7.3.2. WS-Context

There are no issues with using other, existing WS specifications.

7.3.3. WS-RF

Issues with using BPEL and WS-TransactionManagement (part of WS-CAF).

7.4. Tooling

7.4.1. WS-I only

Widely available tooling (commercial and open source).

7.4.2. WS-Context

No available tooling yet – Vendors supporting WS-Context are in the process of implementing necessary tools

7.4.3. WS-RF

Experimental implementations emerging.

7.5. Conceptual Model

7.5.1. WS-I and WS-Context

Resources are not explicitly modelled the focus is on services.

7.5.2. WSRF

Explicitly models resources and interactions with them.

8. Frequently Asked Questions

8.1. Under what circumstances might a consumer want to identify a data resource?

The DAIS specifications are written to present data as a first class citizen in the grid space. I.e. DAIS assumes that consumers wish to interact with data resources directly for query and update purposes. As data resources are treated as first class citizens there is a need to identify one from another.

Using the DAIS interfaces it is possible to present data resources with service interfaces. For example, the consumer is not aware how the data resource is physically implemented and what techniques are used to support the required quality of service requirements. Hence the identity that is used to identify data resources in the grid space may not match directly those identities used to identify physical data resources.

8.2. How could data resources be identified?

Various bodies are working on general identity schemes and many schemes already exist, for example, URN and URI. WSRF presents a scheme based on WS-Addressing endpoint references which presents the identity of a data resource as seen through a web service.

8.3. How does a consumer identify a data resource to a web service?

Typically service oriented architecture describes the use of registries to find web services. This applies equally to web services used in the grid space. It is anticipated that identity of a data resource is supplied to a registry which returns the endpoints of web services that provide access to the identified data resource.

The relationship between the web service and the data resource and the mechanism by which this relationship is qualified varies depending on the type of web service contextualisation chosen.

8.3.1. WS-I

The data resource identity is passed in as part of the body of each message sent to the web service. The web service uses this to determine which data resource(s) to operate on.

8.3.2. WS-I plus WS-Context

The data resource identity is associated with a context in some way. The context is then passed in as part of the header of each message sent to the web service.

8.3.3. WS-RF

The EPR resulting from the registry query contains the information required to identify the web service and the resource.

8.4. Is the data resource identity valid outside the context of a web service?

This is of relevance when the identity of a data resource is obtained by out of band means or must be stored for later use, e.g. a data resources name is included in a book or an article in order that the reader can start interacting with it. This may be useful if the resource identifier is to be used in a different context, for example, read by a human or stored in a database for some unknown future use.

8.4.1. WS-I

Data resource identifiers are used to identify suitable web services and are passed in with messages. There is no need to explicitly retrieve the data resource identifier as it is already available to an application.

8.4.2. WS-I plus WS-Context

Data resource identifiers are used to identify suitable web services and are used to create the implicit context for messages. Retrieving an identifier means extracting it from the context.

8.4.3. WSRF

Data resource identifiers are used to identify suitable web services. In order to retrieve an identifier from an existing EPR it must be available as one of the properties of the ws resource.

8.5. How are web services discovered that can operate on a data resource?

Web services which are able to act on an identified data resource can be discovered using a registry. Such web services may also be discovered by out of band means such as a configuration file.

8.6. How can you transfer data resource access from one web service to another?

This could be useful if there is a requirement to pass a message to a data resource that is not supported by the service interface currently being used by an application

8.6.1. WS-I

Take the data resource identifier that is passed in with each message and use it to look up in a registry an appropriate service.

8.6.2. WS-I plus WS-Context

Obtain the data resource identity from the current context and use it to look up an appropriate service in a registry.

8.6.3. WSRF

Retrieve meta data describing the data resource from resource properties and use this to look up an appropriate service in a registry.

8.7. How is a session with a data resource created?

Creating a session across a data resource is orthogonal to the identification of the resource. For example, a security session might describe the security characteristics for a sequence of messages sent to a data resource. Generally many sessions may be in progress at the same time. A number of web services specifications use the concept of a session. The mechanism chosen to represent a session depends, to a certain extent, on which web services specifications are chosen.

8.7.1. WS-I

A session identifier can be created that is used along with the resource identifier, that is sent in the message body

8.7.2. WS-I plus WS-Context

A session id can be created and passed as part of the context identifier of the context in the header.

8.7.3. WSRF

A separate implicit or explicit session identifier must be provided in order to identify the session.

8.8. How is a session involving more than one data resource created?

This presents the same problem as a session for a single data resource. An identifiable session must be created. The identifier can then be provided implicitly or explicitly along with each message to each data resource. In the case of multiple data resources the information that describes the session must be available to all of the web services that are part of the session.

8.9. How are clients identified?

This has not been discussed in this paper but it is expected that clients will be identified using information in the context that arrives with each message. Such context depends on the services policy for accepting messages.

8.10. How does tooling hide the various aspects of each solution?

Tooling can be applied at various levels. Tooling here means that tooling used by application developers in order to implement access to remote data resources.

8.10.1. WS-I

The application developer must locate a suitable web service using a registry and the data resource identifier. Tooling is unlikely to hide the element of each message that must be provided in order to identify the data resource.

8.10.2. WS-I plus WS-Context

The application developer must locate a suitable web service using a registry and the data resource identifier. The context must be defined including the data resource identifier.

Tooling can automatically add context to each message. The application must first indicate the context to use.

8.10.3. WSRF

The application developer must locate a suitable web service/ws resource using a registry and the data resource identifier. Given the resulting EPR, tooling can ensure that the correct context is provided with each message.

8.11. How are transactions used?

The answer really depends on what is meant by transactions. If this means a set of specified messages required to implement a transaction protocol, such as ws atomic transaction, then it is assumed that the mechanisms defined by the appropriate specification are adopted in all cases. In the case of ws atomic transaction the transaction context is carried implicitly and will work with any of the approaches outlined here for resource identification.

8.12. What happens when data resources are added?

In all cases the consumer must find out by out of band means that a new data resource is available.

8.13. What happens when data resources are removed?

8.13.1. WS-I

A message using the resource identifier will fail. The consumer must handle the failure.

8.13.2. WS-I plus WS-Context

A message using the resource identifier as part of its context will fail. The consumer must handle the failure. Whichever system is managing the context must remove the data resource identity from any contexts that it is part of.

8.13.3. WSRF

Messages sent to the EPR will fail. The consumer must handle the failure and stop using the EPR.

8.14. What is the equivalent of the WSRF EPR in the other schemes?

8.14.1. WS-I

The data resource identifier and the endpoint information used to locate the web service taken together.

8.14.2. WS-I plus WS-Context

The context passed in with each message and the endpoint information used to locate the web service taken together.

8.15. What happens if a web service fails?

In all cases a message sent to the web service will result in failure, probably arising from a network timeout. The consumer must handle the failure and discover another web service able to provide access to the data resource.

Tooling could hide this discovery phase or extra information could be associated with the original web service address to indicate alternative service that can be used.

8.16. Are the approaches mutually exclusive?

It is not the case that the approaches are mutually exclusive. The approaches can be combined and used together. For example, within the same application WS-Context could be used to model sessions while WS-Addressing constructs could be used to refer to WS-RF resources.

9. References

1. GGF, *Data Access and Integration Services (DAIS)*:
    https://forge.gridforum.org/projects/dais-wg.

2. Tuecke, S., et al., *Open Grid Services Infrastructure (OGSI) - Version 1.0*. 2003:
    https://forge.gridforum.org/projects/ogsi-wg.

3. Parastatidis, S., et al., *A Grid Application Framework based on Web Services Specifications and Practices*. 2003: http://www.neresc.ac.uk/ws-gaf.

4. WS-I, *Web Services Interoperrability (WS-I) Interoperability Profile 1.0a*: http://www.ws-i.org.

5. OASIS(WS-CAF), *Web Services Context (WS-CTX)*:
    http://www.iona.com/devcenter/standards/WS-CAF/WSCTX.pdf.

6. Moats, R., *RFC 2141: URN Syntax*. 1997, IETF: http://www.ietf.org/rfc/rfc2141.txt.

7. *Web Services Resource Framework (WS-RF)*. 2004: http://www.globus.org/wsrf.

8. *Web Services Addressing (WS-Addressing)*:
    http://msdn.microsoft.com/ws/2004/03/ws-addressing.

9. W3C, *Web Services Message Delivery Version 1.0 (W3C Member Submission)*:
    http://www.w3.org/Submission/2004/SUBM-ws-messagedelivery-20040426/.

10. OASIS, *Web Services Composite Application Framework (WS-CAF)*:
    http://www.iona.com/devcenter/standards/WS-CAF.

10. Editor Information

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

Sastry Malladi,
Oracle Corporation Ltd,
500, Oracle Parkway, MS 4OP 958
Redwood Shores, CA 94065
USA

Savas Parastatidis,
School of Computing Science,
University of Newcastle upon Tyne,
Newcastle upon Tyne, NE1 7RU
United Kingdom.

11. 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, in particular those at GGF10 who attended DAIS session 4 – Discussions of the implications of WS Resource Framework

12. Trademarks

IBM, DB2 and WebSphere are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Oracle is a trademark of Oracle Corporation

Globus Project and Globus Tookit are trademarks of the University of Chicago

13. 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.

14. Full Copyright Notice