1. Introduction

Web Services Resource Framework ([WSRF]) is a generic and open framework for modeling and accessing stateful resources using Web services. While Web services don’t need WSRF to implement applications that manage state, WSRF defines conventions for managing state so that applications discover, inspect, and interact with stateful resources in standard and interoperable ways.

Like other base Web services technologies (WSDL, SOAP), WSRF specifies machine-readable syntax, but its semantics are only described in plain text in the specifications, and WSRF does not use any semantic technologies (like RDF, OWL, WSMO). This report investigates whether and how WSRF could be combined with semantic technologies to enable further automatization of Web service resource interactions.

In section 2 below we briefly summarize WSRF. Section 3 analyzes where WSRF could benefit from using semantic technologies.

2. Brief summary of WSRF
WSRF was originally created by the Globus Alliance together with IBM and HP, in order to bring closer Web services and grid computing, but WSRF is in no way specific to grid computing.

WSRF consists of a number of interrelated specifications:

- **WS-Resource** contains the main terminology definitions [WSRF],
- **WS-ResourceProperties** specifies how the state of a service is modeled and accessed [WSRF-RP],
- **WS-ResourceLifetime** defines operations for service destruction and lifetime management [WSRF-RL],
- **WS-ServiceGroup** presents the notion of a group of services and specifies simple management functionalities for service groups [WSRF-SG],
- **WS-BaseFaults** is the definition of the common fault format for WSRF [WSRF-BF].

The first document – WSRF, sometimes also called WS-RAP for Resource Access Pattern – defines a WS-Resource as a stateful Web service that is identifiable and has zero or more properties expressible in XML. Every message to a WS-Resource must follow the so-called **WS-Resource Access Pattern**, which means that the resource identifier must be contained explicitly in the message. Further, a **WS-Resource Reference** is introduced, based on WS-Addressing Endpoint Reference ([WSAddressing]), for communicating references to WS-Resources.

In short, WSRF can be seen as promoting that every (sub)resource is represented as a Web service, as opposed to a single Web service whose methods accept a resource identifier parameter.

The following three subsections contain short summaries of the three main specifications dealing with resource properties, resource lifetime and service groups respectively. All these specifications use WS-BaseFaults for the common fault format.

### 2.1 WS-ResourceProperties

As already mentioned, every WS-Resource has zero or more properties expressible in XML, representing a view on the WS-Resource’s state. Without WSRF, clients of a Web service can access and manipulate its state using an application-specific interface. WSRF provides a generic and standard interface that complements the application-specific one in order to facilitate reuse and integration tasks.

All properties of a WS-Resource are modeled as XML elements inside the so-called Resource Properties Document. This document is described using an XML Schema ([XMLSchema]) element declaration. The element schema is then referenced from a WSDL ([WSDL]) interface, advertising that Web services with this interface have this known set of properties. WSRF does not constrain the cardinalities of properties or the extensibility of the resource properties document.

The following is an example of a WSDL document that defines a disk drive interface and indicates the properties of a disk drive, adopted from [WSRF-RP].
To allow generic use of resource properties, WSRF defines the following operations that a WS-Resource may support. The only mandatory operation is GetResourceProperty, all the other operations are optional.

- **GetResourceProperty** allows the client to request the value(s) of a single property identified by its namespace-qualified name (QName). Note that there may be zero or more values (depending on the cardinality specified in the resource properties document schema).
- **GetResourcePropertyDocument** retrieves the value of the whole resource property document.
- **GetMultipleResourceProperties** is an aggregate operation for retrieving the values of one or more particular resource properties in a single invocation.
- **QueryResourceProperties** allows the client to query the resource properties document using a query expression such as XPath ([XPath]).
- **PutResourcePropertyDocument** allows the client to completely replace the values of all properties of a resource.
- **InsertResourceProperties** allows the client to add new values to a single resource property.
- **UpdateResourceProperties** allows the client to change the value of a selected property by replacing the existing values with new ones.
- **DeleteResourceProperties** allows the client to remove all values of a single resource property.
- **SetResourceProperties** is an aggregate operation for doing multiple inserts, updates and/or deletes in a single invocation.

Finally, WS-ResourceProperties optionally uses WS-Notification ([WSBaseNotification, WSTopics]) to allow clients to request notification of inserts, updates and deletions made to the values of one or more resource properties.
2.2 WS-ResourceLifetime

The WS-ResourceLifetime specification standardizes the means by which a WS-Resource can be destroyed and it also defines how the lifetime of a resource can be monitored and manipulated. However, the WS-ResourceLifetime specification does not specify the means by which a resource is created. The following list enumerates the functionalities of WS-ResourceLifetime:

- **Immediate destruction** – a resource may support a simple operation named Destroy, which can then be used by the client to indicate that a resource should be destroyed (freed) immediately.
- **CurrentTime resource property** must be provided by all WS-Resources that implement WS-ResourceLifetime. This property allows the client to inspect and set a resource’s termination time without requiring a specific accuracy of clock synchronization.
- **TerminationTime resource property** contains the current termination time of the resource; it may be nil to indicate that the resource has no scheduled destruction time.
- **Requesting change to termination time** can only be done using a specific operation called SetTerminationTime, using either a specific time point or a requested duration (from the current time). The resource may, if necessary, reject any termination time change requests.

WS-ResourceLifetime does not mandate that the resource be destroyed immediately after the scheduled destruction time, however the client must not assume that the resource will be available after that time.


2.3 WS-ServiceGroup

The WS-ServiceGroup specification defines a means of representing and managing heterogeneous by-reference collections of Web services. This specification can be used to organize collections of WS-Resources, for example to build registries, or to build services that can perform collective operations on a collection of WS-Resources.

The WS-ServiceGroup specification can express group membership rules, membership constraints, and classifications using the resource property model from WS-ResourceProperties. Groups can be defined as a collection of members that meet some constraints as expressed through resource properties.

A service group is a WS-Resource and the members of the group are modeled as Entry properties in the resource property document, together with various membership constraints that a group can impose. Additionally, the member entries (not the member services) are also WS-Resources (referenced from the Entry properties) which may use WS-ResourceLifetime methods for scheduling the removal of a member service from the group (or removing it immediately).

For adding members to a group, WS-ServiceGroup defines a single operation called Add which adds one member to a group, specifying the member service reference and the initial termination time for the membership.

Similarly to the other WSRF specifications, WS-ServiceGroup also provides optional means for notification of service group modification using WS-Notification.

3. Extending WSRF with semantics

WSRF can be described as a set of protocols for manipulating WS resources. As such, it defines a number of messages (using the SOAP protocol) whose semantics are described in plain English. We do not consider this a limitation, as formal semantics are really only necessary when data gets combined, and it is seldom (if ever) the case that protocol messages would get reused outside the scope of the protocol itself. In any case, researching semantic protocols is certainly not the intent of this report.

On the other hand, WSRF deals with application data (resource properties) in the form of XML. Some such data is also defined within WSRF itself - e.g. **TerminationTime** in WS-ResourceLifetime
(see section 2.2) and the service group and service group entry WS-Resources in WS-ServiceGroup (see section 2.3). WSRF mandates that such application data be described using XML Schema and it proposes languages like XPath for querying the data. XML Schema together with XPath 2 (see [XPath2]) can express type hierarchies but they lack any axiomatic reasoning abilities. Research in annotating XML Schemas with semantics and using some semantic query language (or extending XPath with semantics) can potentially have far-reaching applicability, and the results would be reusable in WSRF as well, because WSRF does not constrain the XML Schemas used to describe the data and it allows any dialect of query language, as long as it can be identified with a dialect URI.

WSRF is closely tied to WS-Notification and it is not surprising that WS-Notification has very similar capabilities in that it also deals with XML data (described with XML Schema) but has extensible querying language support. Notification subscription based on semantic matching appears to be a natural combination, but again this would reuse the same results in semantic annotation, understanding and querying of XML data as mentioned above.

Finally, it might be very useful to include resource property availability matching in Web service discovery, i.e. looking for services based on what data they make available as their resource properties, as opposed or in addition to the capabilities of the application-specific service interfaces. Solving this problem, however, again seems to boil down to semantic handling of XML Schema.

Apparently, then, extending WSRF with semantics means extending XML Schema and query languages with semantics, which is out of scope of this particular report.

4. Conclusions and Further Work

We conclude that any semantic extensions to WSRF will be "only" applications of semantic extensions to XML Schema and XML querying technologies. We suggest that a subsequent report focuses on this area.

This document is currently a very first draft intended for initial discussion, it can be expected to evolve and change significantly in subsequent revisions.

References


Acknowledgement

The work is funded by the European Commission under the projects DIP, Knowledge Web, InfraWebs, SEKT, SWWS, ASG and Esperonto; by Science Foundation Ireland under the DERI-Lion project; by the FIT-IT (Forschung, Innovation, Technologie - Informationstechnologie) under the projects RW² and TSC.

The editors would like to thank to all the members of the WSMO working group for their advice and input to this document.

$Date: 2005/08/08 08:28:38 $

webmaster