OASIS ebXML Registry

Proposal: REST Interface

Category: New Feature

Date: July 3, 2002

Version 0.3

Authors: Matthew MacKenzie

Status of this Document

This note describes the initial proposal for the REST Interface work item for OASIS ebXML Registry V3.0. It is expected that the Federated Registries sub-team of the OASIS ebXML Registry TC will improve upon this initial proposal and then submit it for consideration by ebXML Registry TC at large.

1 Abstract

This document proposes a new feature of the OASIS ebXML Registry targeted for version 3.0. REST, or REpresentational State Transfer, is an architectural style of exposing applications via the web or other URI centric transports. The key tenet of the style is the use of URIs, or in the case of http, URL's to define the actions and parameters of an interfaces invocation. REST also tends to be biased toward the http GET action, as opposed to POST or PUT, mainly because POST/PUT based applications tend to hide all of the request information in the content which is POSTed, thereby devaluing the location specificity of the URI.

This document proposes a hybrid REST approach, with POST being used where GET is not practical. When the invocation parameters are too numerous or complicated, using POST is necessary, however, this is a hybrid approach because we try to still keep the URI somewhat meaningful even when performing a POST.

2 Motivation

The following motivations drive this proposal:

1. Provide a mechanism to be used in conjunction with the ObjectRef object which is a proposed addition to the registry information model, version 3.0, for referencing objects which are physically located in another registry. This mechanism mustn't hamper a registry's ability to manage large numbers of ObjectRefs by being too "heavy" in processing or network demands.
2. Enable distribution of registry content.
3. Provide another integration route for developers who are integrating the use of ebXML Registry into their offerings.
4. Incorporate the functionality described in an earlier proposal/best practice document for ebXML Registry v2, entitled "URL Interface to OASIS ebXML Registry".

3 External Dependencies

This proposal depends upon the following external artifacts:

- HTTP 1.1. The REST interface must be implemented upon an implementation of HTTP 1.1.

4 REST Interface

This section defines the REST interface to ebXML Registry 3.0+.

4.1 Use Cases

This section defines a couple of use cases for the REST interface.

4.1.1 Use Case: Inter-registry Object References

A non-ebXML registry such as UDDI wishes to reference and access the repository items in the repository of an ebXML Registry. For example a bindingTemplate may reference a WSDL document that is stored in an ebXML Registry's repository.

4.2 What is REST?

REST, which stands for Representational State Transfer, is an architectural style for distributed hypermedia systems. When you expose an interface to the world (or some subset thereof), you are essentially embedding method calls in the request URI. For example, if we were exposing a class named Catalog using REST, a client would send a http GET request to a URL that is formed something like the following:

http://www.mysite.com/restprocessor?object=Catalog&method=listItems

The return value would be sent back to the client synchronously in a format that is appropriate for such a request, or perhaps there is a URL parameter that can be set to define the return format (XML, HTML or CSV perhaps?). REST is more of a concept than a technology, and better yet, REST is easily implemented using standard facilities found on a web server or development environment.

4.3 Definition of the REST Interface for ebXML Registry

The specification of the REST Interface for ebXML Registry is constrained to the specification of what URI parameters must be used to specify the interface, method and invocation parameters being used.

4.3.1 What needs to be exposed?

At the bare minimum, it is necessary to expose functionality via REST to retrieve Registry Objects with. This is required to support the Registry Federation feature of ebXML Registry 3.0.
The minimum interface that needs to be exposed is explained below:

interface: ObjectQueryManager
operation: submitAdhocQueryRequest
parameters: id
response: RegistryResponse

NOTE:

Since a certain amount of interface mapping is required to expose all of the registry’s lifecycle management via REST, this document will describe how this should be done, although implementing a complete REST interface is not required by this proposal.

4.3.2 URI Parameters

This section defines the URI parameters that must be used by the REST Interface.

Parameter Name	Required	Purpose	Notes
interface	Yes	Declares the interface, or object to perform methods upon.	Example: ObjectQueryManager
operation	Yes	Declares the method to be performed on the specified interface.	Example: submitAdhocQueryRequest
param-<key> [optional]	No	Declares named parameters to be passed into the specified method call.	Example: param-id=899-677
var-<key> [optional]	No	Declares variables.	Example: var-output=HTML

5 QueryManager REST Interface

The REST Interface to QueryManager consists of the interface name “QueryManager”, and the one method defined in that interface, submitAdhocQueryRequest.
□
There are two ways to access the QueryManager via the REST interface: http GET based, and http POST based. *The GET based method represents the minimum implementation of this proposal.*

5.1 HTTP GET Based Access to QueryManager

In order to facilitate simple, ID based access to a RegistryObject, two new methods will be added to QueryManager:

- getRegistryEntryByID
- getRegistryObjectByID

To execute these requests, a URI parameter, named “id” must be used to specify the ID. The response returned will be a RegistryResponse in XML fomat. Below is a sample request and response:

Request:
□
```
GET /rest?interface=QueryManager&method=getRegistryEntryByID&param-id=urn:uuid:8788-
hhghh-ttttt HTTP/1.0
□
```
□
Response:

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 555

<?xml version="1.0"?>
<RegistryResponse />
```

5.2 HTTP POST Based Access to QueryManager

The submitAdhocQueryRequest method takes a properly formed AdhocQueryRequest, and returns a RegistryResponse in XML format. In the REST interface, the AdhocQueryRequest is delivered using the http POST action. Below is a sample request and response:

Request:

```
POST /rest?interface=QueryManager&method=submitAdhocQueryRequest HTTP/1.0
User-Agent: Foo-ebXML/1.0
Host: www.registryserver.com
Content-Type: text/xml
Content-Length: 555

<?xml version="1.0"?>
<AdhocQueryRequest />
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 555

<?xml version="1.0"?>
<RegistryResponse />
```

Please refer to the most current ebXML RS and RIM specifications for details on how to for the requests and responses mentioned above.

6 LifecycleManager REST Interface

The REST Interface to LifecycleManager consists of the interface name "LifecycleManager", and all methods defined in that interface including:

- approveObjects
- deprecateObjects
- removeObjects
- submitObjects
- updateObjects

- addSlots
- removeSlots

The requests for each method must be delivered using http POST in XML format. The return value will always be a RegistryResponse in XML format. Below is a sample request and response:

Request:

```
POST /rest?interface=LifecycleManager&method=approveObjects HTTP/1.0
User-Agent: Foo-ebXML/1.0
Host: www.registryserver.com
Content-Type: text/xml
Content-Length: 555

<?xml version="1.0"?>
<ApproveObjectsRequest />
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 555

<?xml version="1.0"?>
<RegistryResponse />
```

Please refer to the most current ebXML RS and RIM specifications for details on how to for the requests and responses mentioned above.

7 References

Fielding, Roy Thomas. *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine, 2000.

MacKenzie, Chad Matthew. *URL Interface to OASIS ebXML Registry. Best Practices Document*. http://groups.yahoo.com/group/ebxmlrr-dev/files/UAM/