OASIS ebXML Registry Technical Committee

Subject: Support for External Repository Items
Version: ~~1~~ 2
Author: Len Gallagher
Date: ~~4 October 2001~~ 11 October 2001
Type: ebRIM Issue Paper with Outline of a Proposal

Introduction

This is a revision of a proposal first distributed to this list on October 4. After comments to this list from Scott Hinkelman and off-line discussions with Nikola Stojanovic, it is clear that the attribute name "contentURI" is too ambiguous to be useful. This revised proposal changes it to "discoveryURL" in the Registry Information Model and changes it to "attachmentURI" when it is used in Registry Service XML elements for document submission or retrieval to reference a SOAP attachment. The discussion is left essentially unchanged. I retain strikeout markers in the Proposal Outline to show how little changes in this revision.

Since the very beginning of our Registry/Repository efforts, some of us have held the notion that the "Registry" may describe a "repository item" that resides elsewhere. The repository item could reside in some other ebXML conformant Registry, in some registry that is not ebXML conformant, or in some filestore that doesn't even pretend to be a registry. One of our most basic assumptions is that a RegistryEntry instance in our Registry will describe that repository item, link to any supporting metadata instances, give registration status and characteristics, and provide a mechanism for accessing the repository item. Let's agree to call these people the "Support for External Repository Item" advocates.

Other Registry team members are holding the notion that every repository item will reside in the Registry and will be accessible from its RegistryEntry instance by applying the ebRS GetContentRequest registry service. This service would take the "id" of a RegistryEntry instance as input and return the "repository item" described by that RegistryEntry. Let's agree to call these people the "Internal Repository Item Only" advocates.

What follows below is a discussion of how our existing ebRIM and ebRS specifications treat this topic and reasons why I think it is important to not exclude the "Support for External Repository Item" point of view. I close with an outline of a proposal that would remove the ambiguity from our specifications and make it clear that our Registry can support registration of repository items that reside in some external place.

NOTE: I'm making this proposal to the entire Registry group because it has an impact on many different existing sub-teams, e.g. query, ex-scheme, and cooperating registries. I thought it would be better for the whole group to see it at the same time so that comments could come from many directions.

Discussion

If one reads the ebRIM and ebRS specifications carefully, some paragraphs can be interpreted to favor the "Support for External Repository Item" point of view and some paragraphs can be interpreted to support the "Internal Repository Item Only" point of view. For example, the

GetContentRequest (cf ebRS line 3039) service seems to assume that the item exists in the Registry and that its content will be attached as a payload to the GetContentResponse (cf lines 3041-3044). There are no special responses to indicate that the item is not in this Registry and thus not accessible by this method.

On the other hand, the "contentURI" attribute of ExtrinsicObject (cf ebRIM section 6.6.2) states that this attribute is a URI to the repository item catalogued by the ExtrinsicObject instance and must be resolvable by the registry. If "Internal Repository Item Only" is the intended interpretation, then there would be no need at all for the "contentURI" attribute on ExtrinsicObject. The emphasis on the word "resolvable" was the result of my recommendation that the URI should really be a URL that was generally web-resolvable to locate a file whose contents were the repository item.

Another example of text favoring the "Support for External Repository Item" notion is the result of a ReturnRepositoryItem query (cf ebRS section 8.2.8) which supports the explicit notion of the referenced item being external to this Registry (cf rule 6b lines 1887-1890).

What I'd like to see happen is for the ebRIM to favor more clearly the "Support for External Repository Item" interpretation for ClassificationScheme and Package, in addition to ExtrinsicObject. This can best be done by adding a new attribute "discoveryURL" to the RegistryEntry class, by re-naming the "contentURI" attribute as "attachmentURI" in appropriate XML elements for submission and retrieval of documents as SOAP attachments, and by removing the term "contentURI" entirely from our specification.

Then I'd like to see more explicit RIM support for the notion that the "discoveryURL" is a web-resolvable URL that is as valid for internal repository items as it is for external repository items. For example, if the NAICS or UNSPSC classification schemes were external to the Registry, then the "discoveryURL" attribute could point to them as follows:

discoveryURL = "http://www.census.gov/epcd/naics/naicscod.txt"
or
discoveryURL = "ftp://xsun.sdct.itl.nist.gov/regrep/scheme/unspsc.txt"

NOTE: The above are real URL's and will return a complete description of each classification scheme, even if the format of the description and the structure of the node representation is not ebXML standard. However, if the NAICS and UNSPSC were internal classification schemes, then it would still be valuable to have a URL reference to locate them from outside the registry.

Suppose both NAICS and UNSPSC were internal to a NIST ebXML Registry implementation. Then the following contentURI values could be used as a "web shorthand" for a connection to the NIST Registry and submission of a GetContentRequest with the "id" of the ClassificationScheme instance as input.

discoveryURL = "http://registry.nist.gov/cgi/GetContent/id=naics_1997"
or
discoveryURL = "http://registry.nist.gov/cgi/GetContent/id=unspsc_2001"

NOTE: Neither or these URL's are real!! However, if they were real they would return the same result as does the GetContentRequest they are a shorthand for.

Conclusion

I see real value in clarifying that our Registry specifications support registration of external repository items, even if those items are intrinsic ClassificationScheme or Package instances. Such registration would be accomplished by submitting a new RegistryEntry (or a subclass) instance to the Registry, setting the objectType attribute to whatever type of object the item is, and setting the "discoveryURL" attribute to a URL that locates a complete description of the item. The following is an outline of a proposal that would accomplish this.

Revised Proposal Outline

1) ~~Move the "contentURI" attribute from ExtrinsicObject to RegistryEntry.~~ Add a new attribute named "discoveryURL" to the RegistryEntry class.

2) Clarify that the intended ~~value~~ usage of the "contentURI" attribute in XML submission or retrieval elements is as an internal pointer to a file that may be attached to a SOAP message. It is necessary to distinguish from among possibly many such attachments. As such, it is better renamed as "attachmentURI". It is NOT a required attribute in any Registry Class and therefore should be deleted from ExtrinsicObject so that it no longer appears in any ebRIM class.
~~is as a pointer to a URL that locates a file that contains the repository item. NOTE: There's been a general assumption that an ebXML Registry only allows you to register something that can be represented as an electronic file. In some cases, hopefully rare, this file might just be a web page that can be further navigated to retrieve the actual repository item.~~

3) Decide if ~~"contentURI"~~ "discoveryURL" makes sense for internal repository items. If not, declare that the ~~"contentURI"~~ "discoveryURL" attribute is non-null if and only if the RegistryEntry instance describes an external repository item. If it does make sense, then define ~~"contentURI"~~ "discoveryURL" to have a "Required" value that is a web-resolvable URL no matter if the repository item is internal or external.

4) Consider moving the remaining two attributes of ExtrinsicObject, i.e. "mimeType" and "isOpaque", to RegistryEntry. Note that these two attributes have as much value for external repository items as they do for internal ones. If these attributes get moved, then decide if there is any value in retaining the ExtrinsicObject class. It would no longer have any additional attributes or methods beyond those in RegistryEntry.

5) In ebRS, clarify what happens if a GetContentRequest references the "id" of a RegistryEntry that describes an external repository item. One solution is to do what the ReturnRepositoryItem query does and return an indicator that the repository item cannot be returned because it is an external repository item. It would then be very helpful to do as the ReturnRepositoryItem query does and return the ~~"contentURI"~~ "discoveryURL" so that the client software wouldn't be left hanging with no helpful information.

NOTE: If the local Registry has a cooperative agreement with the external Registry, then the local Registry might still return that item to the client by first retrieving it from the other Registry and then returning it to the client software as if it were an internal repository item. The client software need not get the "external item" message unless the Repository has no legal way to get the item for the client.

6) In ebRS, the “contentURI” attribute is sometimes used on submittal of an extrinsic object to identify a file appended to the SOAP message. This use remains unchanged, but it would be highlighted by changing the name of the attribute to “attachmentURI”. ~~because if a repository item is being submitted to the Registry, it is certainly not an external repository item.~~ The Registry would receive the item, store it, create a URL to retrieve it directly (only if #3 above is decided that way), and set the ~~contentURI~~ “discoveryURL” attribute to an appropriate ~~new~~ value.

These anticipated changes to the ebRIM and ebRS documents are relatively minor. If this proposal receives a positive review, and if the discussion clarifies the options described in points #3 through #5, then I volunteer to produce a complete detailed proposal within a few days.