

Oasis ebXML Registry and Repository Specification Review

Summary	1
Analysis for April 2002 Oasis Vote.....	2
Detailed Analysis.....	2
Introduction.....	2
Detailed Errors and Issues	3
Security	4
Categorization	4
Global Discovery	5
Automated Invocation.....	5
Cooperative Use Case Work.....	5
Status.....	5
Next Steps	6
Relationship to other ebXML Specifications	6
ebXML Core Components (ebCC).....	6
ebXML Message Specification (ebMS)	6
BPSS and CPPA	7
BPSS:.....	7
CPP/A:	7
References.....	8

Summary

The Oasis ebXML Registry Technical Committee has approved and released two specifications that are scheduled to be voted upon by all Oasis voting members during the month of April. The two specifications are: 1) [ebXML Registry Information Model](#) (ebRIM); and 2) [ebXML Registry Services](#) (ebRS). A detailed analysis of my position for this vote can be found later in this document.

While not clearly stipulated anywhere, these two specifications are intended for joint implementation. The ebRS describes the normative services that are required to be provided by an ebXML compliant Registry. The second document, ebRIM, describes the reference schema model that supports the registry.

At this point-in-time, there is limited support for these specifications present in the Industry. Companies that have vocalized and in some cases demonstrated support for these specifications include, Sun, XMLGlobal, Sterling Commerce, IONA, with potential users including Boeing, AIAG, the Korean Trading Network (KTNET) and a few others.

In direct contrast several key companies that have stated that they have no plans to or have not publically demonstrated any support for these specifications include IBM, Microsoft and Systinet.

Analysis for April 2002 Oasis Vote

Intel should vote No. The problems identified and discussed below have not yet been fixed. The current Oasis process does not allow the Oasis ebXML Registry TC to create the required Errata in parallel with an ongoing specification review and approval. To vote “Yes” for these to become an established Oasis Standard is not prudent. The complete fixes for the issues identified have not yet been completely documented and approved by the TC.

Intel recognizes that the results of the Oasis Standards Process should be a specification or in this case a pair of specifications as extraordinary quality work. They should be nearly flawless. These specifications contain too many issues. I strongly recommend waiting until the identified fixes have been completely documented, incorporated within the specifications, and the next revisions of the specifications have been approved by the TC before reconsidering them to be cast as Oasis Standards.

The specification should be complemented with real implementations based against their specifications. While there were emails sent early in Dec 2001 discussing v2 implementations, it is obvious at the bottom of the page at [8] that almost 4 months later there still are not a sufficient number of V2 based implementations.

To vote “Yes” based on the issues listed herein and the complete list of issues documented at [7] is simply wrong and would do an injustice to the Oasis Standard approval process.

These specifications offer some advantages over UDDI in very limited areas but in general do not offer a better solution for the industry. They are overly complex and offer generalized capabilities that are lacking users. In many respects the capabilities offered by the specifications are “solutions waiting for problems.”

The specifications have been modeled appropriately in terms of Web services and do not necessary favor one computing platform over another.

A detailed analysis of several key errors and issues follows.

Detailed Analysis

Introduction

UDDI describes itself as “the building block that will enable businesses to quickly, easily and dynamically find and transact business with one another using their preferred applications.”

ebXML describes its purpose as “Developing specifications for interoperable XML registries and repositories.” The ebXML Registry and Repository specifications describe a very generalized model. Users can register almost anything. While interesting, this capability is not necessary for business-to-business eCommerce.

Detailed Errors and Issues

The following extract from Appendix A.1 within [1] describes a “normative” WSDL implementation:

A.1 Registry Service Abstract Specification

The normative definition of the Abstract Registry Service in WSDL is defined at the following location on the web:

<http://www.oasis-open.org/committees/regrep/documents/2.0/services/Registry.wsdl>

I identified the following error when I attempted to test the WSDL files established with the specifications.

```
<ErrorOutput>
Error: There was an error processing 'http://www.oasis-
open.org/committees/regrep/documents/2.0/services/RegistrySOAPBin
ding.wsdl'
  - The document was understood, but it could not be processed.
  - The WSDL document contains links that could not be resolved.
  - There is an error in XML document (21, 26).
  - Namespace prefix 'xsd' is not defined.
</ErrorOutput>
<ErrorHints>I was unable to diagnose why this WSDL didn't work.
You may want to verify that you are using the 2001 XSD Schema
(although this isn't required by the WSDL spec, it is the only
schema version supported by many SOAP SDKs.
</ErrorHints>
```

I outlined the simple fix for this issue. The Oasis ebXML Registry TC has decided against the creation of a Specification Errata to correct for this. This said, if I were to implement the normative WSDL-based registry Services interface, I would be unable to execute these specified services.

I have since identified a relatively minor issue in [1]. The name of the file defining the normative binding for SOAP as specified in the following section is wrong. The correct spelling for the file name is registrySOAPBinding.wsdl.

A.2 Registry Service SOAP Binding

The normative definition of the concrete Registry Service binding to SOAP in WSDL is defined at the following location on the web:

<http://www.oasis-open.org/committees/regrep/documents/2.0/services/SOAPBinding.wsdl>

Against the request of many on the team list service, the specification authors believed that maintaining backward compatibility between v1.0 and these v2.0 specifications was

not required. The official comment about this is “V2.0 is not backward compatible with V1.0. In the interest of improving the specs we sacrificed backward compatibility for V2.0. We are committed to backward compatibility between V2 and V3.”

I commented to the team that some areas of the RS are greatly "over-prescribed" (e.g., Semantic Rules Lines 1452-1838 - there are 46 pages worth of Filter Query support alone). This comment was dismissed.

Within section 9.7 Access Control, there are only 3 roles defined, ContentOwner, RegistryAdministrator, and RegistryGuest. However, there is no consistency between Section 5.3, Registry Users - Table1 – the Actor column, and the information within Table11 – Default Access Control Policies – the Role column. The proposed resolution to the reviewer who raised the issue is “Yes there is a consistency issue here. It is a minor issue since we do not currently provide interface for custom access control policies.”

Security

Security modeling within this version of the specification is very minimal. The approach can be characterized as “any known entity (Submitting Organization) can publish content and anyone can view published content.” The Registry information model has been designed to allow more sophisticated security policies in future versions of this specification.

Lines 3653:3665 within [1] specify that a Registry Client MUST supply digital signatures for submitted content. While this requirement supports what most believe to be business-to-business commerce, I believe that it places an undue burden upon Oasis ebXML Registry implementations. There are also valid use cases where accepting unsigned content is appropriate. It is my opinion that this is another example of where these specifications are “over prescribed.” I propose that it may be appropriate to move this material to a non-normative appendix.

The discussion regarding Authentication in lines 3768:3784 within [1] require the use of digital certificates. It is written that the certificate may be included within the message itself or may be provided through other unspecified means. Authentication of the payload must also be done separately using the digital signature discussed within the previous paragraph. I repeat my argument that there are valid use cases where digitally signing the content may not be required.

This same portion of the specification [1] additionally states that Authentication must identify the “privileges” that the message sender is authorized to. I believe that Authorization should be more clearly separated from the Authentication process.

Categorization

Because you can register anything within an ebXML Registry, publishers have the ability to register an entire classification (i.e., categorization) scheme. Entries within an ebXML registry may directly reference nodes within these internal categories or may reference

nodes within externally managed taxonomies. This capability is unique with the ebXML specifications.

UDDI allows you to register entries that directly reference checked and unchecked taxonomies but publishers are not able to “register” their own “internal” taxonomies. Within UDDI, node operators must agree to “install” internal taxonomies. Please note that UDDI bootstraps with a substantial number of built-in taxonomies, including but not limited to NAICS, UNSPSC, ISO3166, and others.

Global Discovery

These ebXML Registry and Repository specifications offer no capability for global discovery. As such, through the use of appropriately designed tModels, and corresponding service registrations, UDDI has been posited as an appropriate registry of ebXML registries. This usage model taken to its natural limits, within the confines of basic web service discovery and invocation, obviates the need for ebXML Registries.

There is also no specification for the federation of ebXML registries. While this may be covered in future work, this limits the ability to deploy and implement ebXML compliant registries in real-world business-to-business scenarios.

Automated Invocation

With Version 2 specifications, ebXML Registry Services are now modeled through the use of Web Service Description Language (WSDL) definitions. As such, ebXML-based services can be automatically invoked just as easily as any other web services registered within UDDI modeled with WSDL definitions.

During testing, I uncovered some namespace issues (detailed above) with the specified ebXML registry services WSDL file. These issues have been accepted and should be fixed in the next version.

Cooperative Use Case Work

The breadth of this work is follow-on efforts from early work done by Sun and IBM [4]. The current effort documented at [5] takes the initial work and describes actual UDDI tModels that enable the Global Discovery of ebXML-based registries. The current effort has been a joint effort between Sun and Intel.

Status

A series of working notes are captured within a PowerPoint presentation located on the Oasis ebXML Registry Technical Committee mail archives at [5]. For members of the UDDI WG, a first draft of the UDDI Technical Note, “[UDDI AS THE REGISTRY FOR EBXML COMPONENTS](#),” is available at [6]. The UDDI WG provided three reviewers. The reviewers have provided substantial guidance regarding re-organizing the proposed tModels and their categorization schemes.

Next Steps

Complete the next revisions of the UDDI tModels and service examples and begin the review cycle with the UDDI WG again.

Relationship to other ebXML Specifications

ebXML Core Components (ebCC)

The UN/CEFACT *Core Component* solution described in the technical specification, v1.8 Core Components Specification (CC), presents a methodology for developing a common set of semantic building blocks that represent the general types of business data in use today.

CC never makes a direct reference to the Oasis ebXML TC Specifications [1] and [2]. In contrast, the references within the CC specification are made to the previously approved ebXML specifications:

ebXML Registry Information Model v1.0
ebXML Registry Services Specification v1.0.

With CC, within section 5.1.2, several series of steps are discussed that describe discovery in a registry or submission to a registry. During this entire set of descriptive steps, there is no direct mention of the use or requirement of ebXML compliant registries. Later within the specification, within section 6.1.4, I encountered the following statement that relaxes the requirement to utilize an ebXML compliant registry:

“As originally articulated in the ebXML architecture concept and perpetuated in the developing UN/CEFACT architecture concept, all *Core Components* will be recorded in an ebXML compliant registry and stored in a related repository. However, small and medium enterprise (SME) organisations may not be able to readily access such architecture. As such, it is important that the full range of *UN/CEFACT Core Components* be published in a freely available catalogue.”

This appears to allow Core Components to be registered and stored anywhere. I see the two highlighted statements in 6.1.4 to be in direct conflict with each other, but this is not an analysis about the CC specification itself. While not ruling out ebXML based registries, I interpret the intent as obviating the specific requirement for core components to be registered and stored within ebXML compliant registries.

ebXML Message Specification (ebMS)

The ebXML Messaging Specification (ebMS) is lists [1] and [2] as a non-normative reference. There are no other direct references to ebRS within ebMS. There is only one direct references to functionality expected by the implementers of ebRS, that is [1] and [2]. The reference is:

“The value of a *CPAId* element MUST be unique within a namespace mutually agreed by the two parties. This could be a concatenation of the *From* and *To PartyId* values, a URI prefixed with the Internet domain name of one of the

parties, or a namespace offered and managed by some other naming or registry service. It is RECOMMENDED that the *CPAId* be a URI.”

In other words, there is nothing specific within ebMS that mandates the use of [1] or [2].

BPSS and CPPA

There are specific references found within the BPSS and CPP/A specifications. The BPSS specification contains rather weak language, e.g., “gets stored,” within the ebXML repository. In contrast, the CPPA specification uses more specific language that allows users to implement with or without ebXML Registry compliant specifications. This is an interesting inconsistency that has never been discussed in the CPPA, BPSS, or Registry TCs.

BPSS:

The XML version of the *Business Process Specification* gets stored in the ebXML repository and registered in the ebXML registry for future retrieval.

CPP/A:

CPPs MAY be stored in a repository such as is provided by the ebXML Registry.

The *Process-Specification* document MAY be stored in a repository such as the ebXML Registry.

Figure 2 [3] illustrates forming a *CPP*. *Party A* tabulates the information to be placed in a repository for the discovery process, constructs a *CPP* that contains this information, and enters it into an ebXML Registry or similar repository along with additional information about the *Party*.

This specification makes the assumption that a *CPP* that has been registered in an ebXML or other Registry will be referenced by some Registry-assigned globally-unique identifier that MAY be used to distinguish among multiple *CPPs* belonging to the same *Party*. See section 7.1 [3] for more information.

When a *CPP* is placed in an ebXML or other Registry, the Registry assigns it a globally unique identifier (GUID) that is part of its metadata.

NOTE: A Registry cannot insert the GUID into the *CPP*. In general, a Registry does not alter the content of documents submitted to it. Furthermore, a *CPP* MAY be signed and alteration of a signed *CPP* would invalidate the signature.

References

- 1) <http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebrim.pdf> (ebRIM)
- 2) <http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebrs.pdf> (ebRS).
- 3) <http://www.ebxml.org/specs/ebCCP.pdf>
- 4) <http://www.ebxml.org/specs/rrUDDI.pdf>
- 5) <http://lists.oasis-open.org/archives/regrep-cooperating/200110/msg00012.html>
- 6) <http://groups.yahoo.com/group/uddi-wg/files/Best%20Practices%20and%20Technical%20Notes/Step%201%20-%20submitted/tn-uddi-ebxml%201.2002.doc>.
- 7) <https://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebXMLRegistry-v2-spec-IssuesList.xls>
- 8) <https://www.oasis-open.org/committees/regrep/>