UDDI Version 2.00 Replication Specification

Errata 3

05 July 2002

This version:

2.03.3

Editors:

Joel David Munter, Intel

Contributors (alphabetically):

Luc Clément, Microsoft
Barbara McKee, IBM
Andrew Hately, IBM
UDDI Business Registry Operators Council

Copyright © 2001-2002 by Accenture, Ariba, Inc., Commerce One, Inc., Fujitsu Limited, Hewlett-Packard Company, i2 Technologies, Inc., Intel Corporation, International Business Machines Corporation, Microsoft Corporation, Oracle Corporation, SAP AG, Sun Microsystems, Inc., and VeriSign, Inc. All Rights Reserved.

Contents

Introduction

This document contains the 1st, 2nd and 3rd errata to the Open Draft of the UDDI V2.00 Replication Specification. The errors and requests for clarification covered by these errata were brought to the editors' attention during the review of the Open Draft of that document. Once the review period is completed and the specification becomes final, the changes given below will be factored into the final document.

As additional errors and requests for clarification are dealt with during the review period, additional editions of this document may become necessary. Because of this, users of the Open Draft Replication Specification should verify that they have the latest version of the Errata.

Errata 1

Final XML Schema Data Types

The following types have been updated to match the W3C Recommendation XML Schema in the UDDI V2 Replication Schema file.

1) The type for the soapReplicationURL element has been changed to anyURI which is listed as one of the simple types in XML Schema
2) The type for timeOfConfigurationUpdate element was changed to string which is listed as one of the simple types in the XML Schema
3) The type for the certificate element was changed to base64Binary and the encoding element removed to conform to XML Schema

maximumTimeToSyncRegistry (Specification & Schema File changes)

This element, while noble in thought and intent, will be very difficult to verify with conformance testing and certification. The element name is tied to the (public) UDDI Business Registry and as such is not general. With both issues in mind, the element name has been changed to maximumTimeToSyncRegistry and minOccurs="0" and maxOccurs="1" attributes have been added to the element definition. A commensurate change has been made to the UDDI V2 Replication Schema file. The new definition within the Configuration file is:

```
<element name="replicationConfiguration">
    <complexType>
      <sequence>
        <element name="serialNumber" type="repl:USN_type"/>
        <element name="timeOfConfigurationUpdate" type="string"/>
        <element name="registryContact">
            <complexType>
                <sequence>
                    <element ref="api_v2:contact"/>
                </sequence>
            </complexType>
        </element>
        <element ref="repl:operator" minOccurs="0" maxOccurs="unbounded"/>
        <element name="maximumTimeToSyncRegistry" type="integer" minOccurs="0" maxOccurs="1"/>
        <element name="maximumTimeToGetChanges" type="integer"/>
        <element ref="repl:communicationGraph" minOccurs="0" maxOccurs="1"/>
      </sequence>
    </complexType>
</element>
```

Ambiguous interpretation of responseLimitCount and responseLimitVector (Specification & Schema File changes)

In section 4.1.2 of the UDDI V2.00 Replication Specification the text describes the responsibilities of the recipient of a get_changeRecords message for the case when responseLimitCount or responseLimitVector are supplied in the message. In the same section of the specification, there is currently ambiguous language that allows the recipient of the message to ignore the response limit constraints. Two paragraphs within the section have been modified. Those two paragraphs have been corrected to read:

> More specifically, the particular change records that are worthwhile to return are determined by the recipient by comparing the USNs in the caller's high water mark vector with the originating USNs of each of the changes the recipient seen from others or generated itself. The recipient should only return changes whose originating USNs greater than those listed in the

changesAlreadySeen highWaterMarkVector and less than the limit required by either the responseLimitCount or the responseLimitVector.

In nodes that support pre-bundled replication responses, the recipient of the get_changeRecords message may return more change records than requested records by the caller. In this scenario, the caller must also be prepared to deal with such redundant changes where a USN is less than the USN specified in the changesAlreadySeen highWaterMarkVector.

Change to tModel Hide and Delete (Specification File change)

In section 4.3.3 and 4.3.4 the tModel hide and delete behavior is improperly described as a multi-step deletion process. The desired effect of the delete_tModel API is to hide the tModel, but it can continue to be referenced and accessed through the get_tModelDetail message. The delete_tModel API corresponds to the changeRecordHide API in the Replication Specification. An administrative function for permanently removing a tModel was intended for the changeRecordDelete for tModels. This allows a UDDI node administrator to remove a tModel from the registry if a situation arises where it is necessary to permanently remove a tModel. The following corrections are being made to section 4.3.3 and 4.3.4

This is the new paragraph for section 4.3.3 and replaces the five paragraphs currently contained in that section after the element declaration.

A changeRecordHide element corresponds to the behavior of hiding a tModel described in the delete_tModel API in the UDDI Version 2.0 Programmer's Reference Specification. A tModel listed in a changeRecordHide should be marked as hidden, so that it is not returned in response to a find_tModel API call.

This text replaces the last sentence of the paragraphs following the element declaration in section 4.3.4

The changeRecordDelete element for deleting tModels corresponds to the administrative deletion of a tModel described in section 2.3 of the UDDI Version 2.0 Operator's Specification. The changeRecordDelete for a tModel does not correspond to any API described in the UDDI Version 2.0 Programmer's Reference Specification and should only appear in the replication stream as the result of an administrative function to permanently remove a tModel.

Incorrect element name, "soapReplicationURL" (Schema File only)

The element definition for name="soapReplicationRootURL" has been corrected to "soapReplicationURL."

This URL identifies the root against which the SOAP messages involved in UDDI replication should be made.

attribute name="acknowledgementRequested" on changeRecord (Schema File only)

To keep in conformance with the latest XML Schema, the value of the "use" attribute of the attribute named, "acknowledgementRequested," has been changed from "default" to "required".

The new complete definition is:

```
<element name="changeRecord">
    <complexType>
        <sequence>
            <element name="changeID" type="repl:changeRecordID_type"/>
            <group ref="repl:changeRecordPayload_type"/>
        </sequence>
```

```
            <attribute name="acknowledgementRequested" type="boolean" use="required"/>
        </complexType>
    </element>
```

type for certificate corrected (Specification and Schema File changes)

To keep in conformance with the latest XML Schema, the schema definition for the certificate element has been corrected. The use of encoding is no longer valid.

The new complete definition is:

```
<element name="certificate" type="base64binary" minOccurs="0" maxOccurs="unbounded">
    <annotation>
        <documentation>This element, if present, contains the certificate of the operator. This is just a handy place in which these certificates can be administratively cached.</documentation>
    </annotation>
</element>
```

Incorrect references to the namespace "api" (Specification only)

All references to the namespace "api" within all of the schema excerpts have been corrected to "api_v2." There were 83 separate occurrences.

councilContact has been renamed to registryContact (Specification and Schema File changes)

To maintain the proper generalization of element naming, the councilContact element has been renamed to registryContact. The element is still derived from the api_v2:contact element.

Errata 2

Update XML Schema File

The UDDI Replication Schema File has been updated. Several minor issues were corrected.

1) changeRecordSetAssertions: This was corrected from a Group Ref to an Element Ref.

2) The spelling of the type “base64Binary” was corrected.

3) The “example_uddi_v2_replication _elements” element was removed from within the schema.

4) As the minOccurs and maxOccurs of a particle within an “All” group must be 0 or 1, the cardinality of the following was corrected:

    a) <element ref="repl:notify_changeRecordsAvailable" />
    b) <element ref="repl:replicationConfiguration" />
    c) <element ref="repl:get_changeRecords" />
    d) <element ref="repl:changeRecords" />
    e) <element ref="repl:do_ping" />
    f) <element ref="repl:communicationGraph" />

5) Corrected the spelling of:

    a) responseLimitVector
    b) responseLimitCount

Errata 3

Update XML Schema File

The UDDI Replication Schema File has been updated. Several minor issues were corrected.

1. changeRecordSetAssertions: This payload type was removed. It was found to be indeterminate. The corresponding element within the choice element within changeRecordPayload_type, changeRecordSetAssertions, was removed.

2. The get_HighWaterMarks message was added.

3. The highWaterMarks and operatorNodeID element definitions were added

4. The operatorNodeID within the operator element is now defined by reference to the operatorNodeID element within the replication schema file.

5. The facets of the acknowledgementRequested attribute within the changeRecord element were changed to use="required" and type="boolean." The "default" facet was removed.

changeRecordSetAssertions was removed (Specification and Schema)

It was determined that changeRecordSetAssertions was indeterminate. Within Section 4.3 *Change Records*, the corresponding element within the choice element within changeRecordPayload_type, changeRecordSetAssertions, was removed. The correct replication behavior can be achieved with the commensurate set of changeRecordPublisherAssertion and changeRecordDeleteAssertion type change records. Section 4.3.10 *changeRecordSetAssertions* is no longer valid.

get_HighWaterMarks Message added (Specification and Schema)

To support adequate testing, a new API message, get_HighWaterMarks was added to the Replication schema and the specification. This API is called from one node to another node within the registry without any arguments. The message returns the latest known highWaterMarks for all of the nodes known by the node receiving the call. The schema for the new API message is:

```
<element name="get_highWaterMarks">
    <complexType>
        <sequence/>
    </complexType>
</element>
```

The return from this is:

```
<element name="highWaterMarks" type="repl:highWaterMarkVector_type"/>
```

Where highWaterMarkVector_type is defined as:

```
<complexType name="highWaterMarkVector_type" final="restriction">
    <sequence>
        <element name="highWaterMark" type="repl:changeRecordID_type"
                 minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
```

Where changeRecordID_type is defined as:

```
<complexType name="changeRecordID_type" final="restriction">
     <sequence>
         <element name="nodeID" type="repl:operatorNodeID_type"/>
         <element name="originatingUSN" type="repl:USN_type"/>
     </sequence>
</complexType>
```

0 allowed for originatingUSN in highWaterMark (Specification)

If the highest originatingUSN for a specific node within the registry is not known, then a node responding to a getHighWaterMark MUST return a highWaterMark for that node with an originatingUSN of 0 (zero).

```
<highWaterMark>
   <nodeID>…</nodeID>
   <originatingUSN>0</originatingUSN>
</highWaterMark>
```

Error Message now specified for Replication (Specification)

Processing an inbound replication message may fail due to a server internal error. The common behavior for all error cases is to return an 'E_fatalError' error code. Error reporting SHALL be that specified by Section 5.1.2 *Error reporting with the dispositionReport element* of the UDDI Programmer's API Reference specification as follows:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Envelope xmlns="http://schemas.xmlsoaporg.org/soap/envelope/">
   <Body>
        <Fault>
            <faultcode>Server</faultcode>
            <faultstring>Server Error</faultstring>
                <detail>
                    <dispositionReport xmlns="urn:uddi-org:api_v2">
                        <result errno="205xx">
                            <errInfo errCode="E_fatalError">The changeRecord type
                            is not unrecognized: XYZ</errInfo>
                        </result>
                    </dispositionReport>
                </detail>
        </Fault>
   </Body>
</Envelope>
```

Response from notify_ChangeRecordsAvailable now required (Specification)

To provide for more determinate closed loop processing, the following text was added to section 4.1.1, *notify_changeRecordsAvailable message*.

> A node MUST respond with a disposition report with the E_success error code when a valid notify_changeRecordsAvailable message is received. Success reporting SHALL be that specified by Section 5.1.2 *Error reporting with the dispositionReport element* of the UDDI Programmer's API Reference as follows:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Envelope xmlns="http://schemas.xmlsoaporg.org/soap/envelope/">
    <Body>
        <dispositionReport xmlns="urn:uddi-org:api_v2" >
            <result errno="0" >
                <errInfo errCode="E_success" />
            </result>
        </dispositionReport>
    </Body>
</Envelope>
```

Proper Handling of Dangling Keys (Specification)

The following was added to Section 4.3 *Change Records*:

> Upon the receipt of changeRecords related to publisherAssertions that refer to businesses that have been previously deleted, or access point information that refers to invalid bindingKeys, or a tModelKey of a keyedReference that refers to a tModel that no longer exist, or any attempts to project a service that no longer exist at the node, nodes MUST NOT raise replication errors. Nodes MUST include the respective changeRecords in a response to relevant get_changeRecord messages.

Proper Handling of Invalid changeRecordPayload types (Specification)

The following was added to Section 4.3 Change Records:

> In the event that any changeRecordPayload_type listed below is deprecated in a future version of this specification, transmissions of the change records of the deprecated changeRecordPayload_type MUST be treated as replication errors. The corresponding handling of those replication transmission errors is specified within Section 4.2 *Bug Detection and Processing*.

changeRecordPublisherAssertion behavior was clarified (Specification)

Within Section 4.3.5 changeRecordPublisherAssertion, the following was added:

> The fromBusinessCheck and toBusinessCheck elements are Boolean values that represent which side of the business relationship is being inserted. A changeRecordPublisherAssertion message may reference one or both sides of the relationship.

Within Section 4.3.5 changeRecordPublisherAssertion the sentences:

> If the publisherAssertion value is matched by data within the registry and a new side of the relationship is implicit from the data being replicated, then the existing record in the target Registry is marked with the second side of the business relationship. If both sides of the business relationship are present within the replication message, then the message represents a new business relationship.

was changed to:

> A changeRecordPublisherAssertion element indicates that one or both sides of the business relationship are to be inserted.
>
> a. changeRecordPublisherAssertion with:
>
> <fromBusinessCheck>**true**</fromBusinessCheck> and <toBusinessCheck>**true**</toBusinessCheck> Is used to indicate that both sides of the publisherAssertion (i.e., business relationship) are to be inserted. The two businessEntities that are referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.
>
> b. changeRecordPublisherAssertion with:

<fromBusinessCheck>**true**</fromBusinessCheck> and <toBusinessCheck>**false**</toBusinessCheck> Is used to indicate that the fromBusinessCheck side of the publisherAssertion (i.e., business relationship) is to be inserted. The businessEntity that is referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.

c. changeRecordPublisherAssertion with:

<fromBusinessCheck>**false**</fromBusinessCheck> and <toBusinessCheck>**true**</toBusinessCheck> Is used to indicate that the toBusinessCheck side of the publisherAssertion (i.e., business relationship) is to be inserted. The businessEntity that is referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.

d. changeRecordPublisherAssertion with:

<fromBusinessCheck>**false**</fromBusinessCheck> and <toBusinessCheck>**false**</toBusinessCheck> If this is received in the replication stream, such a changeRecord will not generate any change to the registry. The node SHOULD log any events such as this.

changeRecordDeleteAssertion behavior was clarified (Specification)

Within Section 4.3.6 changeRecordDeleteAssertion, the following was added:

> The fromBusinessCheck and toBusinessCheck elements are Boolean values that represent which side of the business relationship is being inserted. A changeRecordDeleteAssertion message may reference one or both sides of the relationship.

Within Section 4.3.5 changeRecordDeleteAssertion the sentences:

> If both sides of the business relationship are present within the replication message, then the message represents a new business relationship.

was changed to:

> A changeRecordDeleteAssertion element indicates that one or both sides of the business relationship are to be inserted.
>
> a. changeRecordDeleteAssertion with:
>
> <fromBusinessCheck>**true**</fromBusinessCheck> and <toBusinessCheck>**true**</toBusinessCheck> Is used to indicate that both sides of the publisherAssertion (i.e., business relationship) are to be deleted. The two businessEntities that are referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.
>
> b. changeRecordDeleteAssertion with:
>
> <fromBusinessCheck>**true**</fromBusinessCheck> and <toBusinessCheck>**false**</toBusinessCheck> Is used to indicate that the fromBusinessCheck side of the publisherAssertion (i.e., business relationship) is to be deleted. The businessEntity that is referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.
>
> c. changeRecordDeleteAssertion with:
>
> <fromBusinessCheck>**false**</fromBusinessCheck> and <toBusinessCheck>**true**</toBusinessCheck> Is used to indicate that the toBusinessCheck side of the publisherAssertion (i.e., business relationship) is to be deleted. The businessEntity that is referred to within the publisherAssertion MUST be in the custody of the operator node that originates the changeRecord.

d. changeRecordDeleteAssertion with:

<fromBusinessCheck>**false**</fromBusinessCheck> and
<toBusinessCheck>**false**</toBusinessCheck> If this is received in the replication stream, such a changeRecord will not generate any change to the registry. The node SHOULD log any events such as this.

References

1) UDDI Version 2.0 Replication Specification: (http://www.uddi.org/specification.html)

2) UDDI Version 2.0 Replication Schema: (http://www.uddi.org/specification.html)