OASIS

UDDI Spec TC

UDDI Version 2.01 Operator's Specification

UDDI Committee Specification, 19 July 2002

Editors:
> Luc Clément, Microsoft

Contributors:
> Bob Atkinson, Microsoft
> Tom Bellwood, IBM
> Sharon Boeyen, Entrust
> Matthew Dovey
> David Ehnebuske, IBM
> Greg Eisenberg, Microsoft
> Shel Finkelstein, Sun Microsystems
> Franz-Josef Fritz, SAP
> Tom Glover, IBM
> Andrew Hately, IBM
> Alan Karp, HP
> Sam Lee, Oracle
> Wil Marshman, HP
> Barbara McKee, IBM
> Joel Munter, Intel
> Andrew Nielsen, HP
> Sachar Paulus, SAP
> Venkatesan Ranganathan, HP
> Chuck Reeves, Microsoft
> Christian Thomas, Intel

Abstract:

This document describes the behavior and operational parameters required of all UDDI Node Operators.

Status:

This specification has attained the status of OASIS Standard.

Committee members should send comments on this Committee Specification to the uddi-spec@lists.oasis-open.org list. Others should subscribe to and send comments to the uddi-spec-comment@lists.oasis-open.org list. To subscribe, send an email message to uddi-spec-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

For information on whether any intellectual property claims have been disclosed that may be essential to implementing this Committee Specification, and any offers of licensing terms, please refer to the Intellectual Property Rights section of the UDDI Spec TC web page (http://www.oasis-open.org/committees/uddi-spec/).

Table of Contents

1 Introduction and Overview

This document is one of several that define the UDDI Version 2.0 Specification. It covers UDDI Node Operator requirements beyond those specified in the UDDI Version 2.0 Programmer's Reference and the UDDI Version 2.0 Replication Specification. The UDDI Version 2.0 Programmer's Reference provides a short introduction to the UDDI project and describes the public UDDI interfaces. The replication specifications describe the protocol used for UDDI Operator Nodes to synchronize registry information. This document describes the behavior and operational parameters required of all UDDI Node Operators.

There are three primary audiences for this document:

- Implementers of the UDDI Version 2.0 specifications.

- UDDI Node Operators.

- Prospective UDDI Node Operators.

Implementers will find information in this document that defines one required protocol (Custody Transfer)[1], as well as recommendations and the operational behaviors all UDDI Node Operators are required to implement as a part of their UDDI services. The goal of these policies and behaviors is to ensure that UDDI services works smoothly and consistently, both for the user community as a whole, and for the individual Node Operators. The document also touches upon such topics as site security and data management.

This document does not stand alone. In order to fully understand the full scope of the UDDI specifications, each of the documents referenced in Section 9 of this document should be reviewed in detail.

2 Managing Directory Information

UDDI Node Operators store and provide registration information for business and Web services. This section provides general guidelines for managing and maintaining this information within the UDDI registry.

2.1 Information Storage

UDDI registry entries are registered through services provided by any one of the Operator Nodes. These data registration services are defined as the "publish" operations documented in the Programmers API Specification. Publisher registration and custody transfer services are exposed by a Web browser based interface implemented at each Operator Node. An Operator Node is exclusively responsible for allowing the publisher to update or delete directory entries that were originally entered through its interfaces. As such, the Operator Node is described as having 'custody' of the set of registry entries created through its site. No UDDI Operator Nodes are permitted to make changes to entries that were not originally registered at their site. The term 'custody' is used frequently when describing management policies for UDDI information.

---

[1] The Custody Transfer tModel is but one of several tModels defined for UDDI. Please refer to the remainder of the UDDI Specifications for more information.

When storing information within the UDDI registry, the Node Operator must maintain a valid email address for the publisher of each businessEntity, tModel, and relationship assertion in its custody. This email address must not be accessible outside the Operator Node; either through replication processing between UDDI Operator Nodes or the UDDI APIs. The only exception to this requirement is in the implementation of the Custody Transfer operation described in Section 5 of this document.

2.2 Backup and Recovery

Each Node Operator is responsible for durably recording and backing up the information stored in the UDDI registry. The operator may not rely in this regard on the ability to communicate with and receive old changes from other extant Node Operators in order to restore themselves in the face of a system failure or other outage.

2.3 Updates and Deletions

The UDDI registry contains multiple types of related information. When making updates to the registered information, the integrity of the overall registry must be maintained.

When deleting information from the UDDI registry, the Node Operator must adhere to the following rules:

- When deletion of a businessEntity is initiated, the deletion of all contained businessServices for that businessEntity must also be initiated[2]. In contrast, tModels referenced there from are not so automatically deleted. The references to tModels are simply pointers, and are not contained directly within the businessEntity.

- When the deletion of a businessService is initiated, the deletion of all contained bindingTemplates for that businessService must also be initiated. Similarly, businessServices referenced through a hosting redirector are not deleted with the referencing businessService. These are also reference pointers, and not contained directly within the businessService being deleted. Again, referenced tModels are not automatically deleted.

- tModel deletion (more accurately described as deprecation or "hiding") behavior is different from that of the other UDDI entities. The result of a tModel deletion request is that it is stored in a deprecated state. After a tModel is deprecated, it must never be included in the results returned from a find_tModel request. However, calls to get_tModelDetail must continue to return the details of the tModel entry. The purpose is to support existing services that continue to implement the deprecated tModel.

- Permanent deletions of tModel information within the registry may be made administratively in the event "spam" or unauthorized information is identified within the UDDI registry. In this event, a UDDI Node Operator may insert a delete operation into the replication stream. The publisher identifier for this operation is the account associated with the Operator Node. Note that a permanent deletion of tModel information from the registry must have the prior approval of the other Node Operators participating in the replication cloud.

2.4 Operator Resignation from UDDI

For whatever reason, a given Node Operator may cease performing its responsibilities as an operator, including support of the UDDI API and replication protocols. In this case, the process

---

[2] Note that deletion of the businessService may result in the deletion of bindingTemplates as described immediately below.

described below must be followed in order to facilitate migration of the information from the resigning Operator Node to the remaining UDDI registry sites.

2.4.1 Orderly Operator Resignation

An orderly resignation of an Operator Node is one that is facilitated and supported by the Node Operator for the duration of the transition period. If an operator determines that they will no longer continue to operate their registry node, they must provide written notification to the remaining UDDI registry sites participating in the replication scheme. The Node Operator must also provide notice to all of its registered publishers and facilitate the transfer of registry information to the other operator sites. The transition period is 45 calendar days from when receipt of notification is acknowledged by the other operators. The process for transition is as follows:

The Node Operator must post a visible notice of their shutdown schedule on their web site.

The Node Operator must provide an email notification to all of its registered publishers indicating that it is no longer operating a UDDI Operator Node, and that entries must be transferred to an alternate Operator Node.

Each publisher must initiate the custody transfer process at the Source Operator[3], and select the Target Operator for their entries to be migrated. The publisher is given a 15 calendar day notice to complete the custody transfer operation.

After 15 calendar days from the initial notification message, a second notification is to be sent to the remaining publisher accounts with entries at the Source Operator. Again, the publisher must be instructed to respond within 15 calendar days and complete the migration to another site.

After a total of 30 calendar days from the initial notification, any remaining entries at the Source Operator are considered 'orphaned'. The registry entries remaining in the custody of the Source Node are processed in the same manner as required by the abrupt resignation described in Section 2.4.2 of this document.

The Source Operator must remain online for at least 15 additional calendar days, or until all other operator nodes have successfully processed all the replication updates, before shutting down. The Source Operator is required to remain online for a maximum of 30 days. If the other Operator Nodes have not successfully processed all the pending updates in that time period, any remaining issues will be resolved out of band through coordination with the other Node Operators participating in the replication cloud.

Once notification that the operator is resigning from UDDI operation is sent to the publishers, it is strongly recommended that the Node Operator no longer accept any new publisher accounts, or additional entries or changes to the registry[4] – either from the Web browser or UDDI programming interfaces. Read, replication, and custody transfer operations must be supported until the site is shut down. Once the final replication file is generated by the resigning Operator Node, no further changes to registry information, or initiation of the custody transfer operation may be permitted at the Source Operator site.

When the Operator Node is shut down, the other Node Operators participating in the replication cloud must update and distribute a revised replication configuration file as appropriate for the remaining Operator Nodes.

---

[3] See section 5 for the Custody Transfer protocol definition, and descriptions of source and target operator sites.

[4] While each operator may determine their own approach to this, one alternative to consider is implementing this restriction through lowering publishing limits to zero for each account.

2.4.2 Abrupt Operator Resignation

Should an operator fail to fulfill its responsibilities for a transition period, and cease operation in an abrupt manner, the UDDI operators will attempt to facilitate the transfer of registry entries to minimize the negative effect on other registered organizations. As account information for each publisher is not available to the other Operator Nodes, this is a best effort activity only, and some information is likely to fail to be transitioned properly.

The other UDDI Node Operators participating in the replication cloud will coordinate distribution of the transition effort amongst themselves. Guidelines are as follows:

businessEntities: Contact information in the orphaned business entities may be used to contact the business or individual responsible for the entry. Should the contact confirm their responsibility for the entry and desire to transfer to another Operator Node, the operator site will coordinate the creation of new publisher accounts, and associating business entities with them.

tModels: Since tModel registry entries do not contain contact information, it is more difficult to determine how to contact the business or individual responsible for them. In this case, manual intervention may be required. As this is an expensive process, only those tModels that are in use (referenced by a bindingTemplate or other registry entry) will be considered for transition. The remaining UDDI Node Operators will make a best effort to determine the contact responsible for the entry through specifications or information in the entry. Should this information not be available, or the expense in manually facilitating the transition be determined too significant by the other Node Operators participating in the replication cloud, the remaining in use tModels will be transitioned in fair proportion to the remaining Operator Nodes and held in 'custody' of the operator's respective publisher account.

Any entries that are successfully transitioned to a new operator will be confirmed at the other UDDI Operator Nodes through the replication process described in the custody transfer protocol.

3 Security

There are three major categories of security considerations for UDDI: Unauthorized Access, Disclosure of Information and Denial of Service. Each Node Operator is responsible for implementing its own security mechanisms, but all operators need to adhere to certain policies to ensure interoperability and to implement a core set of common security policies.

Inter-operator security and authentication requirements are defined in the UDDI Version 2.0 Replication Specification.

3.1 Required Policies

Each operator must have a set of clearly stated policies for data replication, data integrity and confidentiality, availability, administration, and audit registered with the other UDDI operators participating in the replication cloud, and visibly posted on their UDDI registry web site. Each policy subsection below asserts the minimum policies required of every UDDI Node Operator.

3.1.1 Data Replication Policy Assertions

Policies for data replication are provided in the UDDI Version 2.0 Replication Specification. Additionally, the following policies must be followed.

- Each operator is required to synchronize its copy of the UDDI registry with new replication data from each of the other adjacent operators, as required by the replication configuration, at least every 12 hours.

3.1.2 Data Management, Integrity, and Confidentiality Policy Assertions

This section describes how operators store and modify information, and with who is allowed to modify the information.

- Each operator is responsible for implementing the publish API defined in the Programmer's Reference via SSL, through HTTPS[5], to ensure data integrity and confidentiality across the network.

- Each operator is responsible for implementing its own access control mechanism. See Section 5 of this document for details on data custody.

- Each operator is responsible for properly generating UUIDs for the keyed entities saved at its site. See Section 6 for information on generating UUIDs.

- Each operator is responsible to generate the operator and authorizedName attributes of businessEntity and tModel elements saved at its site. The value of the authorizedName attribute shall be sufficient for the custodial operator to uniquely identify a publisher account as required to carry out the required authorization and audit functions. While the specific account information for an individual publisher may not be exposed outside the Operator Node implementation, the operator must be able to determine the exact publisher account that is associated with a given registry entry.

- The data that must be protected includes the business entities, tModels, and relationship assertions (hosted locally and replicated from other Node Operators), the replication configuration files, and any file containing ownership, permission, or access control information.

- Each operator must document its conventions for truncation of result sets in search response messages. This includes whether truncation is based on the number of results or the size of message returned, and the maximum number or records or message size respectively.

- Each operator must document its policy for establishing Tier 2 publisher accounts, and the publication restrictions implemented for Tier 1 publisher accounts.

3.1.3 Administration (including Privacy) Policy Assertions

Each operator must implement these policy assertions and document them in its policy files.

- Each operator must make available to registrants the UDDI notice and acceptance process for each registrant as documented in Exhibit B of the UDDI Operators Agreement. For online users there will be a browser-based Graphical User Interface with a click through Interface. See Section 10 of this document for details.

- Each operator must retain an indication of acknowledgment of these policies by the registrant for audit purposes. Section 3.3 of the UDDI Operators Agreement identifies non-compliance concerns and operator responsibilities for complying with Third Party audits.

- Each operator must provide appropriate privacy and security notices, including instructions for the operator who will handle this information should it choose to no longer be a UDDI Node Operator.

---

[5] See Section 9 of this document for details on obtaining SSL specifications.

3.1.4 Availability Policy Assertions

This section details the efforts required to ensure that users have continued access to UDDI registry information and resources. In the event of an outage at any UDDI Operator Node, all registry information is available from each of the alternate operators. Additionally, each Node Operator must adhere to the following:

* Each Operator Node must make publicly available their policies for UDDI service availability.

* Each Operator Node must make clearly visible planned outage and maintenance schedules on their web site.

3.1.5 Audit Policy Assertions

Each operator is responsible for collecting audit information for the 12 most recent months of operation and for making this information available to the Operators Council should a dispute be filed with that body. The process of handling a dispute claim is documented in Section 3.1.6 of this document.

As a point of implementation, it should be noted that the requirements for the replication change record journal and the audit log are similar in nature, and can likely share much implementation within a UDDI Operator Node.

Audit requirements are related to interaction through both the API, and replication processing.

* All save, update, and delete operations upon businessEntity, businessService, bindingTemplate, and tModel entities must be audited, be they through a Node Operator's user interface or through the specified UDDI API. For such operations, the following items must be made manifest in the audit log.

    o An identification of the publisher account on whose authority the publish, update, or delete was carried out,

    o An identification of the instant in time at that Operator Node at which the operation occurred,

    o If invoked through the UDDI network API, the name of the API used to carry out the operation,

    o The UUID keys which identify the top-level entity(s) involved,

    o The local USN[6] assigned to the operation in the operator's change record journal. Note that multiple change records may be generated from a single save or delete operation. In this case, unique journal entries should be made for each entry updated, each with an identical time instant.

* All save, update, and delete operations upon publisherAssertion entities must be audited, but they through a Node Operator's user interface or through the specified UDDI API. For such operations, the following items must be made manifest in the audit log.

    o An identification of the publisher account on whose authority the publish, update, or delete was carried out,

    o An identification of the instant in time at that Operator Node at which the operation occurred,

---

[6] USNs are defined in the UDDI Version 2.0 Replication Specification.

- o If invoked through the UDDI network API, the name of the API used to carry out the operation,

- o The UUID keys which identify the top-level entities involved. Note that both keys party to the publisherAssertion must be stored.

- o The local USN[7] assigned to the operation in the operator's change record journal. Note that multiple change records may be generated from a single save or delete operation. In this case, unique journal entries should be made for each entry updated, each with an identical time instant.

- Registrations of new authorized publishers at each Operator Node must be audited. For such registrations, at least the following information must be recorded.

  - o An identification of the publisher account that was created,

  - o An identification of the instant in time at that Operator Node at which the operation occurred,

  - o An email address by which the publisher can be contacted.

- The data custody transfer from one publisher account to another must be audited at both the new and originating Operator Nodes. The following audit information must be recorded for such change requests.

  - o An identification of the publisher account on whose authority the custody transfer operation was carried out,

  - o When the change of custody is completed, the local USN assigned to the change record for the custody transfer in the Node Operator's change record journal,

  - o An identification of the instant in time at that Operator Node at which the operation occurred,

  - o The UUID keys of the UDDI data whose custody was transferred,

  - o The UUID of both the custody-relinquishing and custody-accepting Operator Nodes involved in the transfer.

- The receipt of change records from other nodes during replication must be audited. The following audit information must be recorded for such change records.

  - o The local USN assigned to the operation in the operator's change record journal.

  - o The UUID of the originating node and the originating USN associated with the change record.

3.1.5.1 Audited Information

The following information is required to be included in the UDDI Node Operator audit trail.

Publisher account: This is a unique identifier for the authenticated account that was used to complete the operation in the UDDI registry. It may be either an internal identifier, or the identifier that is returned as the value of the authorizedName attribute. In either case, the identifier value must be sufficient to uniquely identify a specific publisher account.

Time Stamp: The date and time the operation was performed. This value should be in the standard XML timeInstant[8] data type.

---

[7] USNs are defined in the UDDI Version 2.0 Replication Specification.

[8] See the UDDI Version 2.0 Replication Specification for additional references to the timeInstant data type.

Name of API: The name of the message processed. This value generally corresponds to the name of the first element within the SOAP body of the message.   Examples include *save_business* or *delete_tModel*.

UUID of major entity(s): The unique identifiers for each registry entry that was changed. These include *businessKey*, *serviceKey*, *bindingKey*, and *tModelKey* attributes, as well as identifiers for changes to relationship assertions.

Email address: The validated email address that was used by the publisher when their account was created at the Operator Node, and any subsequent updates made by the publisher.

3.1.6 Policy for Contesting Entries

The UDDI Terms of Use require that information published to the UDDI registry be accurate, and that the publisher has authorization from the business to represent them within UDDI. However, unscrupulous individuals may choose to violate this agreement and publish unauthorized information. In this case, the Node Operator should follow the process outlined here in a best effort attempt to address the issue.

1. Unauthorized entries may be brought to the attention of the Node Operator by a member of the UDDI Community, the business potentially misrepresented, or another party. Notification must include appropriate documentation to prove that the entry is not authorized, or misrepresents the party in question. Notification must be made via Email to the UDDI coordinator of the operator activities.

2. Upon notification, the Node Operator will determine whether the report warrants further review. If so, the Node Operator will notify the publisher via email of the entry that it is under review, and will request that the publisher either remove the entry, or provide proof of authorization to represent the company. Alternately, per the UDDI terms of use, the Node Operator may elect to simply remove the registry entry. In this case, it is expected that notification of the removal will be provided to the publisher.

3. If the original publisher continues to assert that the entry is valid, the Node Operator may again choose to delete the entry, or notify the party reporting the issue that an external authority must resolve it before any action is taken.

4. Should an operator delete a registry entry, this delete operation must be completed under a publisher account associated with the operator node. The audit log must reflect the publisher account used to delete the registry entry.

In any case, the Node Operator is not responsible for arbitration between the parties, and may choose to remove any entries at any time. An effort should be made to assist in resolving any issues, but only in those cases where it is clear that a business is being misrepresented. All other issues should be settled directly between the parties though established legal processes.

3.2 Approved Certificate Authorities

Operators must obtain their X.509 V3 certificate from any of the following Certificate Authorities:

- Verisign: Secure Server Security CA (http://www.verisign.com)

- Equifax: Thawte Server CA (http://www.equifax.com)

Note that the UDDI operators may certify additional Certificate Authorities. Any additions to this list will be available from the uddi.org web site.

In certificates that the Certificate Authorities issue, the 'issuer name' field will, respectively, be:

- OU=Secure Server Certification Authority, O=RSA Data Security, Inc., C=US

- CN=Equifax Secure E-Business CA-2, O=Equifax Secure Inc, C=US

4 Data Management Requirements

Each UDDI operator must adhere to certain rules regarding manipulation and management of registry data. These rules are described below.

4.1 XML Standards

Given the use of XML and SOAP within UDDI, an operator must take care to adhere to these standards when processing data. Specific additional requirements layered upon these standards may be included in the UDDI specification. Each operator is responsible for implementing these additional requirements as well as those defined by the published standards.

4.1.1 XML White Space Processing Requirements

Leading and trailing white space (space, tab, carriage return, line feed, etc.) must be stripped by operators from all data received before processing such data. Processing includes all functions carried out by an operator once it receives any API call. For example, before performing a query the operator must strip all leading and trailing white space from the query keys. As another example, a business entity to be saved must first have all leading and trailing white space stripped from its fields.

As a matter of explanation, this requirement is intended to address challenges around predictably managing leading white space in terms of searching, name matching, and the like.   Removing any leading and trailing white space simplifies the ability to match data values in the registry.

Note that stripping of white space only occurs for element and attribute values. White space between tags must be handled properly per the XML specifications.

Example:

Assume for the purposes of this example that the element <desc> has a length limitation of 13. A save request includes the data

```
<desc>(space)(space)A sample foo string(space)(space)</desc>
```

where (space) denotes a single white space character. Embedded white space in this example is a single space character as well.

When the text is trimmed, the following results:

```
<desc>A sample foo string</desc>
```

Next, truncation to the required field length results in the following:

```
<desc>A sample foo(space)</desc>
```

Again, the trailing space is trimmed, and the string that is saved to the registry is:

```
<desc>A sample foo</desc>
```

4.2 Publication Restrictions

"Garbage" data within UDDI is defined to be useless data either intentionally or carelessly added by users via the publish API defined in the Programmers API Specification. This garbage can impact UDDI in two ways. The primary concern is that garbage not be permitted to collect at

such a rate as to actually shut down one or more UDDI operators. This could be, for example, a vast amount of data added to UDDI as a form of a denial of service attack upon the registry.

The secondary concern is that dense groups of garbage data not be permitted to cloud query results for specific entities. For example, a critical mass of data that appears to be related to the "Super Duper Widget Company" but in actuality is not could prevent legitimate users from discerning which of the many query results is the real "Super Duper Widget Company."

In order to make it more difficult for such garbage data to be created, publisher accounts are grouped into two categories, Tier 1 and Tier 2, each of which have different limits on the number of entities they may create. When users attempt to save an entity, which would exceed their limit, the operator must refuse the operation and return the error code E_accountLimitExceeded. See the UDDI Version 2.0 Programmer's Reference for more information.

4.2.1 Tier 1 Publisher Accounts

Each Tier 1 account at an Operator Node is restricted in the number of entities it is allowed to create by default. The default registration limits for these accounts at public UDDI Operator Nodes are listed in the table below. Note that private implementations may implement alternate limits, or may remove publication limits altogether.

Tier 1 Account Limits for Public Registries

Entity Type	Tier One Publishing Limit
businessEntity	1
businessServices per businessEntity	4
bindingTemplates per businessService	2
tModels	100
relationship assertions	10

4.2.2 Tier 2 Publisher Accounts

It is often the case that certain users (such as market makers, registrars, and large organizations) have a legitimate need to create more than the limit of entities placed on Tier 1 publisher accounts. To do so, such users must contact their operator to negotiate higher limits. An operator must verify the identity of the user requesting higher limits and must track the new limits of such users. Operators have the right to reject such requests for any reason.

4.3 Data Type Lengths

UDDI Operators are required to implement length restrictions on data stored in the registry. The UDDI Version 2.0 Data Structure Reference includes the maximum length for each element and attribute.

If an operator receives data longer than the lengths indicated in the UDDI Version 2.0 Data Structure Reference, it must truncate them to the maximum length before saving to the UDDI registry per the process described in section 4.1.1 above. Note that leading and trailing white space removal must occur before any data truncation. Responses to the request reflect the data actually saved to the registry, which may include elements that have been trimmed and truncated before they were stored.

As an additional comment, it should be noted that the truncation process might, in turn, generate additional trailing white space. This trailing white space must again be truncated before storing the information to the registry.

Implementation Note: Version 1.0 API requests must exhibit version 1.0 compliant behavior for all information in the UDDI registry - including that information saved through version 2.0 interfaces. This impacts two areas. Data exceeding the version 1.0 field size limitations must be properly truncated to the version 1.0 size limits before the response is returned to the caller. Second, additional elements and attributes introduced by the version 2.0 specification may not be returned in response to a version 1.0 request.

4.4 XML Validation

Each UDDI node operator is required to validate XML messages, and required restrictions on the data before processing. The following validation checks must be performed before performing a save_xxx operation, and before accepting replication data from other nodes. In all cases, if one of the errors listed below is detected in an inbound replication stream, the request may not be saved to the registry, and replication processing is stopped until the issue is resolved at the source operator.

4.4.1 Schema Validation

Each Node Operator must validate incoming data against both the API Replication, and Custody Transfer XML schemas as a part of all operations processing registry data. The error code returned in this case is E_fatalError. The error text will clearly indicate the nature of the problem processing the message.

4.4.2 XML Encoding

All XML messages must be encoded in UTF-8. The encoding attribute must be present in the message. If the encoding is incorrect, or not present, an error is generated.

The error code returned in this case is E_fatalError. The error text will clearly indicate the nature of the problem processing the message.

4.4.3 UUID Validation

The UUID must be in the format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX (8-4-4-4-12) where each character is a hexadecimal value from the set {A-F,a-f,0-9}. UUIDs are case insensitive (upper and lower case character values are equivalent). UUIDs for tModels must be prepended with the identifier 'uuid:' in all locations where they are present. The node operator must verify that these requirements are met when processing requests. Otherwise, an error is generated.

The error code returned in this case is E_fatalError. The error text will clearly indicate the nature of the problem processing the message.

4.4.4 UUID Reference Checking

All data references that contain UUID keys of other data in fact refer to data that actually exists in the data store. Specifically, this means that:

- every use of a tModelKey contains a value that in fact references a tModel which is found in the registry,

- every use of businessKey contains a value that in fact references a businessEntity which is found in the registry,

- every use of a serviceKey contains a value that references a businessService which is found in the registry,

- every use of a bindingKey contains a value that references a bindingTemplate which is found in the registry,

- every use of an operatorNodeID contains a value that in fact references an operatorNodeID found in the Replication Configuration File.

Of these, the first requirement is pragmatically the most significant, since tModels are often created, updated, and referenced independently from the other top level elements, whereas these other elements are typically created and updated together in the same location.

If UUID reference checks fail for any of the reasons listed above, the error code returned is E_invalidKey. The error text returned should include the UUID referenced, and where in the message it was included.

4.4.5 UUID Uniqueness

Each UUID stored must be verified to be unique. The Operator Node must not allow the generation or storage of a UUID that is already present somewhere in the registry. This applies to both the Operator Node where the entry was originally saved, and all other Operator Nodes where the entries are saved upon replication processing.

4.4.6 Checked Taxonomies and Identifiers

When saving new or updated registry entries with checked taxonomy or identifier values, these keys must be verified by each operator.

The error code returned in this case is E_invalidValue. The error text returned should include the taxonomy or identifier UUID and name, and the value included in the request that was detected to be in error.

4.4.7 Data Custody

Only the custodian Operator Node may originate change records that update or otherwise change the contents of data.

The error code returned in this case is E_userMismatch.

4.4.8 Field Length Restrictions

Each operator must verify field length restrictions before processing updates to the UDDI registry. On inbound replication, entries returned from other operators must be checked to verify that the required truncation of entries was completed before the information was saved to the originating operator.

On a save operation, any fields exceeding length restrictions established for the version of the UDDI specification indicated are truncated before they are saved to the registry. The response message contains the actual values saved to the registry. It is the responsibility of the client application to detect whether any of the data was truncated due to exceeding length restrictions.

On an inquiry operation, any fields exceeding length restrictions established for the version of the UDDI specification indicated are truncated before they are returned to the caller.

Violation of field length restrictions halts replication processing, but UDDI API requests truncate the data per the UDDI specifications.

4.4.9 White Space Removal

Each operator must verify leading and trailing white space removal before processing updates to the UDDI registry. On inbound replication, entries returned from other operators must be checked to verify that the required leading and trailing white space removal was completed before the information was saved to the originating operator.

Violation of the white space removal requirements halts replication processing, but UDDI API requests remove white space per the UDDI specifications.

5 Custody Transfer

The UDDI Business Registry contains entries representing service types and businesses. Each entry in the registry is associated with exactly one publisher account and node operator. There are a number of scenarios where the publisher may choose to transfer one or more of its registry entries to an alternate publisher account, and/or to another UDDI registry node operator.

The simpler case is that of transferring entries between two accounts that are registered at the same Operator Node. Usage scenarios for this type of custody transfer include the following:

5.1 Scenarios – Intra-operator Transfer:

- Change in employee responsible for the entry: The employee publisher account that is associated with a given registry entry is no longer responsible for making updates to the company's information. Responsibility needs to be transferred to a different individual.

- Businesses merged: Multiple businesses need to be consolidated under a single publisher account.

- Consolidation of registry entries: There are multiple entries for a given publisher or business, and are to be consolidated under a single publisher account.

The more complex case is the transfer of a set of entries across node operators, and by extension, across publisher accounts. All of the above scenarios apply, as well as those listed below.

5.2 Scenarios – Inter-operator Transfer:

Unsatisfactory service level: The functionality or service level provided by a given operator node is insufficient, and the publisher wishes to move their information to another site.

Change in availability for an operator node: An operator is no longer providing UDDI services, and all publisher accounts need to be migrated to an alternate.

Business Mergers, Acquisitions, or Consolidations: Changes in organizational structure may result in the need to make changes to the set of publisher accounts used to manage the registry entries.

For any of these scenarios, a consistent mechanism is provided to facilitate the transfer of information across publisher accounts or operator nodes.

5.3 Custody Transfer Considerations

It is expected that the entity custody transfer operation will not occur frequently. This activity is generally the result of employee or organizational changes, or consolidation of registry entries across publishers. An exceptional case is a change in the set of available registry operator sites. In this release, operation performance (in terms of time to complete the transfer) is not considered a high priority.

Public UDDI interfaces supporting custody transfer initiation will not be provided in this release of the UDDI specifications. The presence of these interfaces may introduce security and integrity issues that are not as prevalent with a browser-based manual initiation.

Only *businessEntity*, *tModel,* and related relationship assertion entries can be transferred between publisher accounts. When a businessEntity is transferred, all related business services and corresponding bindings are transferred as well. The Target Operator needs to take care to update change permissions on all sub-elements as a part of the transfer operation.

Complexity in the entity transfer process is introduced by the need to precisely correlate publisher accounts together across operator nodes[9]. During this process, explicit manual confirmation by the account owners is required.

Note that the Source Publisher (see below) is not required to transfer all of its entries in a single Custody Transfer request, nor is it required to transfer all of its entries to the same Target Publisher or Target Operator. Any combination or subset of UDDI registry entries may be transferred to any number of target publishers or Node Operators.

5.4 Transfer Execution

Transfer of information between operator nodes is a multi-step process requiring coordination between both operators and publishers. The process is initiated and confirmed manually by the publishers, with additional interaction between the operator nodes through both UDDI API and replication stream updates. The transfer operation is further secured through the out of band exchange of a shared secret between the source and target publishers.

The Node Operator may only respond to UDDI API requests related to custody transfer if they are initiated by a recognized UDDI Operator Node. This verification should be established through the same security mechanisms defined by the UDDI Version 2.0 Replication Specifications.

5.4.1 Role Definitions

The interaction diagram below includes the following roles. It describes the roles and communication between operators and publishers in completing a successful registry entry transfer.

Source Publisher: The individual originating the transfer request. This individual is identified with its corresponding publisher account at the Source Operator.

Source Operator: The custodian operator for the registry entities before the transfer operation is completed (the "from" operator node).

Target Publisher: The publisher account where the registry entities are being transferred to.

Target Operator: The target custodian operator for the registry entities (the "to" Operator Node).

---

[9] In the UDDI Version 2.0 timeframe, node operators do not share user account information. The transfer process facilitates sharing of sufficient account detail information to complete the operation.

All Operators: All the UDDI Operator Nodes except the source node.
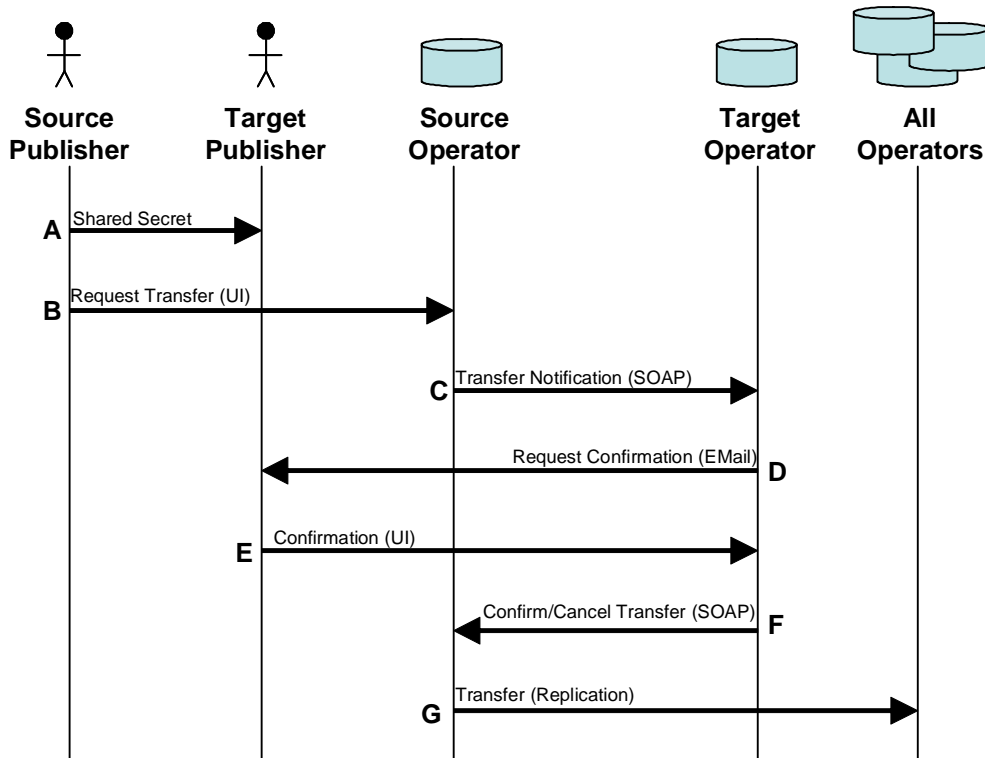
Figure 1: Custody Transfer Process

5.4.2 Custody Transfer Steps

Execution of the custody transfer operation requires multiple steps. The six primary steps are described in the following sections.

Step A: The *source publisher* provides a shared secret to the *destination publisher*. This shared secret acts as a verification mechanism to ensure that the destination publisher is indeed the desired individual. Before initiating the transfer operation, this information is exchanged through a private, out of band, mechanism such as a phone call, fax, secure email, or other means. It must not be shared with anyone other than the *target publisher* and *Source Operator*. When initiating the transfer operation at the source operator site, the source publisher will be required to provide this information.

Step B: The *source publisher* requests transfer of one or more of its registry entries. The publisher may request the transfer of *businessEntities* and/or *tModels*. Publisher assertions are transferred with *businessEntities.* The publisher may not request the transfer of *businessServices* or *bindingTemplates*, as these are contained within their parent *businessEntity*, and may not be separated.

Each Node Operator must provide a user interface feature to allow the publisher to manage the registry entries they have published. Publishers accessing this feature must be authenticated and secured using the same mechanism used to protect registry entries from unauthorized update or deletion. For each entity eligible for transfer (*businessEntity* or *tModel*), the publisher must be able to indicate which UDDI node operator and *destination publisher* account the entry is to be transferred to. The list of operators presented to the user must match the registration list

included within the UDDI Node Operator identifiers[10]. For each transfer group, the source publisher must provide a shared secret. This information is included in the *request_transfer* message passed to the *target operator.*

The user interface must require the confirmation of the transfer request before initiation of the transfer process. The confirmation prompt must indicate the *Target Operator*, *target publisher*, name[11] and key[12] for each registry entry to be transferred. The *shared* secret must not be displayed. The *source publisher* must also be given the option to cancel the transfer request at any time before **Step G** (this is described in more detail later).

Step C: Once the source publisher manually confirms the transfer request, the *Source Operator* makes a UDDI API request to the *Target Operator* to provide notification of the transfer request.

The *request_transfer* message is posted to the Target Operator, and includes the keys for the registry entries being transferred. Note that multiple keys may be transferred within a single transfer request.

Step D: Once the transfer request is received by the *Target Operator*, the transfer must be verified by the target publisher account holder. A confirmation email message is sent to the target publisher indicated in the *publisherEMail* element transmitted in the *request_transfer* message.

The confirmation email message sent to the target publisher account holder is as follows:

```
To: publisherEMail
Subject: UDDI Transfer Confirmation Request

The following UDDI entity transfer has been requested.  Please confirm the
transfer to the publisher account by following the link below and the
instructions provided there.

Transfer Group Key:

Transfer Group:
Key Type           Key Name               Key Value
(type)                     (name)                 (uuid)


…

Source UDDI Node Information
Publisher Name: authorizedName
Publisher Email: publisherEMail
Operator: sourceOperator

Target UDDI Node Information
Operator: targetOperator

Confirmation URL: URL
```

The values indicated in *italics* above are transmitted in the *request_transfer* API request. Those for Key Type (businessKey or tModelKey), Key Name (value of the description element), and Key Value (the UUID for the registry entry) are obtained directly from the UDDI registry. Each key should be listed along with its name value on individual lines in the message. The Target Operator generates the Confirmation URL in order to process the transfer request confirmation.

---

[10] A tModel has been established for the checked identifier set of UDDI Operator Nodes.

[11] The value in the <name> element for the entity to be transferred.

[12] The value of the businessKey or tModelKey attribute for the entity to be transferred.

Step E: Upon receipt of the transfer confirmation email message, the target publisher manually follows the embedded hyperlink (URL) provided in the *Confirmation* URL value to the Target Operator node. This link ultimately resolves to a page hosted at the Target Operator site where the publisher can complete (confirm or cancel) the transfer operation.

When the link is followed, the user must be precisely matched to a publisher account at the Target Operator site. If the publisher already has an account at the target node, they must be authenticated using that account information before confirming the transfer operation. If no account is present, they may be presented with an option to create a new account at transfer time. In either case, the publisher must be authenticated before being allowed to confirm the transfer.

Once authenticated, the *target publisher* must be prompted to enter the value of the *shared secret* as was provided by the *source publisher*. The shared secret values must match exactly. If the values do not match exactly, the *target publisher* may not be granted access to the transfer request information. Otherwise, the *target publisher* is presented with information regarding the transfer request, matching the original notification email message. The target publisher must be allowed to either confirm or cancel the transfer operation.

Note: Only five unsuccessful attempts to match the shared secret are allowed for any one custody transfer operation. On the fifth attempt, the target operator must cancel the transfer request by sending a cancel_transfer message to the source operator. An indication should be provided to the target publisher to indicate that they have exhausted the allowed number of attempts, and that the transfer operation must be re-initiated by the source publisher in order to proceed.

Step F: Confirmation of the request is reported back to the Source Operator through a *confirm_transfer* API message. If the transfer request is cancelled, a *cancel_transfer* API request is sent to the Source Operator. The syntax for both of these requests is included in the Appendix of this document.

Step G: This step applies when the transfer operation has been confirmed through the receipt of a *confirm_transfer* request at the Source Operator node. If the request has not been cancelled by the source publisher, the Source Operator relinquishes custody of the registry entry by creating a *custodyTransfer* replication journal entry. Once this entry is created, custody transfer is released, and all subsequent updates must be processed at the target UDDI registry site.

When this journal entry is processed by the Target Operator, the values of the operator and authorizedName attributes are changed to reflect the information at the Target site. Additionally, the value of the discoveryURL element for the businessEntity useType is updated to reflect the new URL used to access the XML representation of the businessEntity at the new operator site. Note that multiple discoveryURL elements may be present in the business entity. The Target Operator may only change the entry for the discoveryURL for the useType of businessEntity.

5.4.3 Custody Transfer Cancellation

The custody transfer operation can be cancelled at any time before the *custodyTransfer* journal entry is created.

The source publisher may return to the Source Operator site and cancel the operation through the user interface. The UDDI node operator must provide this functionality as a part of the custody transfer functionality. If the operation is cancelled at the source site, the corresponding *confirm_transfer* request that may be sent by the target operator results in the E_transferAborted Fault being returned.

If the target publisher does not accept the custody transfer, a *cancel_transfer* message is transmitted to the Source Operator site, and custody remains with the original publisher.

Note that cancellation of the transfer request applies to the entire transfer group. The set of keys included in a transfer group are processed as a single batch.

5.4.4 Entity Change Restrictions

The publisher may make changes to the entity to be transferred at any time before the *confirm_transfer* message is received at the Source Operator. Any changes made before the *confirm_transfer* API message is received and processed by the Source Operator node will be correctly handled by the standard replication process. Once the *custodyTransfer* replication journal entry is created, no additional updates are allowed at the Source Operator, and custody is formally relinquished to the Target Operator.

Note: There is a small time window when the transferred entity may not be updated at any operator site. This occurs for the period between when the Source Operator creates the *custodyTransfer* entry in the replication stream, and when the registry entry is processed at the target site and the new publisher account is associated with the registry entry.

5.5 Operator Access Point Registration

Each UDDI Node Operator must register the access point for the Custody Transfer Protocol tModel. To protect against Denial of Service attacks, or unauthorized attempts to access the service, the **hostingRedirector** functionality may be used as desired by the Node Operator.

6 UUID Algorithm

Heavy use is made of UUIDs within UDDI. Operators are responsible for creating these UUIDs and must make certain that the process of creating these is a correct one. For versions 1.0 and 2.0 of the UDDI specification, operators are required to use the DCE UUID generation algorithm as defined below. This section describes the accepted UUID generation algorithms.

6.1 UUID Generation Algorithms

A number of algorithms for generating UUIDs are described in the following reference.

   http://www.ics.uci.edu/pub/ietf/webdav/uuid-guid/draft-leach-uuids-guids-01.txt[13]

Implementation of these algorithms has been approved for use within UDDI Node Operator implementations. Additionally, careful consideration must be made in determining which algorithm is selected for a given implementation, and that the random number-based approach includes generation of truly random numbers.

The UDDI operators reserve the right to review the algorithm used to generate UUIDs as implemented by a new public UDDI Node Operator, as well as a large sample set of values generated by the algorithm, prior to the public site becoming part of the production cloud.

It is also important to note that a number of checks on the GUID generation and formatting are required by the public UDDI Operator Nodes. These verification steps are outlined in section 4.4 of this document.

---

[13] This document will also be available for download from the uddi.org web site.

7 Operator Web User Interface

Each operator is required to provide a HTML 3.2 based user interface on the Internet for use by all UDDI users. This may also include client scripting (e.g., JavaScript) for the purposes of client validation, as may be required by each individual Operator Node. These interfaces must provide the ability to perform the following activities:

Establishing User Credentials: This user interface will allow an individual or business (a publisher) to establish a login and password on a particular UDDI operator node.

Account Activation: This interface allows a publisher to enable their account at a given Operator Node.

Changing Ownership of Business Entities and tModels: This user interface will allow an owner of Business Entities and/or tModels to transfer ownership to another publisher account. Details on the custody transfer process are provided in section 5.

Contacting the Administrator: This interface allows anyone to contact the administrator of the UDDI Operator Node.

7.1 Establishing User Credentials

Before publishers are allowed to register information at an operator node, they need to establish an account with the operator. The process for this is operator-specific but must meet the minimum criteria explained here.

It is essential that there is a mechanism to identify a name and email of a contact for any entity that registers business entities in UDDI. Although this data is not reflected in the schema provided by UDDI, it should be stored and obtainable by an operator for business entities mastered in their specific operator instance. This information may not be left blank in the registration – valid values must be obtained through verification of the email address (and a response received via email from that account). User information may not be shared publicly, or with the other Operator Nodes[14]. The only account information that is visible outside a given Operator is the public identifier for the publisher.

An operator is required to collect this information via an HTML 3.2 user interface on the operator's UDDI web site. An operator must also provide a means via this HTML user interface to update the user's credentials (to change a password, for example).

Each operator must obtain an indication of acknowledgment of the operator policies by the registrant for audit purposes. Section 3.3 of the UDDI Operators Agreement identifies non-compliance concerns and operator responsibilities for complying with Third Party audits.

7.2 Account Activation

Although the user registers with the operator and establishes an account at the operator node, the account itself is not activated until the user completes the activation procedure. To require an account to be "activated" means that all UDDI APIs that require authentication will fail until the account is activated.

For the account to be activated, the operator must email the user instructions in plain text that identifies a website for the user to visit to actually activate the account. The email must describe clearly the purpose for this activation and how to perform it. The activation website may require no more than that the user supply the account's authentication credentials in order to activate the account.

---

[14] With the exception of the information required to execute the Custody Transfer protocol.

7.3 Changing Entity Ownership

A UDDI Node Operator must provide an HTML 3.2 user interface to permit a user to transfer ownership of data it currently owns to another user. The protocol for transfer of registry entries is fully described in Section 5 of this document.

7.4 Contacting the Administrator

A UDDI Node Operator must provide a mechanism to contact the site administrator via Email.

7.5 Use of SSL

Communication with the UDDI Node Operator User Interface must be secured via SSL all operations where passwords, security information, or registry entry changes are communicated, as well as all authenticated operations on registry data. This includes, but is not limited to, the following items:

- Publisher account creation and maintenance

- Registry entry management

- Establishing relationship assertions

- Initiating, confirming, or canceling custody transfer operations.

8 Reference Materials

This document refers to other UDDI documents as well as several widely recognized standards and specifications. Detailed information on these may be found as follows:

8.1 UDDI Specifications and Documents

UDDI Version 2.0 Operator's Specification (this document)

http://www.uddi.org/pubs/Operators_v2.htm

UDDI Version 2.0 Custody Transfer Schema

http://www.uddi.org/schema/uddi_v2custody.xsd

UDDI Version 2.0 Programmer's API Specification

http://www.uddi.org/pubs/ProgrammersAPI_v2.htm

UDDI Version 2.0 API Schema

http://www.uddi.org/schema/uddi_v2.xsd

UDDI Version 2.0 Replication Specification

http://www.uddi.org/pubs/Replication_v2.htm

UDDI Version 2.0 Replication Schema

http://www.uddi.org/schema/uddi_v2replication.xsd

UUID Generation Algorithms

http://www.uddi.org/pubs/draft-leach-uuids-guids-01.txt

8.2 Standards and Other Specifications

HTML 3.2

http://www.w3.org/TR/REC-html32

HTTP 1.1

http://www.w3.org/Protocols/HTTP/1.1/rfc2616.pdf

XML

http://www.w3.org/TR/1998/REC-xml-19980210.html

XML Schema

http://www.w3.org/TR/2001/PR-xmlschema-0-20010330
http://www.w3.org/TR/2001/PR-xmlschema-1-20010330
http://www.w3.org/TR/2001/PR-xmlschema-2-20010330

SOAP 1.1

http://www.w3.org/TR/2000/NOTE-SOAP-20000508

SSL

http://www.ietf.org/rfc/rfc2246
http://www.ietf.org/rfc/rfc2818
http://www.ietf.org/rfc/rfc2817

X.509 V3 Certificate

http://www.ietf.org/rfc/rfc2459

Appendix A: External Validation Services for Taxonomies and Identifiers

This appendix describes the UDDI Node Operator requirements related to support of external taxonomy and identifier validation services. Details of the protocol and supporting interfaces are described in the UDDI Version 2.0 Programmer's API Specification. Throughout this section, any requirements for validating taxonomy values also apply to validating identifier values.

A.1 Registration and Approval Considerations

- A.1.1: Node Operators must offer prospective providers of external validation services a mechanism for seeking approval and activation of the service. The providers of external validation services are required to provide the access point of the validation service, tModel information for the taxonomy or identifier system, and whether they support operator caching of valid values.

- A.1.2: To initiate the service approval process, the Node Operator holding custody of the taxonomy or identifier system's tModel registration must perform the following steps in order:

  1. Verify that the taxonomy or identifier system tModel is not registered. If it is, approval is denied.

  2. Notify the other operators of this potential validating taxonomy or identifier system.

- A.1.3: The Node Operator publishes the tModel for the taxonomy or identifier system, categorizing it as *unvalidatable* within the UDDI tModel types taxonomy.

- A.1.4: The operator performs validation tests as appropriate, or as prescribed by the set of Node Operators in the replication cloud, to determine whether the validation service should be approved. The individual operator may perform additional tasks to determine whether the validation service should be activated.

- A.1.5: The operator informs the other UDDI node operators in the replication cloud of the intent to approve the validation service.

- A.1.6: The operator publishes a bindingTemplate under its validation businessService in its operational businessEntity (see below) for the approved validation service.

  A hosting redirector is used to pair this bindingKey with the access point and caching option provided by the service provider. Redirection is usually handled by the operator, but can be delegated to a redirector service provided by someone else, often the taxonomy provider.

  The tModelKey of the taxonomy and the tModelKey of the uddi-org:taxonomy tModel must be placed into the approved service binding's tModelInstanceInfo structure. This enables the validation service for the particular taxonomy to be found by all of the operators.

  An operator must place all of the bindingTemplates for the same service (those sharing the same specification tModel) under the same businessService.

- A.1.7: The operator sets up the redirector pairing of the previous bindingTemplate with the access point provided by the service provider.

- A.1.8: · The operator transfers ownership of the taxonomy or identifier system tModel to the provider of the taxonomy or identifier system and provides notification that the taxonomy or identifier system and its validation service can be activated by removing the unvalidatable categorization..

A.2   Save_xxx API Considerations

Save behavior described by this section applies whenever an entity is saved which has an associated identifierBag or categoryBag.

- A.2.1: Validation is performed for each entity to be saved to the registry.

- A.2.2: A save of new keyedReferences associated with an unvalidatable tModel or one which is not available should fail with E_unvalidatable. Existing keyedReferences associated with an unvalidatable tModel are to be considered still valid and should not be re-checked.

- A.2.3: The approved validation service bindingTemplate associated with a given tModel is found in one of the businessEntitys that are identified with a uddi-org:operators identity. A bindingTemplate for a particular taxonomy has tModelKeys for the taxonomy and for the uddi-org:taxonomy tModel.

- A.2.4: A non-validating external taxonomy is recognized by the absence of an approved validation service, and the absence of the unvalidatable categorization on the taxonomy's tModel.

- A.2.5: Valid values that are obtained from an external validating taxonomy that is cacheable may be cached by an operator, thereby necessitating calls to the external validation service only when a referenced keyValue is not found in the cache.

- A.2.6: The bindingDetail that the hostingRedirector yields must contain the accessPoint of the service, and in the instanceParms, whether the taxonomy is cacheable (uddi-org:cacheable) or not (uddi-org:notCacheable).

- A.2.7: An error condition returned by the validation service results in the failure of the save function. Errors must be returned to the invoker of the save operation.

A.3   Other Considerations

- A.3.1: Each operator must publish an operational businessEntity with which call-out services such as category and identity validation are associated. The operational businessEntity has a reference to the operator's identity associated with the uddi-org:operators identifier system.

- A.3.2: Operators must deny use of the uddi-org:operators identifier for all saves except for their own operational businessEntity. This prevents other publishers from declaring themselves operators.

- A.3.3: Built-in validating taxonomies do not have to conform to this specification. Operators may add validation services for the built-in taxonomies to their list of approved validation services, but this isn't required.

- A.3.4: Node Operators may collectively and individually take a taxonomy off-line when an external validation service doesn't meet performance expectations. Policies for how and when this is done is up to the other UDDI Node Operators participating in the replication cloud, but some technical options available for taking a validation service offline include:

    Breaking the path to the accessPoint in the hostingRedirector service.

    Asking the validating taxonomy provider to categorize its tModel as unvalidatable. After a replication cycle, attempts to categorize with this taxonomy

will thus be rejected until it is made available again. Existing references to this taxonomy remain valid in subsequent saves.

Having the custodial operator mark the taxonomy's tModel as unvalidatable. This has the same effect as the option above.

Removing the approved validation service from the custodial operator's operational businessEntity. This is a last resort for rogue providers since this has the affect of making the taxonomy or identifier system non-validating. If the tModel has been referenced at all, subsequent attempts to gain approval for another validation service for the tModel will fail.

Appendix B: request_transfer (SOAP API)

The source UDDI node operator posts a request to the Target Operator to initiate the custody transfer operation. The *request_transfer* message includes the key to be transferred, operator identifiers, and information about the publisher account

Syntax:

```
<request_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
   <authInfo/>
<authorizedName/>
<sharedSecret/>
<targetEMail/>
<sourceOperator/>
<targetOperator/>
<requestExpiration/>
<transferKey/>
<transferGroup>
   <registryKey businessKey | tModelKey />…
</transferGroup>
</request_transfer>
```

Arguments:

- **authInfo:** this required element is an element that contains an authentication token. Authentication tokens are obtained through the *get_authToken* API call.

- **authorizedName:** The public identifier for the source publisher account. This information is typically included in the authorizedName attribute for public API requests.

- **sharedSecret:** The confidential confirmation code that must be provided by the target publisher in order to obtain access to the transfer request information at the target operator site.

- **targetEMail:** The email address for the target publisher. This email address is used to communicate with the publisher receiving the information and to confirm the transfer request.

- **sourceOperator:** The public identifier for the Source Operator node. This information is typically included in the operator attribute for public API requests.

- **targetOperator:** The public identifier for the Target Operator node. This information is typically included in the operator attribute for public API requests.

- **requestExpiration:** The time that the transfer request expires. The Source Operator must receive notification of the completed transfer through a confirm_transfer message before this time, or the transfer operation is automatically cancelled. The value for this element is to be 72 hours from the original request time.

- **transferKey:** The UUID of the transfer operation. This is the unique identifier of the custody transfer that is assigned to the set of keys in the transfer group. All subsequent custody transfer operations refer to this key.

- **transferGroup:** The set of keys (businessEntity or tModel) to be transferred as part of the operation.

- **registryKey:** The registry key to be transferred between publisher accounts and/or operator nodes.

Example:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Envelope xmlns=http://schemas.xmlsoap.org/soap/envelope/>
  <Body>
    <request_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
    <api:authInfo>---authorization token here---</api:authInfo>
<authorizedName>sourceID@company.com</authorizedName>
<sharedSecret>my dog has fleas and the weather is warm today.   enter this
secret information in order to access the custody transfer
information.</sharedSecret>
<targetEMail>targetID@domainname.com</targetEMail>
<sourceOperator>Source</sourceOperator>
<targetOperator>Target</targetOperator>
<transferKey>12345678-1234-1234-1234-12345678012</transferKey><transferGroup>
  <registryKey
  businessKey="12345678-1234-1234-1234-123456789012"/>
  <registryKey
  tModelKey="uuid:12345678-1234-1234-1234-123456789012"/>
</transferGroup>
    </request_transfer>
  </Body>
</Envelope>
```

Behavior:

Note: The Operator Node may only respond to requests made by other recognized operators. Otherwise, an E_requestDenied Fault must be returned.

Returns:

On success, this operation returns a SOAP Fault with the error number set to E_success.

Caveats:

If any error occurs in processing this message, a dispositionReport structure will be returned to the caller in a SOAP Fault. The following error number information will be relevant:

E_authTokenExpired: signifies that the authentication token value passed in the authInfo argument is no longer valid because the token has expired.

E_authTokenRequired: signifies that the authentication token value passed in the authInfo argument is either missing or is not valid.

E_invalidKeyPassed: This error is returned when either the key does not exist, is of the wrong type.

E_requestDenied: This error is returned if the Target Operator does not accept the custody transfer, or the requestor is not an authorized Operator Node.

E_unsupported: This error is returned if the custody transfer protocol is not implemented by the Target Operator.

E_userMismatch: signifies that the UUID key value passed in the request refers to data that is not controlled by the individual who is represented in the authentication token.

Appendix C: confirm_transfer (SOAP API)

When a custody transfer operation is confirmed, the Source Operator is notified of the result of the operation through a *confirm_transfer* request. This request is triggered by the manual action of the target publisher at the Target Operator web site.

Syntax:

```
<confirm_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
   <authInfo/>
<authorizedName/>
<sourceOperator/>
<targetOperator/>
<transferKey/>
</confirm_transfer>
```

Arguments:

- **authInfo:** this required element is an element that contains an authentication token. Authentication tokens are obtained through the *get_authToken* API call.

- **authorizedName:** The public identifier for the target publisher account. This information is the new value to be included in the authorizedName attribute for public API requests.

- **sourceOperator:** The public identifier for the Source Operator node. This information is typically included in the operator attribute for public API requests.

- **targetOperator:** The public identifier for the Target Operator node. This information is typically the new value to be included in the operator attribute for public API requests.

- **transferKey:** The UUID of the transfer operation. This is the unique identifier of the custody transfer that is assigned to the set of keys in the transfer group.

Note that this message is similar to the *request_transfer* message, in that the values of the elements transmitted are the same, with the exception of the *authorizedName*. The *authorizedName* is changed to the new account identifier at the Target Operator site.

The email address of the publisher at the target site is not shared with the Source Operator node.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Envelope xmlns=http://schemas.xmlsoap.org/soap/envelope/>
  <Body>
    <confirm_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
      <authInfo>---authorization token here---</authInfo>
<authorizedName>newpublisher12345</authorizedName>
<sourceOperator>Source</sourceOperator>
<targetOperator>Target</targetOperator>
<transferKey>12345678-1234-1234-1234-123456789012</transferKey>
    </confirm_transfer>
  </Body>
</Envelope>
```

Behavior:

When received by the Source Operator, this message serves as notification to change the registry entry and broadcast the update through the replication stream. This message triggers a *custody_transfer* journal entry to be created for the entry. All subsequent changes to this entry are made at the destination operator site.

The operation must be successful for all keys in the *transferGroup*. Otherwise, a SOAP Fault is generated.

Note: The Operator Node may only respond to requests made by other recognized operators. Otherwise, an E_requestDenied Fault must be returned.

Returns:

On success, this operation returns a SOAP Fault with the error number set to E_success.

Caveats:

If any error occurs in processing this message, a dispositionReport structure will be returned to the caller in a SOAP Fault. The following error number information will be relevant:

> **E_authTokenExpired:** signifies that the authentication token value passed in the authInfo argument is no longer valid because the token has expired.

> **E_authTokenRequired:** signifies that the authentication token value passed in the authInfo argument is either missing or is not valid.

> **E_invalidKeyPassed:** This error is returned when either the key does not exist, is of the wrong type.

> **E_requestDenied:** This error is returned if the Target Operator does not accept the custody transfer, or the requestor is not an authorized Operator Node.

> **E_transferAborted:** signifies that the source publisher cancelled the transfer request. The custody transfer will not be completed.

> **E_userMismatch:** signifies that the UUID key value in the passed in the transferKey refers to a transfer group that is not available to the target operator who is represented in the authentication token.

Appendix D: cancel_transfer (SOAP API)

cancel_transfer (SOAP API)

When a custody transfer operation is cancelled, the Source Operator is notified of the result of the operation through a *cancel_transfer* request. This request is triggered by the manual action of the target publisher at the Target Operator web site.

Syntax:

```
<cancel_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
   <authInfo/>
<authorizedName/>
<sourceOperator/>
<targetOperator/>
<transferKey/>
<errInfo/>
</cancel_transfer>
```

Arguments:

- authInfo: this required element is an element that contains an authentication token. Authentication tokens are obtained through the *get_authToken* API call.

- authorizedName: The public identifier for the target publisher account. This information is the new value to be included in the authorizedName attribute for public API requests.

- sourceOperator: The public identifier for the Source Operator node. This information is typically included in the operator attribute for public API requests.

- targetOperator: The public identifier for the Target Operator node. This information is typically the new value to be included in the operator attribute for public API requests.

- transferKey: The UUID of the transfer operation. This is the unique identifier of the custody transfer that is assigned to the set of keys in the transfer group.

- errInfo: Processing details providing more information on why the operation was cancelled. This element has the same structure as the error disposition reports described in the UDDI Version 2.0 Programmers API Specification.

Note that this message is similar to the *request_transfer* message, in that the values of the elements transmitted are the same.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Envelope xmlns=http://schemas.xmlsoap.org/soap/envelope/>
  <Body>
    <cancel_transfer generic="2.0" xmlns="urn:uddi-org:custody_v2">
    <authInfo>---authorization token here---</authInfo>
<authorizedName>newpublisher12345</authorizedName>
<sourceOperator>Source</sourceOperator>
<targetOperator>Target</targetOperator>
<transferKey>12345678-1234-1234-1234-123456789012</transferKey>
<errInfo>…</errInfo>
    </cancel_transfer>
  </Body>
</Envelope>
```

Behavior:

When this request is received by the Source Operator, the pending custody transfer request is cancelled. The publisher may no longer confirm the transfer at the Target Operator site, or cancel the request again at the source. Any subsequent operations on the listed entities must be reinitiated at the source UDDI registry node.

The errInfo element must be populated will additional details regarding the reason for the transfer cancellation. The value of the errCode attribute must be one of the following:

- **E_publisherCancelled:** The target publisher cancelled the custody transfer operation.

- **E_secretUnknown:** The target publisher was unable to match the shared secret, and the five (5) attempt limit was exhausted. The target operator automatically cancelled the transfer operation.

Note: The Operator Node may only respond to requests made by other recognized operators. Otherwise, an E_requestDenied Fault must be returned.

Returns:

On success, this operation returns a SOAP Fault with the error number set to E_success.

Caveats:

If any error occurs in processing this message, a dispositionReport structure will be returned to the caller in a SOAP Fault. The following error number information will be relevant:

E_authTokenExpired: signifies that the authentication token value passed in the authInfo argument is no longer valid because the token has expired.

E_authTokenRequired: signifies that the authentication token value passed in the authInfo argument is either missing or is not valid.

E_invalidKeyPassed: This error is returned when either the key does not exist, is of the wring type.

E_requestDenied: This error is returned if the Target Operator does not accept the custody transfer, or the requestor is not an authorized Operator Node.

E_userMismatch: signifies that the UUID key value in the passed in the transferKey refers to a transfer group that is not available to the target operator who is represented in the authentication token.

Appendix E: Element Length Restrictions

The table below describes the data types and maximum length restrictions for each element used in the custody transfer protocol. Note that many of these elements have been previously been described in the UDDI Version 2.0 API Specification. The name of the element reused from the API specification.

Name	Data Type and Length	Reference
authInfo		see UDDI API Specification (authInfo)
authorizedName		see UDDI API Specification (authorizedName)
sharedSecret	string (255)	
sourceEmail	string (255)	
targetEmail	string (255)	
sourceOperator		see UDDI API Specification (operator)
targetOperator		see UDDI API Specification (operator)
businessKey		see UDDI API Specification (businessKey)
tModelKey		see UDDI API Specification (tModelKey)
requestExpiration	timeInstant	
errInfo		see UDDI API Specification (errInfo)
errCode		see UDDI API Specification (errCode)

Appendix F: Custody Transfer tModel

The Core tModel for the Custody Transfer protocol is as follows:

```
<tModel
  tModelKey="uuid:578B6EEE-9822-49E6-963A-3C03B279A7C0">
  <name>uddi-org:custody-transfer:2-0</name>
  <description xml:lang="en">UDDI Custody Transfer API Version 2.0</description>
  <overviewDoc>
    <description xml:lang="en">This tModel defines the custody transfer API
calls for the UDDI registry.</description>
    <overviewURL>http://www.uddi.org/pubs/operations-V2.00-
20010228.html</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
      keyName="types"
      keyValue="soapSpec"
      tModelKey="uuid:C1ACF26D-9672-4404-9D70-39B756E62AB4"/>
    <keyedReference
      keyName="types"
      keyValue="specification"
      tModelKey="uuid:C1ACF26D-9672-4404-9D70-39B756E62AB4"/>
    <keyedReference
      keyName="types"
      keyValue="xmlSpec"
      tModelKey="uuid:C1ACF26D-9672-4404-9D70-39B756E62AB4"/>
  </categoryBag>
</tModel>
```

Appendix G: Notices

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.

These UDDI Specifications (the "Documents") are provided by the companies named above ("Licensors") under the following license. By using and/or copying this Document, or the Document from which this statement is linked, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions:

Permission to copy, prepare derivative works based on, and distribute the contents of this Document, or the Document from which this statement is linked, and derivative works thereof, in any medium for any purpose and without fee or royalty under copyrights is hereby granted, provided that you include the following on ALL copies of the document, or portions thereof, that you use:

1. A link to the original document posted on uddi.org.

2. An attribution statement : "Copyright © 2000 - 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."

If the Licensors own any patents or patent applications that may be required for implementing and using the specifications contained in the Document in products that comply with the specifications, upon written request, a non-exclusive license under such patents shall be granted on reasonable and non-discriminatory terms.

EXCEPT TO THE EXTENT PROHIBITED BY LOCAL LAW, THIS DOCUMENT (OR THE DOCUMENT TO WHICH THIS STATEMENT IS LINKED) IS PROVIDED "AS IS," AND LICENSORS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF THE INFORMATIONAL CONTENT, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY OR (WITH THE EXCEPTION OF THE RELEVANT PATENT LICENSE RIGHTS ACTUALLY GRANTED UNDER THE PRIOR PARAGRAPH) LICENSOR PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS. Some jurisdictions do not allow exclusions of implied warranties or conditions, so the above exclusion may not apply to you to the extent prohibited by local laws. You may have other rights that vary from country to country, state to state, or province to province.

EXCEPT TO THE EXTENT PROHIBITED BY LOCAL LAW, LICENSORS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL DAMAGES, OR OTHER DAMAGES (INCLUDING LOST PROFIT, LOST DATA, OR DOWNTIME COSTS), ARISING OUT OF ANY USE, INABILITY TO USE, OR THE RESULTS OF USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF, WHETHER BASED IN WARRANTY, CONTRACT, TORT, OR OTHER LEGAL THEORY, AND WHETHER OR NOT ANY LICENSOR WAS ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Some jurisdictions do not allow the exclusion or limitation of liability for incidental or consequential damages, so the above limitation may not apply to you to the extent prohibited by local laws.

Copyright © OASIS Open 2002-2003. *All Rights Reserved.*