

Registry Services specification: version 1.0.beta

Neel Smith, August, 2005

Licensed under the Creative Commons Attribution-NonCommercial-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/1.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Contents

Preface.....	1
Versions.....	1
Requirements for client-server communication.....	1
Common parameters.....	3
The Registry service requests.....	3
The Registry Schemas.....	7

Preface

This document defines version *1.0.beta* of the Registry Services implementation specification. The specification defines interaction between a client and server providing standard identifiers for entities in some specified domain. For an overview of Registry services, see <http://chs75.harvard.edu/projects/diginc/techpub/registry>.

Versions

Release versions are identified by two dot-separated integers, with the first being most significant. E.g., "2.0" is greater than "1.10" but "1.10" is greater than "1.2".

Experimental, pre-release versions may also be made available from time to time. An identifier for a pre-release version is composed of a regular release-version number followed by a string indicating its test status (e.g., "1.2.beta" or "1.2.release-candidate"). Optionally, the identifier for a test version may have a further positive integer indicating its sequence in that test series (e.g., "1.2.beta.2").

A version number is a required attribute on the RegistryService element returned by a GetCapabilities request (defined below).

Requirements for client-server communication

Registry Services use the HyperText Transfer Protocol [IETF RFC 2616] for communication between client and server. An instance of Registry Services referred to hereafter as a Registry's "base URL" is an HTTP Uniform Resource Locator (URL). This URL is not required to return any content, although it is strongly recommended that a request for a Registry URL return basic information identifying the Registry in either text-only or XHTML format.

The Registry's base URL serves as a URL prefix for Registry requests: Registry requests

are formed by adding request parameters directly to the base request URL. More formally, the concatenation of the base request URL with URL parameters must produce a string forming a valid URL according to the requirements of the URL specification [IETF RFC 2396] and the HTTP Common Gateway Interface standard (CGI). Each parameter's name and value pair must be separated from any following parameters by an ampersand ("&", Unicode x0026); name and value for each individual parameter must be separated by an equals sign ("=", Unicode x003D). . All Registry requests include a URL parameter named `request`. The value of this parameter must be the name of a request defined in version 1.0.beta. For some requests, one or more additional parameters are required as defined in this specification. If any of the characters "/", "&", or "=" appear in the value of a parameter, they must therefore be encoded as defined in IETF RFC 2616. In version 1.0.beta, Registry services require use of the GET method for all requests.

Examples of `GetCapabilities` requests formed from valid CTS base URLs:
`http://myhost/myregistry?` is a valid base URL because the request string
`http://myhost/myregistry?request=GetCapabilities` is a valid URL.
`http://myhost/myregistry?configuration=default&` is also a valid base URL
because the request
`http://myhost/myregistry?configuration=default&request=GetCapabilities`
is a valid URL.

In addition, CTS reserves a special meaning for the period character (".", Unicode x002E). When a period appears in an `id` parameter, it separates hierarchical components of a registry entry. If a period instead represents a character in the reference value, it must be URL escaped. Although the specification allows periods and hyphens to appear as literal values for `id` parameters, registrars managing a Registry services instance are strongly discouraged from

including periods in the literal values of individual components of a registry id value.

Parameter names and values are both case sensitive. Order of parameters in a request is not significant.

Common parameters

The following parameters are allowed or required in multiple Registry requests:

- `registryID` • The identifier of a Registry, required to be unique within the scope of the Registry inventory returned by the `GetCapabilities` request.
- `ID` • The unique identifier of an individual entry in a registry's hierarchical reference scheme. The ID value of an entry is a concatenation of one or more strings taken from the `id` attributes of successively contained elements, with each component except the last joined to following components by a period (".", Unicode x002E).

The Registry service requests

Registry Services requires implementation of six requests. These requests are named `GetCapabilities`, `GetValidValues`, `GetRedirects`, `QueryRegistry` and `DownloadRegistry`, and include required and allowed parameters as defined in the following sections.

Except for the `DownloadRegistry` request, the reply to a Registry request is always a well-formed XML document with a root element having the same name as the request. The reply document always contains exactly two elements, the first of which is named `request`. The

format of the `request` element is unspecified, and may be used for any purpose the implementation wishes. As the name of the element is intended to imply, it is strongly recommended that implementing software use this element to report back parameters passed in with the request. Developers may also use this element for other purposes such as embedding debugging information in a reply that complies with the CTS specification.

If the syntax and contents of request parameters do not fully comply with the specifications in this document, the second element of the reply is named `error`; as of version 1.0.beta of the Registry specification, the format and content of error reporting are left up to the implementing software. Future versions of the protocol may define structures for error reporting.

If the syntax and contents of request parameters fully comply with the specifications in this document, then the name and format of the second element vary according to the request, and are defined in the following sections, together with the Relax NG schemas listed in the appendix.

GetCapabilities

Purpose: the `GetCapabilities` request retrieves an inventory or registries managed by an instance of Registry services.

Parameters: no parameters other than the `request` parameter are required, but as with all Registry requests, a request may optionally include a `version` parameter indicating the version of the protocol preferred by the client. As of version 1.0.beta, servers are not required to recognize or alter their reply in response to a `version` parameter.

<i>Parameter</i>	<i>Required/optional</i>	<i>Description</i>
<code>version</code>	optional	Client may indicate preferred version of Registry protocol

Response: validates against `GetCapabilities.rng` (see appendix).

GetValidValues

Purpose: the `GetValidValues` request retrieves a list of all unique index values satisfying an XPath expression for a given registry. If the `query` parameter is omitted, `GetValidValues` returns a list of unique values at the top level of the Registry's ID hierarchy; that is, the result is equivalent to a request with the `query` parameter equal to `/`.

<i>Parameter</i>	<i>Required/optional</i>	<i>Description</i>
<code>registryID</code>	required	Unique ID for a specific registry
<code>query</code>	optional	XPath expression limiting scope of the query
<code>version</code>	optional	Client may indicate preferred version of Registry protocol

Response: validates against `GetValidValues.rng` (see appendix).

DownloadRegistry

Purpose: the `DownloadRegistry` request retrieves the complete contents of a given registry.

<i>Parameter</i>	<i>Required/optional</i>	<i>Description</i>
<code>registryID</code>	required	Unique ID for a specific registry
<code>version</code>	optional	Client may indicate preferred version of Registry protocol

Response: validates against `Registry.rng` (see appendix).

GetRedirects

Purpose: the `GetRedirects` request retrieves all identifier records that have been redirected to a given ID.

<i>Parameter</i>	<i>Required/optional</i>	<i>Description</i>
<code>registryID</code>	required	Unique ID for a specific registry
<code>ID</code>	required	Unique ID for a registry entry, expressed in dot-separated format for entries deeper than the top level of the registry hierarchy.
<code>version</code>	optional	Client may indicate preferred version of Registry protocol

Response: validates against `GetRedirects.rng` (see appendix).

QueryRegistry

Purpose: the `QueryRegistry` request may be used *either* to retrieve an identifier record for a specific ID value, if the optional `ID` parameter is include; *or* to search the description element of an identifier record if the optional `description` parameter is included. One and only one of `ID` and `description` is required.

<i>Parameter</i>	<i>Required/optional</i>	<i>Description</i>
------------------	--------------------------	--------------------

Licensed under the Creative Commons Attribution-NonCommercial-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/1.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

registryID	required	Unique ID for a specific registry
ID	one of ID and description is required	Unique ID for a registry entry, expressed in dot-separated format for entries deeper than the top level of the registry hierarchy.
description	one of ID and description is required	string to search for in entry description
querytype	may optionally be used along with description parameter	a value of wholeword indicates that the search should only return descriptions where description appears as a word bounded by white space or punctuation; a value of substring indicates that the search should return all descriptions containing the string description.
version	optional	Client may indicate preferred version of Registry protocol

Response: validates against QueryRegistry.rng (see appendix).

The Registry Schemas

The following list of Relax NG schemas define the syntax of Registry service replies, and are included by reference as part of the Registry Services specification.

- GetCapabilities.rng
- GetValidValues.rng
- Registry.rng
- GetRedirects.rng
- QueryRegistry.rng