



# ebXML RegRep Profile for XML Schema Version 4.0

## Committee Specification Draft 01-2

25 January 2012

### Specification URIs

#### This version:

<http://docs.oasis-open.org/regrep/regrep-profile-xsd/v4.0/csd01/regrep-profile-xsd.html>  
<http://docs.oasis-open.org/regrep/regrep-profile-xsd/v4.0/csd01/regrep-profile-xsd.odt>  
(Authoritative)  
<http://docs.oasis-open.org/regrep/regrep-profile-xsd/v4.0/csd01/regrep-profile-xsd.pdf>

#### Previous version:

TBD

#### Latest version:

TBD

#### Technical Committee:

OASIS ebXML Registry TC

#### Chairs:

Kathryn Breininger ([Kathryn.r.Breininger@boeing.com](mailto:Kathryn.r.Breininger@boeing.com)), Boeing  
Farrukh Najmi ([farrukh@wellfleetsoftware.com](mailto:farrukh@wellfleetsoftware.com)), Wellfleet Software

#### Editors:

Farrukh Najmi, ([farrukh@wellfleetsoftware.com](mailto:farrukh@wellfleetsoftware.com)), Wellfleet Software  
Oliver Newell ([olivern@ll.mit.edu](mailto:olivern@ll.mit.edu)), MIT Lincoln Laboratories

#### Additional artifacts:

None

#### Related work:

This specification depends upon the ebXML RegRep 4 specification and the ebXML RegRep Profile for XML Namespaces specification.

#### Declared XML namespaces:

urn:oasis:names:tc:ebxml-regrep:profile:xsd

#### Abstract:

This document defines the ebXML RegRep profile for publishing, management, discovery and reuse of XML Schema documents.

#### Status:

This document is a draft specification for review, revision and approval by the OASIS ebXML RegRep TC.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at <http://www.oasis-open.org/committees/regrep/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/regrep/ipr.php>).

**Citation format:**

When referencing this specification the following citation format should be used:

**[regrep-profile-xsd-v4.0]**

*OASIS ebXML RegRep Profile for XML Schema Version 4.0*. 03 January 2012. Committee Specification Draft 01. <http://docs.oasis-open.org/regrep/regrep-profile-xsd/v4.0/csd01/regrep-profile-xsd.html>.

---

## Notices

Copyright © OASIS Open 2011. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

---

# Table of Contents

1	Introduction.....	6
1.1	Terminology.....	6
1.2	Normative References.....	6
1.3	Non-normative References.....	6
1.4	Namespaces.....	6
1.4.1	Namespaces Defined.....	6
1.4.2	Namespaces Referenced.....	7
2	Conformance.....	10
3	Publish Profile.....	11
3.1	Metadata Representing an XML Schema File.....	11
3.2	LID Generation.....	11
3.3	ID Generation.....	11
3.4	Locator Path Generation.....	12
3.5	Models for Managing XML Schema File.....	12
3.6	Publish Using External Repository.....	13
3.7	Publish Using Internal Repository.....	13
3.8	Publishing XML Schema File Dependencies.....	13
3.9	Publishing XML Schema Using a Web Crawler.....	13
4	Cataloging Profile.....	15
4.1	Cataloging Name.....	15
4.2	Cataloging Other Document Nodes.....	15
4.3	QName Representation Format.....	19
5	Discovery Profile.....	21
5.1	Canonical Query: XSDDependenciesQuery.....	21
5.1.1	Parameter Summary.....	21
5.1.2	Query Semantics.....	21
5.2	Canonical Query: XSDDiscoveryQuery.....	21
5.2.1	Parameter Summary.....	21
5.2.2	Query Semantics.....	23
5.3	Canonical Query: XSDUsageQuery.....	23
5.3.1	Parameter Summary.....	23
5.3.2	Query Semantics.....	23
6	Query Manager REST Binding Extension.....	24
6.1	Locators Slot and Locator URLs.....	24
6.1.1	Locator URL for a RegistryObject.....	24
6.1.2	Locator URL for a RepositoryItem.....	25

Appendix A. <a href="#">Acknowledgements</a> .....	26
Appendix B. <a href="#">Revision History</a> .....	27

## Index of Tables

Table 1: Namespaces Defined.....	6
Table 2: Namespaces Referenced.....	9

---

# 1 Introduction

ebXML RegRep is a standard defining the service interfaces, protocols and information model for an integrated registry and repository. The repository stores digital content while the registry stores metadata that describes the content in the repository. This document defines the ebXML RegRep profile for publishing, management, discovery and reuse of XML Schema documents.

## 1.1 Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF [RFC 2119].

The term XSD file may be used interchangeably with the term XML Schema file.

## 1.2 Normative References

- [RFC 2119]** S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. IETF RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>.
- [regrep-overview-v4.0]** OASIS *ebXML RegRep Version 4.0 Part 0: Overview Document*. 15 September 2011. Candidate OASIS Standard 01. <http://docs.oasis-open.org/regrep/regrep-core/v4.0/cos01/regrep-core-overview-v4.0-cos01.html>.
- [regrep-xmllns-v4.0]** OASIS *ebXML RegRep Profile for XML Namespaces Version 4.0*. Committee Specification Draft 01. <http://docs.oasis-open.org/regrep/regrep-profile-xmllns/v4.0/csd01/regrep-profile-xmllns.html>
- [XPath2]** XML Path Language (XPath) 2.0, W3C Recommendation 23 January 2007 <http://www.w3.org/TR/xpath20>
- [XPathFunc]** XQuery 1.0 and XPath 2.0 Functions and Operators, W3C Recommendation 23 January 2007 <http://www.w3.org/TR/xpath-functions>

## 1.3 Non-normative References

- [UML]** Unified Modeling Language <http://www.uml.org>  
<http://www.omg.org/cgi-bin/doc?formal/03-03-01>

## 1.4 Namespaces

### 1.4.1 Namespaces Defined

The following namespaces are defined by this specification.

Prefix	Namespace URI	Defining Specification / Description
rr-xsd	urn:oasis:names:tc:ebxml-regrep:profile:xsd	ebXML RegRep Profile for XML Schema

Table 1: Namespaces Defined

## 33 **1.4.2 Namespaces Referenced**

34 The following is a list of namespaces referenced by this specification. This list is not exhaustive and may  
35 be incomplete.

Namespace Prefix	Namespace URI	Defining Specification
lcm	urn:oasis:names:tc:ebxml-regrep:xsd:lcm:4.0	ebXML RegRep Part 3: XML Schema file xsd/lcm.xsd  Schema used by the LifecycleManager interface.
query	urn:oasis:names:tc:ebxml-regrep:xsd:query:4.0	ebXML RegRep Part 3: XML Schema file xsd/query.xsd  Schema used by the QueryManager interface.
rim	urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0	ebXML RegRep Part 3: XML Schema file xsd/rim.xsd  Schema used for information model objects specified by [regrep-rim-v4.0].
rs	urn:oasis:names:tc:ebxml-regrep:xsd:rs:4.0	ebXML RegRep Part 3: XML Schema file xsd/rs.xsd  Common schema used by registry protocols defined by [regrep-rs-v4.0].
lcm	urn:oasis:names:tc:ebxml-regrep:xsd:lcm:4.0	ebXML RegRep Part 3: XML Schema file xsd/lcm.xsd  Schema used by the LifecycleManager interface.
query	urn:oasis:names:tc:ebxml-regrep:xsd:query:4.0	ebXML RegRep Part 3: XML Schema file xsd/query.xsd  Schema used by the QueryManager interface.
rim	urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0	ebXML RegRep Part 3: XML Schema file xsd/rim.xsd  Schema used for information model objects specified by [regrep-rim-v4.0].
rs	urn:oasis:names:tc:ebxml-regrep:xsd:rs:4.0	ebXML RegRep Part 3: XML Schema file xsd/rs.xsd  Common schema used by registry protocols defined by [regrep-rs-v4.0].
spi	urn:oasis:names:tc:ebxml-regrep:xsd:spi:4.0	ebXML RegRep Part 3: XML Schema file xsd/spi.xsd  Schema used by the service provider interfaces defined by [regrep-rs-v4.0].
xlink	<a href="http://www.w3.org/1999/xlink">http://www.w3.org/1999/xlink</a>	<a href="#">XML Linking Language (XLink) Version 1.1</a>
xsd	<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>	XML Schema <a href="#">[XML Schema Part 1]</a> , <a href="#">[XML Schema Part 2]</a> specification
xsi	" <a href="http://www.w3.org/2001/XMLSchema-instance">http://www.w3.org/2001/XMLSchema-instance</a>	W3C XML Schema specification <a href="#">[XML Schema Part 1]</a> , <a href="#">[XML Schema Part 2]</a> .



*Table 2: Namespaces Referenced*

---

## 36 **2 Conformance**

37 This section defines the requirements for a server implementation claiming to conform to ebXML RegRep Profile  
38 for XML Schema specification.

39 TBD

---

## 40 3 Publish Profile

41 This section specifies how XML Schema documents are published to an ebXML RegRep server.

### 42 3.1 Metadata Representing an XML Schema File

43 A client publishes an XML Schema file by publishing an ExtrinsicObjectType instance as defined by **[regrep-rim-**  
44 **v4.0]** with the additional constraints defined in this section.

### 45 3.2 LID Generation

46 The lid attribute of the ExtrinsicObjectType instance representing the XML Schema document **MUST** be generated  
47 using a concatenation of targetNamespace of the XML Schema file and the fileName of the Xml Schema file using  
48 the following pattern:

49

```
<targetNamespace>:<fileName>
```

50

51 The following is an example from cataloging rim.xsd as specified in **[regrep-xsd-v4.0]**:

```
<RegistryObject xsi:type="ExtrinsicObjectType"  
  id="urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0:rim.xsd"  
  objectType="urn:oasis:names:tc:ebxml-  
regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"  
  ...  
>
```

52

### 53 3.3 ID Generation

54 The id attribute of the ExtrinsicObjectType instance representing the XML Schema document **MUST** be generated  
55 using a concatenation of xsd:schema/@targetNamespace attribute of the XML Schema file, the fileName of the  
56 XML Schema file and optionally the xsd:schema/@version attribute of the XML Schema file using the  
57 following pattern. Note that the version component is optional and if not present then the id attribute value is the  
58 same as the lid attribute value.

59

```
<targetNamespace>:<fileName>{:<version>}?
```

60

61 The following is an example from cataloging rim.xsd as specified in **[regrep-xsd-v4.0]**. In this example  
62 there is no version attribute present and therefore the id attribute value is the same as the lid attribute  
63 value.

```
<RegistryObject xsi:type="ExtrinsicObjectType"  
  id="urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0:rim.xsd"  
  lid="urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0:rim.xsd"  
  objectType="urn:oasis:names:tc:ebxml-  
regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"  
  ...  
>
```

64

65 The following is an example from cataloging [gml.xsd](#). In this example there is a version attribute present  
66 and therefore the id attribute value includes the concatenation of the version attribute value.

```
<RegistryObject xsi:type="ExtrinsicObjectType"
  id="http://www.opengis.net/gml:gml.xsd:3.1.1 2010-01-28"
  lid="http://www.opengis.net/gml:gml.xsd"
  objectType="urn:oasis:names:tc:ebxml-
  regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"
  ...
/>
```

67

### 68 3.4 Locator Path Generation

69 This section specifies the requirements for generation of a locator path for the ExtrinsicObjectType instance  
70 representing the XML Schema document. For a description of locator path see [6.1 Locators Slot and Locator](#)  
71 [URLs](#).

72 A publisher MUST specify a locator path for each XML Schema file published to the server as specified in this  
73 section. The locator path allows the QueryManager interface REST binding to support resource URLs that represent  
74 the hierarchical structure of the content tree that the XML Schema file belongs to. In particular, it allows relative  
75 URL references commonly found in `xsd:import` and `xsd:include` elements of XML Schema files to be resolved when  
76 using the REST interface.

77 A publisher MUST specify a locator path for each XML Schema file that it publishes as follows:

- 78 • The ExtrinsicObject for the XML Schema file MUST have a locators slot as specified in [6.1 Locators](#)  
79 [Slot and Locator URLs](#)
- 80 • The locators slot MUST have at least one locator path element
- 81 • The first path element of the locators slot MUST specify a value that is a path relative to the root of the  
82 content tree being published to the server
- 83 • Subsequent path elements of the locators slot are optional and their value is not governed by this  
84 specification

85 The following is an example from cataloging [gml.xsd](#).

```
<RegistryObject xsi:type="ExtrinsicObjectType"
  id="http://www.opengis.net/gml:gml.xsd:3.1.1 2010-01-28"
  lid="http://www.opengis.net/gml:gml.xsd"
  objectType="urn:oasis:names:tc:ebxml-
  regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"

  <Slot name="urn:oasis:names:tc:ebxml-regrep:rim:RegistryObject:locators">
    <SlotValue xsi:type="CollectionValueType"
      collectionType="urn:oasis:names:tc:ebxml-regrep:CollectionType:List">
      <Element xsi:type="StringValue" >
        <Value>gml/3.1.1/base/gml.xsd</Value>
      </Element>
    </SlotValue>
  </Slot>
  ...
/>
```

86

### 87 3.5 Models for Managing XML Schema File

88 This specification supports the following two models for managing the XML Schema files:

- 89 • **External Repository** - XML Schema files are managed outside the Repository of the RegRep server and  
90 only their metadata is managed within the Registry of the RegRep server
- 91 • **Internal Repository** - XML Schema files are managed as repository items within the Repository of the  
92 RegRep server and their corresponding metadata is managed within the Registry of the RegRep server

### 93 3.6 Publish Using External Repository

94 The ExtrinsicObjectType for the XML Schema file contains a rim:RepositoryItemRef sub-element to reference the  
95 XML Schema file by its URL.

96 The following is an example from cataloging rim.xsd as specified in **[regrep-xsd-v4.0]** using the external  
97 repository model:

```
<RegistryObject xsi:type="ExtrinsicObjectType"
  id="regrep/regrep-core/v4.0/cos01/xsd/rim.xsd"
  lid="regrep/regrep-core/v4.0/cos01/xsd/rim.xsd"
  objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"

  <RepositoryItemRef xlink:href="http://docs.oasis-open.org/regrep/regrep-
core/v4.0/cos01/xsd/rim.xsd"/>
  ...
/>
```

### 98 3.7 Publish Using Internal Repository

99 The ExtrinsicObjectType for the XML Schema file contains a rim:RepositoryItem sub-element to contain the XML  
100 Schema file.

101 The following is an example from cataloging rim.xsd as specified in **[regrep-xsd-v4.0]** using the internal  
102 repository model:

```
<RegistryObject xsi:type="ExtrinsicObjectType"
  id="regrep/regrep-core/v4.0/cos01/xsd/rim.xsd"
  lid="regrep/regrep-core/v4.0/cos01/xsd/rim.xsd"
  objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD"

  <RepositoryItem>...binary encoding of repository item...</RepositoryItem>
  ...
/>
```

103

### 104 3.8 Publishing XML Schema File Dependencies

105 When publishing an XML Schema file there is no requirement that schema files that are dependencies of the file  
106 being published be also published in the same transaction (or ever be published at all). This gives much flexibility in  
107 how XML Schema files get published. Note however that in order for the XSDDependenciesQuery and  
108 XSDUsageQuery to work properly a server SHOULD have all dependencies of a schema published as well.

### 109 3.9 Publishing XML Schema Using a Web Crawler

110 A specialized web crawler may be used to crawl a web site containing XML Schema documents. Such a web  
111 crawler may then publish the XML Schema files that it encounters to the RegRep server. This specification does not  
112 specify such a web-crawler. However, it is likely that such a web crawler would use the interfaces and protocols  
113 specified in **[regrep-rs-v4.0]**.



## 114 4 Cataloging Profile

115 This section defines how XML Schema documents are cataloged when published to the RegRep server.

116 The XML Schema cataloger conforms to the Cataloger interface specified by **[regrep-rs-v4.0]**. The XML Schema  
 117 cataloger catalogs the ExtrinsicObjectType instance representing the XML Schema file and updates it as defined by  
 118 this section. The cataloger processes the XML Schema file content regardless of whether it is represented as a  
 119 RepositoryItem or RepositoryItemRef within the ExtrinsicObjectType instance for the XML Schema file.

### 120 4.1 Cataloging Name

121 The filename component of the path component within the URL for the XML Schema file **MUST** be cataloged into  
 122 the `.rim:Name/rim:LocalizedString/@value` where the `rim:LocalizedString/@lang` is left unspecified.

123 The following is an example from cataloging `rim.xsd` as specified in **[regrep-xsd-v4.0]**:

```
<Name>
  <rim:LocalizedString value="rim.xsd"/>
</Name>
```

### 124 4.2 Cataloging Other Document Nodes

125 The following table summarizes all nodes in an XML Schema document that are cataloged and specifies the rules  
 126 for how they are cataloged. Each cataloging rule is followed by an example of the cataloged metadata produced by  
 127 the cataloging rule. Unless otherwise specified the example is from cataloging `rim.xsd` as specified in **[regrep-xsd-**  
 128 **v4.0]**.

Source Node XPATH	Cataloging Rule
<code>./xsd:annotation/xsd:documentation</code>	<ul style="list-style-type: none"> <li>Each matched node maps to <code>rim:LocalizedString</code> element for the <code>rim:Description</code> element of <code>ExtrinsicObjectType</code> instance</li> <li>The <code>xsd:documentation/@xml:lang</code> attribute, if present, is mapped to the <code>rim:LocalizedString/@lang</code> attribute</li> <li>The content of the <code>xsd:documentation</code> node is mapped to <code>rim:LocalizedString/@value</code> attribute value</li> </ul>
	<pre>&lt;Description&gt;   &lt;LocalizedString xml:lang="en" value="The schema for OASIS ebXML   Registry Information Model"/&gt; &lt;/Description&gt;</pre>
<code>./@targetNamespace</code>	<ul style="list-style-type: none"> <li>Maps to a <code>StringValue</code> Slot with name: <code>"urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:targetNamespace"</code></li> <li>The slot's value is the <code>targetNamespace</code> node content</li> </ul>
	<pre>&lt;Slot name="urn:oasis:names:tc:ebxml- regrep:profile:xsd:slot:targetNamespace"&gt;   &lt;SlotValue xsi:type="StringValue"&gt;     &lt;Value&gt;urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0&lt;/Value&gt;</pre>

Source Node XPATH	Cataloging Rule
	<pre data-bbox="256 243 1365 306" style="background-color: #f0f0f0;"> &lt;/SlotValue&gt; &lt;/Slot&gt;</pre>
<pre data-bbox="198 327 448 359">./@xsi:schemaLocation</pre>	<ul data-bbox="532 327 1425 600" style="list-style-type: none"> <li>• Maps to a MapValueType Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:schemaLocationsMap"</li> <li>• There is a Map/Entry for each schema location specified as content of the xsi:schemaLocation node</li> <li>• The EntryKey for each Entry is a StringValueType whose value is the schema namespace URI</li> <li>• The EntryValue for each Entry is a StringValueType whose value is the schema URL</li> </ul>
	<pre data-bbox="256 636 1365 1056" style="background-color: #f0f0f0;"> &lt;Slot name="urn:oasis:names:tc:ebxml- regrep:profile:xsd:slot:schemaLocationsMap"&gt;   &lt;SlotValue xsi:type="MapValueType"&gt;     &lt;Map&gt;       &lt;Entry&gt;         &lt;EntryKey xsi:type="StringValueType"&gt;           &lt;Value&gt;http://www.w3.org/2001/XMLSchema&lt;/Value&gt;         &lt;/EntryKey&gt;         &lt;EntryValue xsi:type="StringValueType"&gt;           &lt;Value&gt;http://www.w3.org/2001/XMLSchema.xsd&lt;/Value&gt;         &lt;/EntryValue&gt;       &lt;/Entry&gt;     &lt;/Map&gt;   &lt;/SlotValue&gt; &lt;/Slot&gt;</pre>
<pre data-bbox="198 1119 326 1150">./xsd:import</pre>	<ul data-bbox="532 1119 1425 1360" style="list-style-type: none"> <li>• All xsd:import nodes map to a single MapValueType Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:importsMap"</li> <li>• There is a Map/Entry for each xsd:import node</li> <li>• The EntryKey for each Entry is a StringValueType whose value is the value of ./xsd:import/@namespace child node</li> <li>• The EntryValue for each Entry is a StringValueType whose value is the value of ./xsd:import/@schemaLocation child node</li> </ul>
<p data-bbox="198 1392 1029 1423">The following example is from cataloging lcm.xsd as specified in <b>[regrep-xsd-v4.0]</b>.</p>	
	<pre data-bbox="256 1444 1365 1936" style="background-color: #f0f0f0;"> &lt;Slot name="urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:importsMap"&gt;   &lt;SlotValue xsi:type="MapValueType"&gt;     &lt;Map&gt;       &lt;Entry&gt;         &lt;EntryKey xsi:type="StringValueType"&gt;           &lt;Value&gt;urn:oasis:names:tc:ebxml- regrep:xsd:rs:4.0&lt;/Value&gt;         &lt;/EntryKey&gt;         &lt;EntryValue xsi:type="StringValueType"&gt;           &lt;Value&gt;http://docs.oasis-open.org/regrep/regrep- core/v4.0/cos01/xsd/rs.xsd&lt;/Value&gt;         &lt;/EntryValue&gt;       &lt;/Entry&gt;       &lt;Entry&gt;         &lt;EntryKey xsi:type="StringValueType"&gt;           &lt;Value&gt;urn:oasis:names:tc:ebxml- regrep:xsd:rim:4.0&lt;/Value&gt;         &lt;/EntryKey&gt;</pre>



Source Node XPATH	Cataloging Rule
	<pre data-bbox="253 243 1365 468"> &lt;EntryValue xsi:type="StringValue" type="StringValueType"&gt;   &lt;Value&gt;http://docs.oasis-open.org/registre/registre- core/v4.0/cos01/xsd/rim.xsd&lt;/Value&gt; &lt;/EntryValue&gt; &lt;/Entry&gt; &lt;/Map&gt; &lt;/SlotValue&gt; &lt;/Slot&gt; </pre>
<pre data-bbox="198 489 331 510">./xsd:include</pre>	<ul data-bbox="537 489 1403 709" style="list-style-type: none"> <li>• All xsd:include nodes map to a single CollectionValueType Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:<b>includes</b>"</li> <li>• The collectionType is "urn:oasis:names:tc:ebxml-regrep:<b>CollectionType:Set</b>"</li> <li>• There is a Element for each xsd:include node whose xsi:type is StringValueType</li> <li>• The Value for each element is the content of the xsd:include node</li> </ul>
<p data-bbox="198 741 1122 762">The following example is from cataloging <a href="http://schemas.opengis.net/ows/0.4.0/owsAll.xsd">http://schemas.opengis.net/ows/0.4.0/owsAll.xsd</a></p>	
	<pre data-bbox="253 789 1365 1146"> &lt;Slot name="urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:<b>includes</b>"&gt;   &lt;SlotValue xsi:type="CollectionValueType" collectionType="urn:oasis:names:tc:ebxml-regrep:<b>CollectionType:Set</b>"&gt;     &lt;Element xsi:type="StringValueType"&gt;       &lt;Value&gt;http://schemas.opengis.net/ows/0.4.0/owsGetCapabilities.x sd&lt;/Value&gt;     &lt;/Element&gt;     &lt;Element xsi:type="StringValueType"&gt;       &lt;Value&gt;http://schemas.opengis.net/ows/0.4.0/owsExceptionReport.x sd&lt;/Value&gt;     &lt;/Element&gt;   &lt;/SlotValue&gt; &lt;/Slot&gt; </pre>
<pre data-bbox="198 1209 467 1314">./xsd:simpleType/@name   ./xsd:complexType/@na me</pre>	<ul data-bbox="537 1209 1393 1451" style="list-style-type: none"> <li>• Only global (top-level) types are matched</li> <li>• All matched nodes map to a single CollectionValueType Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:<b>declaredTypes</b>"</li> <li>• The collectionType is "urn:oasis:names:tc:ebxml-regrep:CollectionType:Set"</li> <li>• There is an Element for each matched node whose type is StringValueType</li> <li>• The Value for the Element is the value of the matched node (the name)</li> </ul>
<p data-bbox="198 1482 1029 1503">The following example is from cataloging lcn.xsd as specified in <b>[registre-xsd-v4.0]</b>.</p>	
	<pre data-bbox="253 1530 1365 1866"> &lt;Slot name="urn:oasis:names:tc:ebxml- regrep:profile:xsd:slot:<b>declaredTypes</b>"&gt;   &lt;SlotValue xsi:type="CollectionValueType" collectionType="urn:oasis:names:tc:ebxml-regrep:<b>CollectionType:Set</b>"&gt;     &lt;Element xsi:type="StringValueType"&gt;       &lt;Value&gt;UpdateActionType&lt;/Value&gt;     &lt;/Element&gt;     &lt;Element xsi:type="StringValueType"&gt;       &lt;Value&gt;mode&lt;/Value&gt;     &lt;/Element&gt;   &lt;/SlotValue&gt; &lt;/Slot&gt; </pre>

Source Node XPATH	Cataloging Rule
./xsd:element/@name	<ul style="list-style-type: none"> <li>• Only global (top-level) elements are matched</li> <li>• All matched nodes map to a single <code>CollectionValueType</code> Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:<b>declaredElements</b>"</li> <li>• The <code>collectionType</code> is "urn:oasis:names:tc:ebxml-regrep:CollectionType:Set"</li> <li>• There is an <code>Element</code> for each matched node whose type is <code>StringValueType</code></li> <li>• The <code>Value</code> for the <code>Element</code> is the value of the matched node (the name)</li> </ul>

The following example is from cataloging `lcm.xsd` as specified in **[regrep-xsd-v4.0]**.

```

<Slot name="urn:oasis:names:tc:ebxml-
regrep:profile:xsd:slot:declaredElements">
  <SlotValue xsi:type="CollectionValueType"
collectionType="urn:oasis:names:tc:ebxml-regrep:CollectionType:Set">
    <Element xsi:type="StringValueType">
      <Value>RemoveObjectsRequest</Value>
    </Element>
    <Element xsi:type="StringValueType">
      <Value>SubmitObjectsRequest</Value>
    </Element>
    <Element xsi:type="StringValueType">
      <Value>UpdateObjectsRequest</Value>
    </Element>
  </SlotValue>
</Slot>

```

./xsd:attribute/@name	<ul style="list-style-type: none"> <li>• Only global (top-level) attributes are matched</li> <li>• All matched nodes map to a single <code>CollectionValueType</code> Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:<b>declaredAttributes</b>"</li> <li>• The <code>collectionType</code> is "urn:oasis:names:tc:ebxml-regrep:CollectionType:Set"</li> <li>• There is an <code>Element</code> for each matched node whose type is <code>StringValueType</code></li> <li>• The <code>Value</code> for the <code>Element</code> is the value of the matched node (the name)</li> </ul>

The following example is from cataloging `http://schemas.opengis.net/xlink/1.0.0/xlinks.xsd`

```

<Slot name="urn:oasis:names:tc:ebxml-
regrep:profile:xsd:slot:declaredAttributes">
  <SlotValue xsi:type="CollectionValueType"
collectionType="urn:oasis:names:tc:ebxml-regrep:CollectionType:Set">
    <Element xsi:type="StringValueType">
      <Value>to</Value>
    </Element>
    <Element xsi:type="StringValueType">
      <Value>actuate</Value>
    </Element>
    <Element xsi:type="StringValueType">
      <Value>arcrole</Value>
    </Element>
    <Element xsi:type="StringValueType">
      <Value>title</Value>
    </Element>
    <Element xsi:type="StringValueType">

```

Source Node XPATH	Cataloging Rule
	<pre data-bbox="256 243 1365 688"> &lt;Value&gt;role&lt;/Value&gt; &lt;/Element&gt; &lt;Element xsi:type="StringValueTypes"&gt;   &lt;Value&gt;label&lt;/Value&gt; &lt;/Element&gt; &lt;Element xsi:type="StringValueTypes"&gt;   &lt;Value&gt;show&lt;/Value&gt; &lt;/Element&gt; &lt;Element xsi:type="StringValueTypes"&gt;   &lt;Value&gt;from&lt;/Value&gt; &lt;/Element&gt; &lt;Element xsi:type="StringValueTypes"&gt;   &lt;Value&gt;href&lt;/Value&gt; &lt;/Element&gt; &lt;/SlotValue&gt; &lt;/Slot&gt; </pre>
<pre data-bbox="198 751 464 993"> ../xsd:extension/@base   ../xsd:restriction/@base   ../xsd:element/@type   ../xsd:element/@ref   ../xsd:attribute/@type   ../xsd:attribute/@ref </pre>	<ul data-bbox="532 751 1414 1150" style="list-style-type: none"> <li>• Catalogs all references to QNames that are in a namespace other than the targetNamespace for the XML Schema document. Thus, of the nodes that are matched, only those nodes are mapped that are QNames within a different namespace than the targetNamespace of the XML Schema document. In other words, local references are not cataloged.</li> <li>• All mapped nodes are mapped to a single CollectionValueType Slot with name: "urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:references"</li> <li>• The collectionType is "urn:oasis:names:tc:ebxml-regrep:CollectionType:Set"</li> <li>• There is an Element for each matched node whose type is StringValueType</li> <li>• The Value for the Element is the QName of the referenced type, element or attribute. The format of the QName MUST be as specified in <a href="#">4.4 QName Representation Format</a></li> </ul>
<p>The following example is from cataloging lcn.xsd as specified in <b>[regrep-xsd-v4.0]</b>.</p>	
	<pre data-bbox="256 1234 1365 1612"> &lt;Slot name="urn:oasis:names:tc:ebxml-regrep:profile:xsd:slot:references"&gt;   &lt;SlotValue xsi:type="CollectionValueType" collectionType="urn:oasis:names:tc:ebxml-regrep:CollectionType:Set"&gt;     &lt;Element xsi:type="StringValueTypes"&gt;       &lt;Value&gt;{urn:oasis:names:tc:ebxml- <b>regrep:xsd:rim:4.0}RegistryObjectList</b>&lt;/Value&gt;     &lt;/Element&gt;     &lt;Element xsi:type="StringValueTypes"&gt;       &lt;Value&gt;{urn:oasis:names:tc:ebxml- regrep:xsd:rim:4.0}ValueType&lt;/Value&gt;     &lt;/Element&gt;     ...   &lt;/SlotValue&gt; &lt;/Slot&gt; </pre>

129 **4.3 QName Representation Format**

130 A QName represents a **qualified name** as defined in the XML specifications: [XML Schema Part2: Datatypes specification](#), [Namespaces in XML](#), [Namespaces in XML Errata](#).

- 132 This specification requires representing a QName in various situation in a specific format as described in this  
133 section.
- 134 The QName is represented as: "{" + Namespace URI + "}" + local part. If the Namespace URI is null then only the  
135 local part is included in the representation.
- 136 For example to specify rim:RegistryObjectType as QName use: {urn:oasis:names:tc:ebxml-  
137 regrep:xsd:rim:4.0}RegistryObjectType

## 138 5 Discovery Profile

139 This section specifies the canonical queries and canonical query functions that may be used to discovery Discovery  
140 of XML Schema documents.

### 141 5.1 Canonical Query: XSDDependenciesQuery

142 The canonical query XSDDependenciesQuery allows clients to discover XML Schema files that are used by the  
143 specified XML Schema file as a dependency. This query is useful to get a list of XML Schema files that are needed  
144 in order to support specified XML Schema file.

#### 145 5.1.1 Parameter Summary

Parameter	Description	Data Type	Default Value	Cardinality
id	Specifies the id for an ExtrinsicObject that represents an XML Schema file.	string		0..1
levels	Number of dependency levels to search. Use 1 to match immediate dependencies, use positive integer N to match direct and indirect dependencies upto N levels. Use -1 or 0 to match all direct and indirect dependencies.	integer	1	0..1

#### 146 5.1.2 Query Semantics

- 147 ● The id parameter MUST be specified
- 148 ● The query traverses the dependency tree for the XML Schema file matched by the id parameter up to the  
149 specified number of levels and includes each XML Schema file as a dependency matched by the query
- 150 ● This query returns a set of ExtrinsicObjects where each ExtrinsicObject represents an XML Schema file  
151 matches by the query

152 Issue: The query currently does not preserve dependency hierarchy as a tree since matched objects are returned as a  
153 flat list. Should this be improved??

### 154 5.2 Canonical Query: XSDDiscoveryQuery

155 The canonical query XSDDiscoveryQuery allows clients to discover XML Schema files matching specified  
156 parameters.

#### 157 5.2.1 Parameter Summary

Parameter	Description	Data Type	Default Value	Cardinality
declaresAttribute	Specifies a regular expression \$declaresAttribute. Matches XML Schema files that have a node that matches the following XPATH expression  ./xsd:attribute/@name[fn:matches(., \$declaresAttribute)]	string		0..1
declaresElement	Specifies a regular expression \$declaresElement. Matches XML Schema files that have a node	string		0..1

	that matches the following XPATH expression:  ./xsd:element/@name[fn:matches(., \$declaresElement)]			
declaresType	Specifies a regular expression \$declaresType. Matches XML Schema files that have a node that matches the following XPATH expression:  ./xsd:attribute/@name[fn:matches(., \$declaresType)]   ./xsd:complexType/@name[fn:matches(., \$declaresType)]	string		0..1
description	Specifies a regular expression \$description. Matches XML Schema files that have a node that matches the following XPATH expression:  ./xsd:documentation[fn:matches(., \$description)]	string		0..1
importsNamespace	Specifies a regular expression \$importsNamespace. Matches XML Schema files that have a node that matches the following XPATH expression:  ./xsd:import/@namespace[fn:matches(., \$importsNamespace)]	string		0..1
includesSchema	Specifies a regular expression \$includesSchema. Matches XML Schema files that have a node that matches the following XPATH expression:  ./xsd:include/@schemaLocation[fn:matches(., \$includesSchema)]	string		0..1
matchOnAnyParameter	If true then use logical OR between predicates for each parameter	boolean	false	0..1
name	Specifies a regular expression that matches the filename of an XML Schema file	string		0..1
targetNamespace	Specifies a regular expression \$targetNamespace. Matches XML Schema files that have a node that matches the following XPATH expression:  ./@targetNamespace[fn:matches(., \$targetNamespace)]	string		0..1
xsiSchemaLocationNames pace	Specifies a regular expression \$xsiSchemaLocationNamespace. Matches XML Schema files that have a node that matches the following XPATH expression:  ./@xsi:schemaLocation[fn:matches(., fun:string-join(\$xsiSchemaLocationNamespace, '\s'))]	string		0..1
xsiSchemaLocationValue	Specifies a regular expression \$xsiSchemaLocationValue. Matches XML Schema files that have a node that matches the following XPATH expression:	string		0..1

	./@xsi:schemaLocation[fn:matches(., fun:string-join('\s', \$xsiSchemaLocationValue))]			
--	---	--	--	--

158 **5.2.2 Query Semantics**

- 159 ● This query has several optional parameters
- 160 ● Each parameter implies a predicate within the underlying query
- 161 ● Predicates for each supplied parameter are combined using with an implicit LOGICAL AND if  
 162 matchOnAnyParameter is unspecified or false. If it is specified as true then predicates for each supplied  
 163 parameters are combined using a LOGICAL OR
- 164 ● If an optional parameter is not supplied then its corresponding predicate MUST NOT be included in the  
 165 underlying query
- 166 ● This query returns a set of ExtrinsicObjects where each ExtrinsicObject represents an XML Schema file  
 167 matches by the query

168 **5.3 Canonical Query: XSDUsageQuery**

169 The canonical query XSDUsageQuery allows clients to discover XML Schema files that use a particular XML  
 170 Schema file, or alternatively, use a type, attribute or element in a schema file. This query is useful in determining  
 171 which XML Schema files will be impacted when one makes a change to an XML Schema file or the definition of a  
 172 type, attribute or element within an XML Schema file

173 **5.3.1 Parameter Summary**

Parameter	Description	Data Type	Default Value	Cardinality
id	Specifies a regular expression \$id. Matches XML Schema files that reference an XML Schema file whose URL matches the specified id.	string		0..1
qname	Specifies a regular expression \$qname. Matches XML Schema files that have a reference to a QName that matches \$qname.  The QName represnetation format that the regular expression expects to match MUST be as specified in <a href="#">4.4 QName Representation Format</a>	string		0..1

174 **5.3.2 Query Semantics**

- 175 ● Either the id or qname parameter MUST be specified
- 176 ● Both parameters MUST NOT be specified
- 177 ● This query returns a set of ExtrinsicObjects where each ExtrinsicObject represents an XML Schema file  
 178 matches by the query

---

## 179 6 Query Manager REST Binding Extension

### 180 6.1 Locators Slot and Locator URLs

181 This section specifies an extension to the QueryManager REST interface binding specified in **[regrep-rs-v4.0]**.

182 A submitter may define a locators slot for an ExtrinsicObject. The locators slot enables the RegistryObject and its  
183 RepositoryItem (if any) to be accessed via the QueryManager interface REST binding using additional locator URLs  
184 as an alternative to the canonical URL. The locators slot specifies a collection of locator paths such that each locator  
185 path may be used as a suffix to the registry base URL to form a locator URL for the RegistryObject or  
186 RepositoryItem.

187 The following example illustrates a locators slot with one element specifying a locator path:

188

```
<RegistryObject xsi:type="ExtrinsicObjectType"
  objectType="urn:oasis:names:tc:ebxml-
  regrep:ObjectType:RegistryObject:ExtrinsicObject:XML:XSD" ...>

  <Slot name="urn:oasis:names:tc:ebxml-regrep:rim:RegistryObject:locators">
    <SlotValue xsi:type="CollectionValueType"
      collectionType="urn:oasis:names:tc:ebxml-regrep:CollectionType:List">
      <Element xsi:type="StringValueType">
        <Value>regrep/regrep-core/v4.0/cos01/xsd/rim.xsd</Value>
      </Element>
    </SlotValue>
  </Slot>

  <RepositoryItem>...binary encoding of repository item...</RepositoryItem>
  ...
/>
```

189

- 190 • The name of the slot MUST be "urn:oasis:names:tc:ebxml-regrep:rim:RegistryObject:locators"
- 191 • The value of the slot MUST be of type **CollectionValueType**
- The collectionType is "urn:oasis:names:tc:ebxml-regrep:CollectionType:List"
- Each Element of the CollectionValueType is of type StringValueType
- The Value for each element is a locator path as specified by the path component of the URL within section 3.3 of [\[RFC1738\]](#)
- A path segment MUST be absolute and MUST NOT consist of relative path segments such as "." and "..".
- The locator path MUST be unique across all locator paths for all RegistryObjects

#### 192 6.1.1 Locator URL for a RegistryObject

193 The locator URL for RegistryObject has the following pattern:

```
//The {locator} parameter specifies a locator specified for the RegistryObject
GET {server base url}/rest/registryObjects/{locator}
```

194



195 In response to a GET request matching above pattern, a server MUST return the RegistryObject that has a locator  
196 path specified by a locator slot that matches the path specified by the locator segment of the URL.

## 197 **6.1.2 Locator URL for a RepositoryItem**

198 The locator URL for RepositoryItem has the following pattern:

```
//The {locator} parameter specifies a locator specified for the RegistryObject  
GET {server base url}/rest/repositoryItem/{locator}
```

199

200 In response to a GET request matching above pattern, a server MUST return the RepositoryItem for the  
201 RegistryObject that has a locator path specified by a locator slot that matches the path specified by the locator  
202 segment of the URL.

---

## 203 Appendix A. Acknowledgements

204 The following individuals have contributed significantly towards the creation of this specification and are  
205 gratefully acknowledged.

### Technical Committee Contributors:

- 206 ● Kathryn Breininger, Boeing
- 207 ● Carl Mattocks, MetLife
- 208 ● Farrukh Najmi, Wellfleet Software
- 209 ● Oliver Newell, MIT Lincoln Labs
- 210 ● Nikola Stojanovic, Individual

### External Contributors:

- 211 ● Kohsuke Kawaguchi, Individual
- 212 ● Brett Levasseur, MIT Lincoln Labs
- 213 ● Aleksei Valikov, Disy Informationssysteme GmbH

214

215

---

216 **Appendix B. Revision History**

217

<b>Revision</b>	<b>Date</b>	<b>Editor</b>	<b>Changes Made</b>
CSD01-1	03 January 2012	Farrukh Najmi, Oliver Newell	Initial version for 4.0.
CSD01-2	25 January 2012	Farrukh Najmi, Oliver Newell	Specified lid, id, locator generation