



Extensible Resource Identifier (XRI) Version 3.0

Working Draft 02

20 August 2009

Specification URIs:

This Version:

<http://docs.oasis-open.org/xri/xri/v3.0/WD01/xri-syntax-3.0-wd02.html>
<http://docs.oasis-open.org/xri/xri/v3.0/WD01/xri-syntax-3.0-wd02.doc>
<http://docs.oasis-open.org/xri/xri/v3.0/WD01/xri-syntax-3.0-wd02.pdf>

Previous Version:

[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .html](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .html)
[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .doc](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .doc)
[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .pdf](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .pdf)

Latest Version:

[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .html](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .html)
[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .doc](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .doc)
[http://docs.oasis-open.org/\[tc-short-name\] / \[additional path/filename\] .pdf](http://docs.oasis-open.org/[tc-short-name] / [additional path/filename] .pdf)

Technical Committee:

OASIS Extensible Resource Identifier (XRI) TC

Chair(s):

Peter Davis, NeuStar
Drummond Reed, Cordance

Editor(s):

Drummond Reed, Cordance
[Editor name]

Related work:

This specification replaces or supercedes:

- [\[specifications replaced by this standard\]](#)
- [\[specifications replaced by this standard\]](#)

This specification is related to:

- [XRI http: and https: Bindings 1.0](#)
- [XRI info: Binding 1.0](#)
- [XRD 1.0](#)
- [XRI Resolution 3.0](#)

Declared XML Namespace(s):

[list namespaces here]
[list namespaces here]

Abstract:

This document is the normative technical specification for XRI generic syntax and normalization rules.

Status:

This document was last revised or approved by the [TC name | membership of OASIS] on the above date. The level of approval is also listed above. Check the “Latest Version” or “Latest Approved Version” location noted above for possible later revisions of this document.

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/xri/>.

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/xri/ipr.php>).

The non-normative errata page for this specification is located at <http://www.oasis-open.org/committees/xri/>.

Notices

Copyright © OASIS® 2008. 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 names "OASIS", [insert specific trademarked names and abbreviations here] are trademarks 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	Motivations.....	6
1.2	Related Work.....	6
1.2.1	URN (Uniform Resource Name).....	6
1.2.2	Handle and DOI (Digital Object Identifier).....	6
1.2.3	XRD (Extensible Resource Descriptor).....	6
1.2.4	XRI Resolution.....	6
1.2.5	XDI (XRI Data Interchange)	6
1.3	Previous Versions.....	6
1.3.1	XRI 1.0.....	7
1.3.2	XRI 2.0.....	7
1.4	Terminology	7
1.5	Syntax Notation	7
1.6	Normative References	7
1.7	Non-Normative References	8
2	XRI-to-URI/IRI Bindings	9
3	XRI Syntax.....	10
3.1	ABNF	10
3.1.1	XRI 3.0.....	10
3.1.2	IRI (RFC 3987)	11
3.2	Hierarchical Structure	13
3.2.1	Authority	13
3.2.2	Path	13
3.2.3	Query.....	14
3.2.4	Fragment	14
3.3	Segment Structure.....	14
3.3.1	Subsegments.....	14
3.3.2	Global Context Symbols.....	14
3.3.3	Local Context Symbols.....	15
3.3.4	Cross-References	15
3.3.5	Literals.....	16
3.4	Characters	16
3.4.1	Path Characters	16
3.4.2	Reserved Characters	16
3.4.3	General Delimiters.....	16
3.4.4	Sub Delimiters	16
4	Relative XRI References	19
5	XRI Forms and Transformations	20
5.1	Forms.....	20
5.1.1	Native Form	20
5.1.2	XRI Normal Form	20
5.1.3	IRI Normal Form.....	20
5.1.4	URI Normal Form	20

5.2 Transformations	20
5.2.1 Native To/From XRI Normal Form.....	20
5.2.2 XRI Normal Form To/From IRI Normal Form	20
5.2.3 IRI Normal Form To/From URI Normal Form.....	20
6 Normalization and Comparison	21
7 Security and Data Protection Considerations	22
8 Conformance	23
A. Acknowledgements	24
B. Glossary	25
C. Revision History.....	26

1 Introduction

XRI (Extensible Resource Identifier) provides a common language for structured identifiers that may be used to share semantics across protocols, domains, systems, and applications. XRI builds directly on the structure and capabilities of URI (Uniform Resource Identifier) [URI] and IRI (Internationalized Resource Identifier) [IRI]. XRI is a profile of URI and IRI syntax and normalization rules for producing URIs or IRIs that contain additional structure and semantics beyond those specified by [URI] or [IRI].

1.1 Motivations

There are as many reasons for needing a common language for structured identifiers (XRI) as there are for needing a common language for structured data (XML). Some of the most commonly cited motivations are:

- To unambiguously assert that the same resource is being identified across different protocols, e.g., HTTP, HTTPS, FTP, SMTP, XMPP.
- To unambiguously identify the same resource in different contexts, i.e., within different domains, systems, applications, namespaces, etc.
- To assign, resolve, and determine the equivalence of different synonymous identifiers for the same resource, e.g., persistent vs. reassignable synonyms, human-readable vs. machine-friendly synonyms, localized vs. non-localized synonyms.
- To identify different versions of the same resource in a manner that is consistent across multiple domains, systems, and applications.
- To create structured identifiers to address, navigate, and share structured data, such as RDF graphs.

1.2 Related Work

1.2.1 URN (Uniform Resource Name)

URNs were standardized by IETF in [RFC2141]. The key motivation was

1.2.2 Handle and DOI (Digital Object Identifier)

TODO

1.2.3 XRD (Extensible Resource Descriptor)

TODO

1.2.4 XRI Resolution

TODO

1.2.5 XDI (XRI Data Interchange)

XDI is a separate Technical Committee at OASIS. The purpose of XDI is to define a structured data sharing format and protocol based on the RDF graph model, using XRIs to address and describe the graph.

[MORE]

1.3 Previous Versions

This section explains the relationship of this specification to the previous XRI specifications.

37 1.3.1 XRI 1.0

38 TODO

39 1.3.2 XRI 2.0

40 TODO

41 1.4 Terminology

42 The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD
43 NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described
44 in [RFC2119].

45 1.5 Syntax Notation

46 This specification uses the syntax notation employed in [IRI]: Augmented Backus-Naur Form (ABNF),
47 defined in [RFC2234]. Although the ABNF defines syntax in terms of the US-ASCII character encoding,
48 XRI syntax should be interpreted in terms of the character that the ASCII-encoded octet represents,
49 rather than the octet encoding itself, as explained in [URI]. As with URIs, the precise bit-and-byte
50 representation of an XRI reference on the wire or in a document is dependent upon the character
51 encoding of the protocol used to transport it, or the character set of the document that contains it.

52 The following core ABNF productions are used by this specification as defined by section 6.1 of
53 [RFC2234]: ALPHA, CR, CTL, DIGIT, DQUOTE, HEXDIG, LF, OCTET and SP. The complete XRI ABNF
54 syntax is collected in section 3.1.

55 To simplify comparison between generic XRI syntax and generic IRI syntax, the ABNF productions that
56 are unique to XRIs are shown with light green shading, while those inherited from [IRI] are shown with
57 light yellow shading.

58 | This is an example of ABNF specific to XRI.

59 | This is an example of ABNF inherited from IRI.

60 1.6 Normative References

- 61 [IRI] M. Dürst, M. Suignard, *Internationalized Resource Identifiers (IRIs)*,
62 <http://www.ietf.org/rfc/rfc3987.txt>, RFC 3987, January 2005.
- 63 [RFC1737] K. Sollins, L. Masinter, *Functional Requirements for Uniform Resource Names*,
64 <http://www.ietf.org/rfc/rfc1737.txt>, RFC 1737, December 1994.
- 65 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
66 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 67 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
68 <http://www.ietf.org/rfc/rfc2119.txt>, RFC 2119, March 1997.
- 69 [RFC2141] R. Moats, *URN Syntax*, <http://www.ietf.org/rfc/rfc2141.txt>, IETF RFC 2141, May
70 1997.
- 71 [RFC2234] D. H. Crocker and P. Overell, *Augmented BNF for Syntax Specifications: ABNF*,
72 <http://www.ietf.org/rfc/rfc2234.txt>, RFC 2234, November 1997.
- 73 [RFC2718] L. Masinter, H. Alvestrand, D. Zigmund, R. Petke, *Guidelines for New URL*
74 *Schemes*, <http://www.ietf.org/rfc/rfc2718.txt>, RFC 2718, November 1999.
- 75 [RFC2732] R. Hinden, B. Carpenter, L. Masinter, *Format for Literal IPv6 Addresses in URL's*,
76 <http://www.ietf.org/rfc/rfc2732.txt>, RFC 2732, December, 1999.
- 77 [RFC3305] M. Mealing, R. Denenberg, *Uniform Resource Identifiers (URIs), URLs, and*
78 *Uniform Resource Names (URNs): Clarifications and Recommendations*,
79 <http://www.ietf.org/rfc/rfc3305.txt>, RFC 3305, August 2002.

- 80 **[RFC3491]** P. Hoffman, M. Blanchet, *Nameprep: A Stringprep Profile for Internationalized*
81 *Domain Names (IDN)*, <http://www.ietf.org/rfc/rfc3491>, RFC 3491, March 2003.
- 82 **[RFC3629]** F. Yergeau, *UTF-8, A Transformation Format of ISO 10646*,
83 <http://www.faqs.org/rfcs/rfc3629.html>, RFC 3629, November, 2003.
- 84 **[UniXML]** M. Dürst, A. Freytag, *Unicode in XML and other Markup Languages*, Unicode
85 Technical Report #20, World Wide Web Consortium Note, February 2002.
- 86 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifier (URI):*
87 *Generic Syntax*, <http://www.ietf.org/rfc/rfc3986.txt>, STD 66, RFC 3986, January
88 2005.
- 89 **[UTR15]** M. Davis, M. Dürst, *Unicode Normalization Forms*,
90 <http://www.unicode.org/unicode/reports/tr15/tr15-23.html>, April 17, 2003.
- 91 **[Reference]** [Full reference citation]

92 **1.7 Non-Normative References**

- 93 **[Reference]** [Full reference citation]
- 94

NOTE: The proper format for a citation to an OASIS Technical Committee’s work (whether Normative or Non-Normative) is:

OASIS
Stage (Committee Draft 01, Committee Draft 02, Committee Specification 01, etc. or Standard)
Title (italicized or in quotation marks)
Approval Date (Month YYYY)
URI of the actual Authoritative Specification (namespace is not acceptable as the content changes over time)

For example:

EDXL-HAVE OASIS Standard, “Emergency Data Exchange Language (EDXL) Hospital
AVailability Exchange (HAVE) Version 1.0”, November 2008.
http://docs.oasis-open.org/emergency/edxl-have/os/emergency_edxl_have-1.0-spec-os.doc

95
96
97
98

2 XRI-to-URI/IRI Bindings

100 XRIs are called *abstract identifiers* (see the *Glossary* in Appendix B) because they do not, by themselves,
101 resolve directly to resource representations on the Internet or other digital networks. Rather an XRI is
102 always an identifier for an abstract, non-information resource, and it must be bound to a specific
103 *resolution context* before it can be resolved. This binding is achieved by appending the XRI to a URI or
104 IRI to form another syntactically valid URI/IRI. The result is called a *bound XRI*.

105 Different types of bound XRIs are referred to by the scheme name of the URI/IRI to which the XRI is
106 bound, e.g., http: XRI, https: XRI, xmpp: XRI, etc. The URI/IRI to which the XRI is bound is referred to as
107 the *base URI/IRI*. The process of parsing the original XRI from the bound XRI is called *unbinding*, and
108 when necessary to avoid ambiguity, the resulting XRI is referred to as an *unbound XRI*.

109 An XRI MAY be bound to any valid URI/IRI provided the binding produces a syntactically valid URI/IRI.
110 However the resulting URI/IRI may not be recognizable as a bound XRI, either by humans or machines,
111 unless that binding conforms to a formal specification.

112 Following are the requirements for an XRI binding specification:

- 113 • The specification SHOULD be named by the URI/IRI scheme name (or names, if the specification
114 defines more than one binding), e.g., *XRI info: Binding Specification*.
- 115 • The specification SHOULD explain the motivations for this binding, and the applications for which this
116 binding is recommended or preferred over other bindings.
- 117 • The specification MUST specify the ABNF rules for the binding.
- 118 • The specification MUST declare the normal form into which XRIs must be transformed prior to the
119 binding. See section 5, *XRI Forms and Transformations*.
- 120 • The specification MUST define any other encoding rules that apply to XRIs prior to binding.
- 121 • The specification MAY include or reference a resolution specification that defines the resolution of
122 XRIs bound according to the specification.
- 123 • The specification MUST include a conformance section [*ref: some language about conformance*
124 *requirements – ref to SAML AuthN Context or Core where they talk about Profiles*].
- 125 • The specification MUST include a Security and Data Protection section.
126

127 XRI 3.0 includes the following binding specifications:

- 128 • *XRI http: and https: Binding 1.0*.
- 129 • *XRI mailto: Binding 1.0*.
- 130 • *XRI info: Binding 1.0*.

131 3 XRI Syntax

132 3.1 ABNF

133 The ABNF for XRI 3.0 builds on the ABNF for IRI as defined in [IRI]. The complete ABNF trees for both
134 are included in this section for ease of reference.

135 3.1.1 XRI 3.0

136 Following is the normative ABNF syntax for XRI 3.0. Sections 3.2 through 3.4 explain this structure in
137 more detail.

```
138 xri-reference      = xri
139                   / relative-xri-ref
140 xri                 = xri-hier-part [ "?" iquery ] [ "#" ifragment ]
141 relative-xri-ref   = relative-xri-part [ "?" iquery ] [ "#" ifragment ]
142 relative-xri-part = xri-path-abs
143                   / xri-path-noscheme
144                   / ipath-empty
145 xri-hier-part      = xri-authority xri-path-abempty
146 xri-authority     = global-subseg *subseg
147 xri-path           = xri-path-abempty
148                   / xri-path-abs
149                   / xri-path-noscheme
150                   / ipath-empty
151 xri-path-abempty  = *( "/" xri-segment )
152 xri-path-abs      = "/" [ xri-segment-nz *( "/" xri-segment ) ]
153 xri-path-noscheme = xri-segment-nc *( "/" xri-segment )
154 xri-segment       = [ literal ] *subseg
155 xri-segment-nz    = ( literal / subseg ) *subseg
156 xri-segment-nc    = ( literal-nc / subseg ) *subseg
157 subseg            = global-subseg
158                   / local-subseg
159                   / xref
```

```

160 global-subseg = gcs-char [ local-subseg / xref / literal ]
161 local-subseg = lcs-char [ xref / literal ]
162 gcs-char = "=" / "@" / "+" / "$"
163 lcs-char = "*" / "!"
164 xref = "(" [ xref-value ] ")"
165 xref-value = xri-reference
166 / iri
167 literal = 1*xri-pchar
168 literal-nc = 1*xri-pchar-nc
169 xri-pchar = iunreserved / pct-encoded / xri-sub-delims / ":"
170 xri-pchar-nc = iunreserved / pct-encoded / xri-sub-delims
171 xri-reserved = xri-gen-delims / xri-sub-delims
172 xri-gen-delims = ":" / "/" / "?" / "#" / "[" / "]" / "(" / ")"
173 / gcs-char / lcs-char
174 xri-sub-delims = "&" / ";" / "," / "'"

```

3.1.2 IRI (RFC 3987)

The following ABNF from [IRI] is included here for reference only.

```

177 IRI = scheme ":" ihier-part [ "?" iquery ]
178 [ "#" ifragment ]
179 scheme = ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )
180 ihier-part = "//" iauthority ipath-abempty
181 / ipath-abs
182 / ipath-rootless
183 / ipath-empty
184 iauthority = [ iuserinfo "@" ] ihost [ ":" port ]
185 iuserinfo = *( iunreserved / pct-encoded / sub-delims / ":" )
186 ihost = IP-literal / IPv4address / ireg-name
187 IP-literal = "[" ( IPv6address / IPvFuture ) "]"
188 IPvFuture = "v" 1*HEXDIG "." 1*( unreserved / sub-delims / ":" )

```

```

189 IPv6address      =          6( h16 ":" ) ls32
190                  /          "::" 5( h16 ":" ) ls32
191                  / [          h16 ] "::" 4( h16 ":" ) ls32
192                  / [ *1( h16 ":" ) h16 ] "::" 3( h16 ":" ) ls32
193                  / [ *2( h16 ":" ) h16 ] "::" 2( h16 ":" ) ls32
194                  / [ *3( h16 ":" ) h16 ] "::"   h16 ":"   ls32
195                  / [ *4( h16 ":" ) h16 ] "::"                   ls32
196                  / [ *5( h16 ":" ) h16 ] "::"                   h16
197                  / [ *6( h16 ":" ) h16 ] "::"

198 ls32            = ( h16 ":" h16 ) / IPv4address

199 h16             = 1*4HEXDIG

200 IPv4address     = dec-octet "." dec-octet "." dec-octet "." dec-octet

201 dec-octet      = DIGIT           ; 0-9
202                  / %x31-39 DIGIT ; 10-99
203                  / "1" 2DIGIT    ; 100-199
204                  / "2" %x30-34 DIGIT ; 200-249
205                  / "25" %x30-35   ; 250-255

206 ireg-name      = *( iunreserved / pct-encoded / sub-delims )

207 port           = *DIGIT

208 ipath-abempty   = *( "/" isegment )

209 ipath-abs       = "/" [ isegment-nz *( "/" isegment ) ]

210 ipath-rootless = isegment-nz *( "/" isegment )

211 ipath-empty     = 0<ipchar>

212 isegment       = *ipchar

213 isegment-nz    = 1*ipchar

214 iquery         = *( ipchar / iprivate / "/" / "?" )

215 iprivate       = %xE000-F8FF / %xF0000-FFFFD / %x100000-10FFFFD

216 ifragment      = *( ipchar / "/" / "?" )

217 ipchar         = iunreserved / pct-encoded / sub-delims / ":" / "@"

218 iunreserved    = ALPHA / DIGIT / "-" / "." / "_" / "~" / ucschar

219 pct-encoded    = "%" HEXDIG HEXDIG

220 ucschar        = %xA0-D7FF / %xF900-FDCF / %xFDF0-FFEF
221                  / %x10000-1FFFFD / %x20000-2FFFFD / %x30000-3FFFFD
222                  / %x40000-4FFFFD / %x50000-5FFFFD / %x60000-6FFFFD
223                  / %x70000-7FFFFD / %x80000-8FFFFD / %x90000-9FFFFD
224                  / %xA0000-AFFFFD / %xB0000-BFFFFD / %xC0000-CFFFFD
225                  / %xD0000-DFFFFD / %xE1000-EFFFFD

226 reserved      = gen-delims / sub-delims

227 gen-delims     = ":" / "/" / "?" / "#" / "[" / "]" / "@"

```

```

228 sub-delims      = "!" / "$" / "&" / "'" / "(" / ")"
229                / "*" / "+" / "," / ";" / "="
230
231 unreserved      = ALPHA / DIGIT / "-" / "." / "_" / "~"

```

231 3.2 Hierarchical Structure

232 XRI follows the same generic hierarchical structure as URI/IRI. It supports the same query and fragment
 233 syntax, as well as the same distinction between absolute and relative references.

```

234 xri-reference    = xri
235                  / relative-xri-ref
236
237 xri              = xri-hier-part [ "?" iquery ] [ "#" ifragment ]
238
239 relative-xri-ref = relative-xri-part [ "?" iquery ] [ "#" ifragment ]
240
241 relative-xri-part = xri-path-abs
242                  / xri-path-noscheme
243                  / ipath-empty
244
245 xri-hier-part    = xri-authority xri-path-abempty

```

242 3.2.1 Authority

243 Like an absolute URI/IRI, an absolute XRI begins with an authority component. URI/IRI architecture
 244 defines a special syntax for this component. With XRI architecture, the authority segment uses the same
 245 subsegment structure as any other XRI segment with one exception: it MUST begin with a global
 246 subsegment (see section 3.3).

```

247 xri-authority    = global-subseg *subseg

```

248 3.2.2 Path

249 All segments in a relative XRI, and all segments after the authority segment in an absolute XRI, constitute
 250 the path component. The XRI path syntax options are the same as with URI/IRI syntax with the exception
 251 that if a relative XRI that does not begin with a forward slash ("/"), the first path segment MUST NOT
 252 include a colon. This prevents a relative XRI from being interpreted as an absolute URI/IRI.

```

253 xri-path         = xri-path-abempty
254                  / xri-path-abs
255                  / xri-path-noscheme
256                  / ipath-empty
257
258 xri-path-abempty = *( "/" xri-segment )
259
260 xri-path-abs     = "/" [ xri-segment-nz *( "/" xri-segment ) ]
261
262 xri-path-noscheme = xri-segment-nc *( "/" xri-segment )
263
264 ipath-empty      = 0<ipchar>

```

261 3.2.3 Query

262 The query component of XRI syntax is inherited directly from IRI syntax.

```
263 | iquery = *( ipchar / iprivate / "/" / "?" )
```

264 3.2.4 Fragment

265 The fragment component of XRI syntax is also inherited directly from IRI syntax.

```
266 | ifragment = *( ipchar / "/" / "?" )
```

267 3.3 Segment Structure

268 Besides scheme-independence, the fundamental difference between XRI syntax and URI/IRI syntax is
269 segment structure. URI/IRI architecture has no structure within segments; each segment is completely
270 transparent to a URI/IRI processor. XRI segments are further structured into *subsegments*. Each
271 subsegment is delimited with special delimiter characters within a segment in the same way a segment is
272 delimited by forward slashes within a path. Note that the very first subsegment within a segment is the
273 only subsegment that does not require a leading delimiter – it may be a literal (see section 3.3.5).

```
274 | xri-segment = [ literal ] *subseg  
275 | xri-segment-nz = ( literal / subseg ) *subseg  
276 | xri-segment-nc = ( literal-nc / subseg ) *subseg
```

277 3.3.1 Subsegments

278 There are three types of XRI subsegments: global, local, and cross-references (*xref* in the ABNF).
279 Global subsegments must start with a global context symbol, and may be followed by either a local
280 subsegment, a cross-reference, or a literal. Local subsegments must start with a local context symbol,
281 and may be followed by either a cross-reference or a literal.

```
282 | subseg = global-subseg  
283 |         / local-subseg  
284 |         / xref  
285 | global-subseg = gcs-char [ local-subseg / xref / literal ]  
286 | local-subseg = lcs-char [ xref / literal ]
```

287 3.3.2 Global Context Symbols

288 There are four XRI global context symbols, chosen from among the valid subdelimiters in URI/IRI syntax.

```
289 | gcs-char = "=" / "@" / "+" / "$"
```

290 The purpose of global context symbols is to define a small set of contexts shared globally across all XRIs,
291 as specified in Table 1.

292

Context Symbol	Context Name	Definition
=	Personal	Identifiers for which the ultimate authority is an individual person.

@	Organizational	Identifiers for which the ultimate authority is an organization.
+	General	Identifiers for which there is no ultimate authority because they represent generic dictionary concepts or “tags” whose meaning is determined by consensus. (In the English language, for example, these would be the generic nouns.)
\$	Special	Identifiers for which the ultimate authority is a standards setting organization. Authority for identifiers assigned in this context is delegated by the OASIS XRI Technical Committee.

293 *Table 1: Global Context Symbols*

294 3.3.3 Local Context Symbols

295 There are two XRI local context symbols, again chosen from among the valid delimiters in URI/IRI syntax.

296 `lcs-char = "*" / "!"`

297 The purpose of local context symbols is to define two universal characteristics of identifiers in any
 298 context, as specified in Table 2.

299

Context Symbol	Context Name	Definition
*	Reassignable/ Recyclable	The association between the identifier and the resource it identifies MAY change over time.
!	Persistent/ Non-recyclable	The association between the identifier and the resource it identifies MUST NOT change over time.

300 *Table 2: Local Context Symbols*

301 **IMPORTANT:** Because a local context symbol is OPTIONAL following a global context symbol or a
 302 segment delimiter (forward slash), the default characteristic for any literal subsegment not preceded by a
 303 local context symbol is that it is reassignable. XRI authorities SHOULD use the “!” persistent context
 304 identifier for any subsegment intended to be persistent.

305 3.3.4 Cross-References

306 Cross-references are another key component of XRI structure, and also one of its primary extensibility
 307 mechanisms. Cross-reference syntax enables encapsulation of identifiers that would otherwise not be
 308 parsed as a single XRI subsegment (such as a URI or an IRI that is not a bound XRI) to be treated as a
 309 single XRI subsegment. Cross-references one method XRI syntax provides for an identifier assigned in
 310 one context to be reused in another context, permitting identifiers to be shared across contexts. This
 311 simplifies identifying logically equivalent resources across hierarchies (a directory concept referred to as
 312 “heterarchy” or “polyarchy”).

313 A cross-reference is distinguished by enclosing it in parentheses, similar to the way an IPv6 literal is
 314 encapsulated in square brackets as specified in [RFC2732]. A cross-reference may contain either an XRI
 315 reference or an absolute IRI.

316 `xref = "(" [xref-value] ")"`
 317 `xref-value = xri-reference`
 318 `/ iri`

319 3.3.5 Literals

320 Literals are the atomic elements of XRI structure. They are the strings of path characters that ultimately
321 serve to identify a resource in a context.

```
322 literal           = 1*xri-pchar
```

```
323 literal-nc       = 1*xri-pchar-nc
```

324 3.4 Characters

325 The final ABNF rules governing XRI characters and encoding are inherited from **[IRI]**, which is a
326 superset of generic URI syntax as defined in **[URI]**. The primary difference is that XRI global context
327 symbols and local context symbols have been moved into the set of general delimiters.

328 3.4.1 Path Characters

329 As with URI/IRI syntax, XRI path characters may be any character except general delimiters.

```
330 xri-pchar         = iunreserved / pct-encoded / xri-sub-delims / ":"
```

```
331 xri-pchar-nc     = iunreserved / pct-encoded / xri-sub-delims
```

332 3.4.2 Reserved Characters

333 The reserved characters fall into general delimiters and subdelimiters.

```
334 xri-reserved     = xri-gen-delims / xri-sub-delims
```

335 If an XRI reserved character is used as a data character and not as a delimiter, the character **MUST** be
336 percent-encoded per the rules in section 3.4.5. XRI references that differ in the percent-encoding of a
337 reserved character **MUST NOT** be considered equivalent.

338 3.4.2.1 General Delimiters

339 XRI general delimiters are the URI/IRI general delimiter set plus the XRI global context symbols and local
340 context symbols.

```
341 xri-gen-delims   = ":" / "/" / "?" / "#" / "[" / "]" / "(" / ")"
```

```
342                / gcs-char / lcs-char
```

343 3.4.2.2 Subdelimiters

344 XRI subdelimiters are the URI/IRI subdelimiters minus those allocated to the XRI general delimiter set.

```
345 xri-sub-delims  = "&" / ";" / "," / "'"
```

346 3.4.3 Unreserved Characters

347 XRI has the same set of unreserved characters as **[IRI]**.

```
348 iunreserved     = ALPHA / DIGIT / "-" / "." / "_" / "~" / ucchar
```

```

349     ucschar          = %xA0-D7FF / %xF900-FDCF / %xFDF0-FFEF
350                       / %x10000-1FFFD / %x20000-2FFFFD / %x30000-3FFFFD
351                       / %x40000-4FFFFD / %x50000-5FFFFD / %x60000-6FFFFD
352                       / %x70000-7FFFFD / %x80000-8FFFFD / %x90000-9FFFFD
353                       / %xA0000-AFFFFD / %xB0000-BFFFFD / %xC0000-CFFFFD
354                       / %xD0000-DFFFFD / %xE1000-EFFFFD

```

355 Percent-encoding unreserved characters in an XRI does not change what resource is identified by that
356 XRI. However, it may change the result of an XRI comparison (see section [ref]), so unreserved
357 characters SHOULD NOT be percent-encoded.

358 3.4.4 Character Encoding

359 The standard character encoding of XRI is UTF-8, as recommended by [RFC2718]. When an XRI
360 reference is presented as a human-readable identifier, the representation of the XRI reference in the
361 underlying document may use the character encoding of the underlying document. However, this
362 representation must be converted to UTF-8 before the XRI can be processed outside the document. This
363 encoding in UTF-8 MUST include normalization according to Normalization Form KC (NFKC) as defined
364 in [UTR15]. The stricter NFKC is specified rather than Normalization Form C (NFC) used in IRI encoding
365 [IRI] because NFKC reduces the number of UCS compatibility characters allowed in an XRI and
366 increases the probability of equivalence matches.

367 3.4.5 Percent-Encoded Characters

368 XRIs follow the same rules for percent-encoding as IRIs and URIs. That is, any *data* character in an XRI
369 reference MUST be percent-encoded if it does not have a representation using an unreserved character
370 but SHOULD NOT be percent-encoded if it does have a representation using an unreserved character.

371 A percent-encoded octet is a character triplet consisting of the percent character “%” followed by the two
372 hexadecimal digits representing that octet’s numeric value.

```

373     pct-encoded       = "%" HEXDIG HEXDIG

```

374 The uppercase hexadecimal digits “A” through “F” are equivalent to the lowercase digits “a” through “f”,
375 respectively. XRI references that differ only in the case of hexadecimal digits used in percent-encoded
376 octets are equivalent. For consistency, XRI generators and normalizers SHOULD use uppercase
377 hexadecimal digits for percent-encoded triplets.

378 Note that a % symbol used to represent itself in an XRI reference (i.e., as data and not to introduce a
379 percent-encoded triplet) MUST be percent-encoded.

380 3.4.6 Excluded Characters

381 Certain characters, such as the space character, are excluded from XRI syntax and must be percent-
382 encoded in order to be represented within an XRI. Systems responsible for accepting or presenting XRI
383 references may choose to percent-encode excluded characters on input and/or decode them prior to
384 display, as described above. A string that contains these characters in a non-percent-encoded form,
385 however, is not a valid XRI.

386 Note that presenting “space” or other whitespace characters in a non-percent-encoded form is not
387 recommended for several reasons. First, it is often difficult to visually determine the number of spaces or
388 other characters composing a block of whitespace, leading to transcription errors. Second, the space
389 character is often used to delimit an XRI reference, so non-percent-encoded whitespace characters can
390 make it difficult or impossible to determine where the identifier ends. Finally, non-percent-encoded
391 whitespace can be used to maliciously construct subtly different identifiers intended to mislead the reader.
392 For these reasons, non-percent-encoded whitespace characters SHOULD be avoided in presentation,
393 and alternatives to whitespace as a logical separator within XRIs (such as dots or hyphens) SHOULD be
394 used whenever possible.

395 [IRI] provides the following guidance concerning other characters that should be avoided. This guidance
396 applies to XRIs as well.

397 “The UCS contains many areas of characters for which there are strong visual look-
398 alike. Because of the likelihood of transcription errors, these also should be avoided.
399 This includes the full-width equivalents of Latin characters, half-width Katakana
400 characters for Japanese, and many others. This also includes many look-alikes of
401 ‘space’, ‘delims’, and ‘unwise’, characters excluded in **[RFC3491]**.”

402 “Additional information is available from **[UniXML]**. **[UniXML]** is written in the context of
403 running text rather than in the context of identifiers. Nevertheless, it discusses many of
404 the categories of characters not appropriate for IRIs.”

405 Finally, although they are not excluded characters, special care should be taken by user agents with
406 regard to the display of UCS characters that are visual look-alikes (homographs) for XRI delimiters (all
407 characters in the xri-reserved production, section **Error! Reference source not found.**). See section
408 **Error! Reference source not found.**, “**Error! Reference source not found.**” for additional information.

409 **4 Relative XRI References**

410 TODO

411 5 XRI Forms and Transformations

412 TODO

413 5.1 Forms

414 5.1.1 Native Form

415 TODO

416 5.1.2 XRI Normal Form

417 TODO

418 An XRI reference thus percent-encoded is said to be in *XRI-normal form*. Not all XRI references in XRI-
419 normal form are syntactically legal IRI or URI references. Rules for converting an XRI reference to a valid
420 IRI or URI reference are discussed in section **Error! Reference source not found.** An XRI reference is
421 in XRI-normal form if it is minimally percent-encoded and matches the ABNF provided in this document,
422 but it is a valid IRI or URI reference only after it is percent-encoded according to the transformation
423 described in section **Error! Reference source not found.**

424

425 5.1.3 IRI Normal Form

426 TODO

427 5.1.4 URI Normal Form

428 TODO

429 5.2 Transformations

430 5.2.1 Native To/From XRI Normal Form

431 TODO

432 5.2.2 XRI Normal Form To/From IRI Normal Form

433 TODO

434 5.2.3 IRI Normal Form To/From URI Normal Form

435 TODO

436 **6 Normalization and Comparison**

437 TODO

438 **7 Security and Data Protection Considerations**

439 TODO

440

- 441
- All bindings need to include their own Security and Data Protection Considerations sections.

442 **8 Conformance**

443 The last numbered section in the specification must be the Conformance section. Conformance
444 Statements/Clauses go here.

445 **A. Acknowledgements**

446 The following individuals have participated in the creation of this specification and are gratefully
447 acknowledged:

448 **Participants:**

449 [Participant Name, Affiliation | Individual Member]

450 [Participant Name, Affiliation | Individual Member]

451

452 **B. Glossary**

453 TODO [This will be largely inherited from XRI Syntax 2.0]

454

C. Revision History

455

Revision	Date	Editor	Changes Made
1.0	2009-05-20	Drummond Reed	First draft: includes all ABNF and partial text for the ABNF sections.
2.0	2009-08-20	Drummond Reed	Second draft: Section 3 (ABNF section) now complete. Sections 4-8 still TODO.

456

457