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] .doc
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] .doc
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

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

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.

1.3.1 XRI 1.0

TODO

1.3.2 XRI 2.0

TODO

1.4 Terminology

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

1.5 Syntax Notation

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

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

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

```
   This is an example of ABNF specific to XRI.

   This is an example of ABNF inherited from IRI.
```

1.6 Normative References

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

[RFC3491] P. Hoffman, M. Blanchet, *Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN),* http://www.ietf.org/rfc/rfc3491, RFC 3491, March 2003.

[RFC3629] F. Yergeau, *UTF-8, A Transformation Format of ISO 10646,* http://www.faqs.org/rfcs/rfc3629.html, RFC 3629, November, 2003.

[UniXML] M. Dürst, A. Freytag, *Unicode in XML and other Markup Languages*, Unicode Technical Report #20, World Wide Web Consortium Note, February 2002.

[URI] T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax,* http://www.ietf.org/rfc/rfc3986.txt, STD 66, RFC 3986, January 2005.

[UTR15] M. Davis, M. Dürst, *Unicode Normalization Forms*, http://www.unicode.org/unicode/reports/tr15/tr15-23.html, April 17, 2003.

[Reference] [Full reference citation]

1.7 Non-Normative References

[Reference] [Full reference citation]

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

2 XRI-to-URI/IRI Bindings

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

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

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

Following are the requirements for an XRI binding specification:

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

XRI 3.0 includes the following binding specifications:

- *XRI http: and https: Binding 1.0.*
- *XRI mailto: Binding 1.0.*
- *XRI info: Binding 1.0.*

3 XRI Syntax

3.1 ABNF

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

3.1.1 XRI 3.0

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

```
xri-reference     = xri
                  / relative-xri-ref
xri               = xri-hier-part [ "?" iquery ] [ "#" ifragment ]
relative-xri-ref  = relative-xri-part [ "?" iquery ] [ "#" ifragment ]
relative-xri-part = xri-path-abs
                  / xri-path-noscheme
                  / ipath-empty
xri-hier-part     = xri-authority xri-path-abempty
xri-authority     = global-subseg *subseg
xri-path          = xri-path-abempty
                  / xri-path-abs
                  / xri-path-noscheme
                  / ipath-empty
xri-path-abempty  = *( "/" xri-segment )
xri-path-abs      = "/" [ xri-segment-nz *( "/" xri-segment ) ]
xri-path-noscheme = xri-segment-nc *( "/" xri-segment )
xri-segment       = [ literal ] *subseg
xri-segment-nz    = ( literal / subseg ) *subseg
xri-segment-nc    = ( literal-nc / subseg ) *subseg
subseg            = global-subseg
                  / local-subseg
                  / xref
```

```
global-subseg     = gcs-char [ local-subseg / xref / literal ]

local-subseg      = lcs-char [ xref / literal ]

gcs-char          = "=" / "@" / "+" / "$"

lcs-char          = "*" / "!"

xref              = "(" [ xref-value ] ")"

xref-value        = xri-reference

                  / iri

literal           = 1*xri-pchar

literal-nc        = 1*xri-pchar-nc

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

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

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

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

                  / gcs-char / lcs-char

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

3.1.2 IRI (RFC 3987)

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

```
IRI               = scheme ":" ihier-part [ "?" iquery ]
                  [ "#" ifragment ]

scheme            = ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )

ihier-part        = "//" iauthority ipath-abempty
                  / ipath-abs
                  / ipath-rootless
                  / ipath-empty

iauthority        = [ iuserinfo "@" ] ihost [ ":" port ]

iuserinfo         = *( iunreserved / pct-encoded / sub-delims / ":" )

ihost             = IP-literal / IPv4address / ireg-name

IP-literal        = "[" ( IPv6address / IPvFuture  ) "]"

IPvFuture         = "v" 1*HEXDIG "." 1*( unreserved / sub-delims / ":" )
```

```
IPv6address       =                            6( h16 ":" ) ls32
                  /                       "::" 5( h16 ":" ) ls32
                  / [               h16 ] "::" 4( h16 ":" ) ls32
                  / [ *1( h16 ":" ) h16 ] "::" 3( h16 ":" ) ls32
                  / [ *2( h16 ":" ) h16 ] "::" 2( h16 ":" ) ls32
                  / [ *3( h16 ":" ) h16 ] "::"    h16 ":"   ls32
                  / [ *4( h16 ":" ) h16 ] "::"              ls32
                  / [ *5( h16 ":" ) h16 ] "::"              h16
                  / [ *6( h16 ":" ) h16 ] "::"

ls32              = ( h16 ":" h16 ) / IPv4address

h16               = 1*4HEXDIG

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

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

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

port              = *DIGIT

ipath-abempty     = *( "/" isegment )

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

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

ipath-empty       = 0<ipchar>

isegment          = *ipchar

isegment-nz       = 1*ipchar

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

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

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

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

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

pct-encoded       = "%" HEXDIG HEXDIG

ucschar           = %xA0-D7FF / %xF900-FDCF / %xFDF0-FFEF
                  / %x10000-1FFFD / %x20000-2FFFD / %x30000-3FFFD
                  / %x40000-4FFFD / %x50000-5FFFD / %x60000-6FFFD
                  / %x70000-7FFFD / %x80000-8FFFD / %x90000-9FFFD
                  / %xA0000-AFFFD / %xB0000-BFFFD / %xC0000-CFFFD
                  / %xD0000-DFFFD / %xE1000-EFFFD

reserved          = gen-delims / sub-delims

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

```
sub-delims        = "!" / "$" / "&" / "'" / "(" / ")"
                  / "*" / "+" / "," / ";" / "="

unreserved        = ALPHA / DIGIT / "-" / "." / "_" / "~"
```

3.2 Hierarchical Structure

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

```
xri-reference     = xri

                  / relative-xri-ref

xri               = xri-hier-part [ "?" iquery ] [ "#" ifragment ]

relative-xri-ref  = relative-xri-part [ "?" iquery ] [ "#" ifragment ]

relative-xri-part = xri-path-abs

                  / xri-path-noscheme

                  / ipath-empty

xri-hier-part     = xri-authority xri-path-abempty
```

3.2.1 Authority

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

```
xri-authority     = global-subseg *subseg
```

3.2.2 Path

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

```
xri-path          = xri-path-abempty

                  / xri-path-abs

                  / xri-path-noscheme

                  / ipath-empty

xri-path-abempty  = *( "/" xri-segment )

xri-path-abs      = "/" [ xri-segment-nz *( "/" xri-segment ) ]

xri-path-noscheme = xri-segment-nc *( "/" xri-segment )
```

```
ipath-empty       = 0<ipchar>
```

3.2.3 Query

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

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

3.2.4 Fragment

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

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

3.3 Segment Structure

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

```
xri-segment       = [ literal ] *subseg

xri-segment-nz    = ( literal / subseg ) *subseg

xri-segment-nc    = ( literal-nc / subseg ) *subseg
```

3.3.1 Subsegments

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

```
subseg            = global-subseg

                  / local-subseg

                  / xref

global-subseg     = gcs-char [ local-subseg / xref / literal ]

local-subseg      = lcs-char [ xref / literal ]
```

3.3.2 Global Context Symbols

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

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

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

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 preceeded 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
```

3.3.5 Literals

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

```
literal           = 1*xri-pchar

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

3.4 Characters

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

3.4.1 Path Characters

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

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

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

3.4.2 Reserved Characters

The reserved characters fall into general delimiters and subdelimiters.

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

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

3.4.2.1 General Delimiters

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

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

                  / gcs-char / lcs-char
```

3.4.2.2 Subdelimiters

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

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

3.4.3 Unreserved Characters

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

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

```
ucschar          = %xA0-D7FF / %xF900-FDCF / %xFDF0-FFEF
                 / %x10000-1FFFD / %x20000-2FFFD / %x30000-3FFFD
                 / %x40000-4FFFD / %x50000-5FFFD / %x60000-6FFFD
                 / %x70000-7FFFD / %x80000-8FFFD / %x90000-9FFFD
                 / %xA0000-AFFFD / %xB0000-BFFFD / %xC0000-CFFFD
                 / %xD0000-DFFFD / %xE1000-EFFFD
```

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

3.4.4 Character Encoding

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

3.4.5 Percent-Encoded Characters

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

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

```
pct-encoded     = "%" HEXDIG HEXDIG
```

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

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

3.4.6 Excluded Characters

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

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

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

397 “The UCS contains many areas of characters for which there are strong visual look-
398 alikes. 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.

4 Relative XRI References

TODO

5 XRI Forms and Transformations

TODO

5.1 Forms

5.1.1 Native Form

TODO

5.1.2 XRI Normal Form

TODO

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

5.1.3 IRI Normal Form

TODO

5.1.4 URI Normal Form

TODO

5.2 Transformations

5.2.1 Native To/From XRI Normal Form

TODO

5.2.2 XRI Normal Form To/From IRI Normal Form

TODO

5.2.3 IRI Normal Form To/From URI Normal Form

TODO

6 Normalization and Comparison

TODO

7 Security and Data Protection Considerations

TODO

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

8 Conformance

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

A. Acknowledgements

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

Participants:

[Participant Name, Affiliation | Individual Member]

[Participant Name, Affiliation | Individual Member]

B. Glossary

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

C. Revision History

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.