Extensible Resource Identifier (XRI) Syntax V2.0

Committee Draft 01, 14 March 2005

Document identifier:

xri-syntax-V2.0-cd-01

Location:

http://docs.oasis-open.org/xri/xri/V2.0

Editors:

Drummond Reed, Cordance <drummond.reed@cordance.net>
Dave McAlpin, Epok <dave.mcalpin@epok.net>

Contributors:

Peter Davis, Neustar <peter.davis@neustar.biz>
Nat Sakimura, NRI <n-sakimura@nri.co.jp>
Mike Lindelsee, Visa International <mlindels@visa.com>
Gabe Wachob, Visa International <gwachob@visa.com>

Abstract:

This document is the normative technical specification for XRI generic syntax. For a non-normative introduction to the uses and features of XRIs, see *Introduction to XRIs* at **[XRIIntro]**. For the HTTP-based XRI resolution protocol, see *Extensible Resource Identifier (XRI) Resolution V2.0* at **[XRIResolution]**. For the set of XRIs defined to provide metadata about other XRIs, see *Extensible Resource Identifier (XRI) Metadata V2.0* at **[XRIMetadata]**.

Status:

This document was last revised or approved by the XRI Technical Committee on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

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.

Table of Contents

Introduction

1.1 Overview of XRIs

Extensible Resource Identifiers (XRIs) provide a standard means of abstractly identifying a resource independent of any particular concrete representation of that resource—or, in the case of a completely abstract resource, of any representation at all.

As shown in Figure 1, XRIs build on the foundation established by URIs (Uniform Resource Identifiers) and IRIs (Internationalized Resource Identifiers) as defined by **[URI]** and **[IRI]**, respectively.

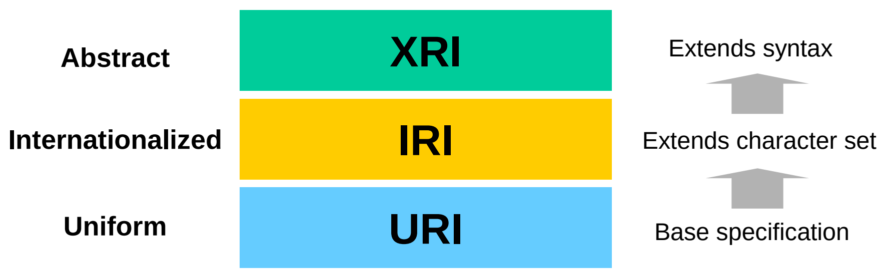

Figure 1: The relationship of XRIs, IRIs, and URIs

The IRI specification created a new identifier by extending the unreserved character set to include characters beyond those allowed in generic URIs. It also defined rules for transforming this identifier into a syntactically legal URI. Similarly, this specification creates a new identifier, an XRI, that extends the syntactic elements (but not the character set) allowed in IRIs. To accommodate applications that expect IRIs or URIs, this specification also defines rules for transforming an XRI reference into a valid IRI or URI reference.

Although an XRI is not a Uniform Resource Name (URN) as defined in *URN Syntax* **[RFC2141]**, an XRI consisting entirely of persistent segments is designed to meet the requirements set out in *Functional Requirements for Uniform Resource Names* **[RFC1737]**.

This document specifies the normative syntax for XRIs, along with associated normalization, processing and equivalence rules. Two additional specifications complete the XRI 2.0 suite:

- *XRI Resolution* **[XRIResolution]** specifies both a standard and a trusted HTTP-based resolution protocol for XRIs. Use of these protocols is not required; XRIs may also be resolved using other protocols or resolution mechanisms.
- *XRI Metadata* **[XRIMetadata]** specifies a small set of standard metadata identifiers registered under the XRI global context symbol "$" that may be used to describe the contents of an XRI reference.

See also *An Introduction to XRIs* **[XRIIntro]** for a non-normative introduction to XRI 2.0 syntax, resolution, and metadata via a set of practical examples.

1.1.1 Generic Syntax

XRI syntax follows the same basic pattern as IRI and URI syntax. A fully-qualified XRI consists of the prefix “xri://” followed by the same four components as a generic authority-based IRI or URI.

```
xri://  authority   / path   ? query   # fragment
```

The definitions of these components are, for the most part, supersets of the equivalent components in the generic IRI or URI syntax. One advantage of this approach is that the vast majority of HTTP URIs and IRIs, which derive directly from generic URI syntax, can be transformed to valid XRIs simply by changing the scheme from “http” to “xri”. This transformation is discussed in Appendix B, “Transforming HTTP IRIs to XRIs”.

XRI syntax extends generic IRI syntax in the following four ways:

1. *Persistent and reassignable segments.* Unlike generic URI syntax, XRI syntax allows the internal components of an XRI reference to be explicitly designated as either persistent or reassignable.
2. *Cross-references.* Cross-references allow XRI references to contain other XRI references or IRIs as syntactically-delimited sub-segments. This provides syntactic support for “compound identifiers”, i.e., the use of well-known, fully-qualified identifiers within the context of another XRI reference. Typical uses of cross-references include using well-known types of metadata in an XRI reference (such as versioning metadata as defined in the *XRI Metadata* specification **[XRIMetadata]**), or the use of globally-defined identifiers to mark parts of an XRI reference as having application- or vocabulary-specific semantics.
3. *Global context symbols.* While XRI syntax supports the same generic syntax used in IRIs for DNS and IP authorities, it also provides shorthand symbols for establishing the abstract global context of an identifier.
4. *Standardized federation.* Federated identifiers are those delegated across multiple authorities, such as DNS names. Generic URI syntax leaves the syntax for federated identifiers up to individual URI schemes, with the exception of explicit support for IP addresses. XRI syntax standardizes federation of both persistent and reassignable identifiers at any level of the path.

1.1.2 URI, URL, URN, and XRI

The evolution and interrelationships of the terms “URI”, “URL”, and “URN” are explained in a report from the Joint W3C/IETF URI Planning Interest Group, *Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations* **[RFC3305]**. According to section 2.1:

> “During the early years of discussion of web identifiers (early to mid 90s), people assumed that an identifier type would be cast into one of two (or possibly more) classes. An identifier might specify the location of a resource (a URL) or its name (a URN), independent of location. Thus a URI was either a URL or a URN.”

This view has since changed, as the report goes on to state in section 2.2:

> “Over time, the importance of this additional level of hierarchy seemed to lessen; the view became that an individual scheme did not need to be cast into one of a discrete set of URI types, such as ‘URL’, ‘URN’, ‘URC’, etc. Web-identifier schemes are, in general, URI schemes, as a given URI scheme may define subspaces.”

This conclusion is shared by **[URI]** which states in section 1.1.3:

> “An individual [URI] scheme does not have to be classified as being just one of ‘name’ or ‘locator’. Instances of URIs from any given scheme may have the characteristics of names or locators or both, often depending on the persistence and care in the assignment of identifiers by the naming authority, rather than on any quality of the scheme.”

XRIs are consistent with this philosophy. Although XRIs are designed to fulfill the requirements of abstract “names” that are resolved into concrete locators, the XRI syntax does not distinguish between identifiers that represent “names”, “locators” or “characteristics.”

1.2 Terminology and Notation

1.2.1 Keywords

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]**. When these words are not capitalized in this document, they are meant in their natural language sense.

1.2.2 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 Appendix A.

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.
```

Lastly, because the prefix “xri://” is optional in absolute XRIs that use a global context symbol (see section 2.2.1.2), some example XRIs are shown without this prefix.

2 Syntax

This section defines the normative syntax for XRIs. Note that additional constraints are inherited from **[IRI]** and **[URI]**, as defined in section 2.2. Also note that some productions in the XRI ABNF are ambiguous. As with IRIs and URIs, a “first-match-wins” rule is used to disambiguate ambiguous productions. See **[URI]** for more details.

2.1 Characters

XRI character set and encoding are inherited from **[IRI]**, which is a superset of generic URI syntax as defined in **[URI]**.

2.1.1 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.

2.1.2 Reserved Characters

The overall XRI reserved character set is the same as the reserved character set defined by **[URI]** and **[IRI]**. Due to the extended syntax of XRIs, however, the allocation of reserved characters between the “general delimiters” and “sub-delimiters” productions is different. Those characters that have defined semantics in generic XRI syntax appear in the xri-gen-delims production. Those characters that do not have defined semantics but that are reserved for use as implementation-specific delimiters appear in the xri-sub-delims production. The rgcs-char production that appears in xri-gen-delims below is discussed in section 2.2.1.2.

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

xri-gen-delims  = ":" / "/" / "?" / "#" / "[" / "]" / "(" / ")"
                / "*" / "!" / rgcs-char

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 2.1.4, “Percent-Encoded Characters”. XRI references that differ in the percent-encoding of a reserved character are not equivalent.

2.1.3 Unreserved Characters

The characters allowed in XRI references that are not reserved are called unreserved. XRI has the same set of unreserved characters as the "iunreserved" production in **[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 2.5, "Normalization and Comparison"), so unreserved characters SHOULD NOT be percent-encoded.

2.1.4 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. Delimiters in an XRI reference that have a representation using a reserved character MUST NOT be percent-encoded.

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 2.3.1. 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 2.3.1.

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.

```
escaped          = "%" 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.

2.1.4.1 Encoding XRI Metadata

In some cases, the transformation of an identifier in its native language and display format into an XRI reference in XRI-normal form may lose information that cannot be retained through percent-encoding. For example, in certain languages, displaying the glyph of a UTF-8 encoded character requires additional language and font information not available in UTF-8. The loss of this information during UTF-8 encoding might cause the resulting XRI to be ambiguous.

XRI syntax offers an option for encoding this language metadata using a cross-reference beginning with the GCS "$" symbol (see section 2.2.1.2). The top level authority for $l language metadata is the *XRI Metadata Specification* **[XRIMetadata]**, specifically section 2. See also section 3 for "$d" date/time metadata, section 4 for "$v" version metadata, and section 5 for "$-" annotation metadata.

2.1.5 Excluded Characters

Certain characters, such as "space", are excluded from the 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 in section 2.1.4. 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.

> "The UCS contains many areas of characters for which there are strong visual look-alikes. Because of the likelihood of transcription errors, these also should be avoided. This includes the full-width equivalents of Latin characters, half-width Katakana characters for Japanese, and many others. This also includes many look-alikes of 'space', 'delims', and 'unwise', characters excluded in **[RFC3491]**."

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

2.2 Syntax Components

XRI syntax builds on generic IRI (and ultimately, URI) syntax. However because XRI syntax includes syntactic elements other than those defined in **[IRI]** and **[URI]**, this specification defines a new protocol element, "XRI", along with rules for transforming XRI references into generic IRI or URI references for applications that expect them (see section 2.3.1, "Transforming XRI References into IRI and URI References"). An XRI reference MUST be constructed such that it qualifies as a valid IRI as defined by **[IRI]** when converted to IRI-normal form and such that it qualifies as a valid URI as defined by **[URI]** when converted to URI-normal form.

As with URIs, an XRI must be in absolute form, while an XRI reference may be either an XRI or a relative XRI reference.

```
XRI-reference     = XRI / relative-XRI-ref

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

absolute-XRI      = [ "xri://" ] xri-hier-part [ "?" iquery ]

xri-value         = xri-no-scheme / relative-XRI-ref

xri-no-scheme     = xri-hier-part [ "?" iquery ]
                    [ "#" ifragment ]

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

xri-hier-part     = ( xri-authority / iauthority )
                    [ xri-path-absolute ] / ipath-empty
```

An XRI begins with an optional prefix "xri://" followed by the same set of hierarchical components as a URI – authority, path, query, and fragment. An XRI is always in absolute form. A relative XRI reference consists of an XRI path followed by an optional XRI query and optional XRI fragment. The absolute-XRI production is provided for contexts that require an XRI in absolute form but that do not allow the fragment identifier.

Finally, in certain contexts where XRIs are used exclusively, the prefix "xri://" is redundant. These contexts can use the xri-value production, which includes all levels of XRI paths.

2.2.1 Authority

XRIs support the same types of authorities as generic IRIs, called *IRI authorities*. XRIs also support an additional type of abstract identification authority called an *XRI authority*.

2.2.1.1 XRI Authority

There are two ways to express an XRI authority: using a global context symbol (GCS), or using a cross-reference (abbreviated in the ABNF as *xref*). Cross-references are covered in section 2.2.2.

```
xri-authority     = gcs-authority / xref-authority
```

2.2.1.2 Global Context Symbol (GCS) Authority

XRIs offer a simple, compact syntax for indicating the logical global context of an identifier: a single prefix character called a *global context symbol*.

```
gcs-authority     = pgcs-authority / rgcs-authority

pgcs-authority    = "!" xri-subseg-pt-nz *xri-subseg

rgcs-authority    = rgcs-char xri-segment

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

The global context symbol characters were selected from the set of symbol characters that are valid in a URI under **[URI]**. The bang character, "!", which is used uniformly in XRI syntax to indicate a persistent identifier segment, serves as the GCS character for global persistent identifiers. The other GCS characters may be used to indicate the global context of either a persistent or a reassignable identifier as shown in Table 1 below:

Symbol Character	Authority Type	Establishes Global Context For
=	Person	Identifiers for whom the authority is an individual person.
@	Organization	Identifiers for whom the authority is an organization or a resource in an organizational context.
+	General public	Identifiers for whom the authority is the general public, i.e., that represent generic "dictionary" concepts for which there is no specific authority. (In the English language, for example, these would be the generic nouns.)
$	Standards body	Identifiers for whom the authority is a specification from a standards body, for example, other XRI specifications (such as *XRI Resolution* **[XRIResolution]** and *XRI Metadata* **[XRIMetadata]**), other OASIS specifications, or (using cross-references) other standards bodies.

Table 1: XRI global context symbols.

2.2.1.3 IRI Authority

XRIs support the same type of authority defined by the "iauthority" production of **[IRI]**.

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

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

ihost             = IP-literal / IPv4address / ireg-name

port              = *DIGIT
```

The syntax is inherited directly from **[IRI]**. First, the “iuserinfo” sub-component permits the identification of a user in the context of a host. Next, the “ihost” sub-component has three options for identifying the host: a registered name (such as a domain name), an IPv4 address, or an IPv6 literal.

A host identifier can be followed by an optional port number. The XRI syntax specification does not define a default port because it is expected this will be inherited from the resolution protocol, such as the HTTP/HTTPS protocol specified in **[XRIResolution]**. Therefore, if the port is omitted in an XRI, it is undefined.

Note that authority segments that begin with GCS characters or cross-references (see below) may match both the “iauthority” and the “xri-authority” productions. For instance, “!!1”, “@example”, “=example”, “+example”, “$example” and “(=example)” all match both productions. As with all XRI syntax, the “first-match-wins” rule is used to resolve ambiguities. Consequently, all the examples listed above would be considered XRI authorities, not IRI authorities.

2.2.2 Cross-References

Cross-references are the primary extensibility mechanism in XRI. They allow an identifier assigned in one context to be reused in another context, permitting identifiers to be shared across contexts. This simplifies identifying logically equivalent resources across hierarchies (a directory concept referred to as “polyarchy”).

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

```
    xref              = "(" ( XRI-reference / IRI ) ")"
```

It is important that the value of a cross-reference be syntactically unambiguous, whether it is an absolute IRI or one of the various forms of an XRI reference. Therefore special attention must be paid to relative XRI references to avoid ambiguity, as discussed in section 2.4.3.

A cross-reference may appear at any node of any XRI except within an IRI authority segment. A cross-reference as the very first sub-segment in an XRI is a valid top-level XRI authority.

```
xref-authority    = xref *xri-subseg
```

This syntax allows any globally-unique identifier in any URI scheme (e.g., an HTTP URI, mailto URI, URN etc.) to specify a global XRI authority.

```
xri://(mailto:john.doe@example.com)/favorites/home
      --example of using a URI as an XRI global authority
```

2.2.3 Path

As with IRIs, the XRI path component is a hierarchal sequence of path segments separated by slash (“/”) characters and terminated by the first question-mark (“?”) or number sign (“#”) character, or by the end of the XRI reference. But while an IRI path segment is considered opaque by a generic URI processor, an XRI path segment can be parsed by an XRI processor into two types of sub-segments: * *segments* (pronounced “star segments”) and *! segments* (pronounced “bang segments”).

```
xri-path          = xri-path-absolute
                  / xri-path-noscheme
                  / ipath-empty

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

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

xri-segment       = xri-subseg-od *xri-subseg

xri-segment-nz    = xri-subseg-od-nz *xri-subseg

xri-subseg        = ( "*" / "!" ) (xref / *xri-pchar)

xri-subseg-nc     = ( "*" / "!" ) (xref / *xri-pchar-nc)

xri-subseg-od     = [ "*" / "!" ] (xref / *xri-pchar)

xri-subseg-od-nz  = [ "*" / "!" ] (xref / 1*xri-pchar)

xri-subseg-od-nx  = [ "*" / "!" ] 1*xri-pchar-nc

xri-subseg-pt-nz  = "!" (xref / 1*xri-pchar)
```

* segments are used to specify *reassignable identifiers*—identifiers that may be reassigned by an identifier authority to represent a different resource at some future date. ! segments are used to specify *persistent identifiers*—identifiers that are permanently assigned to a resource and will not be reassigned at a future date. The default is a * segment, so no leading star ("*") is required for the first (or only) sub-segment.

An XRI path segment may contain the same characters as a URI path segment plus the expanded UCS character set inherited from **[IRI]**. If a star ("*") or bang ("!") appears in a path of an XRI reference, it will be interpreted as a sub-segment delimiter. If this interpretation is not desired for these characters, or for any other special XRI delimiters, these characters MUST be percent-encoded when they appear in the path segment. See section 2.1.4, "Percent-Encoded Characters".

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

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

With the exception of star ("*"), bang ("!") and cross-reference delimiters, an XRI path segment is considered opaque by generic XRI syntax. As with IRIs, XRI extensions or generating applications may define special meanings for other XRI reserved characters for the purpose of delimiting extension-specific or generator-specific sub-components.

Note that XRI syntax is slightly more restrictive than URI syntax in that the first segment of an absolute XRI path may never be empty, even in the absolute form of an XRI.

2.2.4 Query

The XRI query component is identical to the IRI query component as described in section 2.2 of **[IRI]**.

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

2.2.5 Fragment

XRI syntax also supports fragments as described in section 2.2 of **[IRI]**.

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

Since XRI federation syntax can inherently address attributes or sub-resources to any depth, fragments are supported primarily for compatibility with generic URI syntax. XRIs can also employ cross-references to identify media types or other alternative representations of a resource. See section 2.2.2

2.3 Transformations

2.3.1 Transforming XRI References into IRI and URI References

Although XRIs are intended to be used by applications that understand them natively, it may also be desirable to use them in contexts that do not recognize an XRI reference but that allow an Internationalized Resource Identifier reference as described in **[IRI]**, or a fully-conformant URI reference as defined by **[URI]**.

This section specifies the steps for transforming an XRI reference into a valid IRI reference. At the completion of these steps, the XRI reference is in *IRI-Normal Form*. An XRI reference in IRI-Normal Form may then be mapped into a valid URI reference by following the algorithms defined in section 3.1 of **[IRI]**. After that mapping, the XRI reference is in *URI-Normal Form*.

Applications MUST transform XRI references to IRI references using the following steps (or a process that achieves exactly the same result). Before applying these steps, the XRI reference must be in XRI-normal form as defined in section 2.1.4.

1. If the XRI reference is not encoded in UTF-8, convert the XRI reference to a sequence of characters encoded in UTF-8, normalized according to Normalization Form C (NFC) as defined in **[UTR15]**.
2. If the XRI reference is not relative (i.e., if it matches the “XRI” ABNF production) and the optional “xri://” prefix has been omitted, prepend “xri://” to the XRI reference.
3. Optionally add XRI metadata using cross-references as defined in section 2.1.4.1. Note that the addition of XRI metadata may change the resulting IRI or URI reference for the purposes of comparison. The significance or insignificance of specific types of XRI metadata is specified in *Extensible Resource Identifier (XRI) Metadata V2.0* **[XRIMetadata]**.
4. Apply the XRI escaping rules defined in section 2.3.2. Note that this step is not idempotent (i.e., it may yield a different result if applied more than once), so it is very important that implementers not apply this step more than once to avoid changing the semantics of the identifier.

At the completion of step 4, the percent-encoded XRI reference may be used as an IRI reference. An XRI reference in this form is said to be in *IRI-normal form*.

Applying this conversion does not change the equivalence of the identifier, with the possible exception of the addition of XRI metadata as discussed in Step 3.

In general, an application SHOULD use the least-transformed version appropriate for the context in which the identifier appears. For example, if the context allows an XRI reference directly, the identifier SHOULD be an XRI reference in XRI-normal form as described in section 2.1.4. If the context allows an IRI reference but not an XRI reference, the identifier SHOULD be in IRI-normal form. Only when context allows neither XRI nor IRI references should URI-normal form be used.

2.3.2 Escaping Rules for XRI Syntax

This section defines rules for preventing misinterpretation of XRI syntax when an XRI reference is evaluated by a non-XRI-aware parser.

The first rule deals with cross-references as explained in section 2.2.2. Since a cross-reference contains either an IRI or an XRI reference (which itself may contain further nested IRIs or XRI references), it may include characters that, if not escaped, would cause misinterpretation when the XRI reference is used in a context that expects an IRI or URI reference. Consider the following XRI:

```
xri://@example/(xri://@example2/abc?id=1)
```

The generic parsing algorithm described in **[URI]** would separate the above XRI into the following components:

```
scheme = xri
authority = @example
path = /(xri://@example2/abc
query = id=1)
```

The desired separation is:

```
scheme = xri
authority = @example
path = /(xri://@example2/abc?id=1)
query = <undefined>
```

To avoid this type of misinterpretation, certain characters in a cross-reference must be percent-encoded when transforming an XRI reference into IRI-normal form. In particular, the question mark ("?") character must be percent-encoded as "%3F" and the number sign "#" character must be percent-encoded as "%28".

Following this rule, the above example would be expressed as:

```
xri://@example/(xri://@example2%3Fid=1)
```

In addition, the slash "/" character in a cross-reference may also be misinterpreted by a non-XRI-aware parser. Consider:

```
xri://@example.com/(@example/abc)
```

If this were used as a base URI as defined in section 5 of **[URI]**, the algorithm described in section 5.2 of **[URI]** would append a relative-path reference to:

```
xri://@example.com/(@example/
```

instead of the intended:

```
xri://@example.com/
```

This is because the "merge" algorithm in section 5.2.3 of **[URI]** is defined in terms of the last (right-most) slash character. This problem is avoided by encoding slashes within cross-references as "%2F". Following this rule, the above example would be expressed as:

```
xri://@example.com/(@example%2Fabc)
```

Ambiguity is also possible if an XRI reference in XRI-normal form contains characters that have been percent-encoded to indicate that they should not be interpreted as delimiters. For example, consider the following XRI in XRI-normal form:

```
xri://@example.com/(@example/abc%2Fd/ef)
```

This slash character between "c" and "d" is percent-encoded to show that it's not a syntactical element of the XRI, i.e., that it should be interpreted as data and not as a delimiter. To preserve this type of distinction when converting an XRI reference to an IRI reference, the percent "%" character must be percent-encoded as "%25". Following this rule, the above example fully converted would be:

```
xri://@example.com/(@example%2Fabc%252Fd%2Fef)
```

To summarize, the following four special rules MUST be applied during step 4 of section 2.3.1. Before applying these rules, the XRI reference MUST be in XRI-normal form and all IRIs in cross-references MUST be in a percent-encoded form appropriate to their schemes.

1. Percent-encode all percent "%" characters as "%25" across the entire XRI reference.
2. Percent-encode all number sign "#" characters that appear within a cross-reference as "%23".
3. Percent-encode all question mark "?" characters that appear within a cross-reference as "%3F".
4. Percent-encode all slash "/" characters that appear within a cross-reference as "%2F".

2.3.3 Transforming IRI References into XRI References

Transformation of an XRI reference in IRI-normal form into an XRI reference in XRI-normal form MUST use the following steps (or a process that achieves the same result).

1. If the XRI reference is not encoded in UTF-8, convert the XRI reference to a sequence of characters encoded in UTF-8, normalized according to Normalization Form C (NFC) as defined in **[UTR15]**.
2. Perform the following special conversions for XRI syntax:
   a. Convert all percent-encoded slash ("/") characters to their corresponding octets.
   b. Convert all percent-encoded question mark ("?") characters to their corresponding octets.
   c. Convert all percent-encoded number sign ("#") characters to their corresponding octets.
   d. Convert all percent-encoded percent ("%") characters to their corresponding octets.

Note that this process is not idempotent (i.e., it may yield a different result if applied more than once), so it is very important that implementers only apply this process to XRI references in IRI-normal form. If it is applied to an XRI reference in XRI-normal form, the resulting identifier may not be equivalent to the XRI reference before transformation.

2.4 Relative XRI References

2.4.1 Reference Resolution

For XRI references in IRI-normal form or URI-normal form, resolving a relative XRI reference into an absolute XRI reference is straightforward. If the base XRI and the relative XRI reference are in IRI-normal form, section 6.5 of **[IRI]** applies. If the base XRI and the relative XRI reference are in URI-normal form, section 5 of **[URI]** applies.

It is important that XRI references appear in a form appropriate to their context (i.e., in URI-normal form in contexts that expect URI references and in IRI-normal form in contexts that expect IRI references), since the algorithms described in **[IRI]** and **[URI]** may produce incorrect results when applied to XRI references in XRI-normal form, particularly when those XRI references contain cross-references.

In contexts that allow a native XRI reference (i.e., an XRI reference in XRI-normal form), it may be useful to perform relative reference resolution without first converting to IRI- or URI-normal form. In fact, it may be difficult or impossible to convert to IRI- or URI-normal form without first resolving the relative XRI reference to an absolute XRI. The algorithms described in section 5 of **[URI]** apply to XRI references in XRI-normal form provided that the processor:

- treats the characters allowed in IRI references but not in URI references the same as it treats unreserved characters in URI references (as required by section 5 of **[IRI]**) and
- treats all characters within all cross-references the same as unreserved characters in URI references (i.e., treats cross-references as opaque with respect to relative reference resolution).

2.4.2 Reference Resolution Examples

The following are examples of relative XRI reference resolution. These examples are very similar to the examples for resolving relative references in **[URI]**. Starting with the following base XRI in XRI-normal form:

```
xri://@a*a/!b!b/c*c/(xri://@d*d/e)?q
```

a relative reference is transformed to its target XRI as shown in the following examples.

2.4.2.1 Normal Examples

```
!g!g          =  xri://@a*a/!b!b/c*c/!g!g
./!g!g        =  xri://@a*a/!b!b/c*c/!g!g
!g!g/         =  xri://@a*a/!b!b/c*c/!g!g/
/!g!g         =  xri://@a*a/!g!g
//@!g!g       =  Not a legal relative XRI reference
?y            =  xri://@a*a/!b!b/c*c/(xri://@d*d/e)?y
!g!g?y        =  xri://@a*a/!b!b/c*c/!g!g?y
#s            =  xri://@a*a/!b!b/c*c/(xri://@d*d/e)?q#s
!g!g#s        =  xri://@a*a/!b!b/c*c/!g!g#s
!g!g?y#s      =  xri://@a*a/!b!b/c*c/!g!g?y#s
;x            =  xri://@a*a/!b!b/c*c/;x
!g!g;x        =  xri://@a*a/!b!b/c*c/!g!g;x
!g!g;x?y#s    =  xri://@a*a/!b!b/c*c/!g!g;x?y#s
              =  xri://@a*a/!b!b/c*c/(xri://@d*d/e)?q
.             =  xri://@a*a/!b!b/c*c/
./            =  xri://@a*a/!b!b/c*c/
..            =  xri://@a*a/!b!b/
../           =  xri://@a*a/!b!b/
../!g!g       =  xri://@a*a/!b!b/!g!g
../..         =  xri://@a*a/
../../        =  xri://@a*a/
../../!g!g    =  xri://@a*a/!g!g
```

2.4.2.2 Abnormal Examples

As in IRIs and URIs, the ".." syntax cannot be used to change the authority component of an XRI.

```
../../../!g!g    =  xri://@a*a/!g!g
../../../../!g!g =  xri://@a*a/!g!g
```

As in IRIs and URIs, "." and ".." have a special meaning only when they appear as complete path segments.

```
/./!g!g          =  xri://@a*a/!g!g
/../!g!g         =  xri://@a*a/!g!g
!g!g.            =  xri://@a*a/!b!b/c*c/!g!g.
.!g!g            =  xri://@a*a/!b!b/c*c/.!g!g
!g!g..           =  xri://@a*a/!b!b/c*c/!g!g..
..!g!g           =  xri://@a*a/!b!b/c*c/..!g!g
```

XRI parsers, like IRI and URI parsers, must be prepared for superfluous or nonsensical uses of "." and "..".

```
./../!g!g        =  xri://@a*a/!b!b/!g!g
./!g!g/.         =  xri://@a*a/!b!b/c*c/!g!g/
!g!g/./h         =  xri://@a*a/!b!b/c*c/!g!g/h
!g!g/../h        =  xri://@a*a/!b!b/c*c/h
!g!g;x=1/./y     =  xri://@a*a/!b!b/c*c/!g!g;x=1/y
!g!g;x=1/../y    =  xri://@a*a/!b!b/c*c/y
```

XRI parsers, like IRI and URI parsers, must take care to separate the reference's query and/or fragment components from the path component before merging it with the base path and removing dot-segments.

```
!g!g?y/./x       =  xri://@a*a/!b!b/c*c/!g!g?y/./x
!g!g?y/../x      =  xri://@a*a/!b!b/c*c/!g!g?y/../x
!g!g#s/./x       =  xri://@a*a/!b!b/c*c/!g!g#s/./x
!g!g#s/../x      =  xri://@a*a/!b!b/c*c/!g!g#s/../x
```

2.4.3 Leading Segments Containing a Colon

[URI] points out that relative URI references with an initial segment containing a colon may be subject to misinterpretation:

> "A path segment that contains a colon character (e.g., 'this:that') cannot be used as the first segment of a relative-path reference because it would be mistaken for a scheme name. Such a segment must be preceded by a dot-segment (e.g., './this:that') to make a relative-path reference."

Relative XRI references can be similarly misinterpreted. If any segment prior to the first slash ("/") character in a relative XRI reference contains a colon, the relative XRI reference must be rewritten to begin either with "*", if appropriate, or "./". Thus, "a:b" becomes either "*a:b" or "./a:b".

2.4.4 Leading Segments Beginning with a Cross-Reference

A path segment that begins with a cross-reference cannot be used as the first segment of a relative reference because it would be mistaken for an xref-authority. As with a leading segment containing a colon, such a segment must be preceded with "./" to make a relative XRI reference.

2.5 Normalization and Comparison

In general, the normalization and comparison rules for generic IRIs and URIs specified in Section 5 of **[IRI]** and Section 6 of **[URI]** apply to XRIs. This section describes a number of additional XRI-specific rules for normalization and comparison. To reduce the requirements imposed upon a minimally conforming processor, the majority of these rules are RECOMMENDED rather than REQUIRED. An implementation that fails to observe them, however, may frequently treat two XRIs as non-equal when in fact they are equal.

Each application that uses XRI references MAY define additional equivalence rules as appropriate. Due to the level of abstraction XRIs provide, such higher-order equivalence rules may be based on indirect comparisons or specified XRI-to-XRI mappings (for example, mappings of reassignable XRIs to persistent XRIs).

2.5.1 Case

The following rules regarding case sensitivity SHOULD be applied in XRI comparisons.

- Comparison of the scheme component of XRIs and all IRIs used as cross-references is case-insensitive.
- Comparison of authority components (section 2.2.1) is case-insensitive as defined in **[IRI]**.
- As specified in section 2.1.4, comparison of characters in a percent-encoding construction is case-insensitive for the hexadecimal digits "A" through "F", i.e. "%ab" is equivalent to "%AB".

2.5.2 Encoding, Percent-Encoding, and Transformations

- Two XRIs MUST be considered equivalent if they are character-for-character equivalent. Therefore, they are also equivalent if they are byte-for-byte equivalent and use the same character encoding.
- Two XRIs that differ only in whether unreserved characters are percent-encoded SHOULD be considered equivalent. If one XRI percent-encodes one or more unreserved characters, and another XRI differs only in that the same characters are not percent-encoded, they are equivalent.
- All forms of an XRI during the transformation process described in section 2.3.1 SHOULD be considered equivalent, assuming the same XRI metadata is inserted as described in section 2.3.1.

2.5.3 Optional Syntax

- An “xri-segment” (section 2.2.3) that omits the optional leading star (“*”) SHOULD be considered equivalent to the same “xri-segment” prefixed with an star. For example the segment “/foo*bar” is equivalent to the segment “/*foo*bar”.

2.5.4 Cross-References

- If an XRI contains a cross-reference, the rules in this section SHOULD be applied recursively to each cross-reference. For example, the following two XRIs should be considered equivalent:

```
xri://@example/(+example/(+foo))
xri://@example/(+Example/(+FOO))
```

- From the standpoint of XRI syntax, all cross-references beginning with the GCS “$” symbol SHOULD be considered significant unless stated otherwise in the governing specification, for example *Extensible Resource Identifier (XRI) Metadata V2.0* **[XRIMetadata]**. See section 2.1.4.1.

2.5.5 Canonicalization

In general, XRI references do not have a single canonical form. This is particularly true for XRI references that contain IRI cross-references, since many URI schemes, including the HTTP scheme, do not define a canonical form. Additionally, the authority for a particular segment of an XRI reference may define its own rules with respect to case-sensitivity, optional or implicit syntax etc., so canonicalization of those segments is outside the scope of this specification.

It is nevertheless useful to define guidelines for making XRI references reasonably canonical. XRI references that follow these guidelines will be more consistent in presentation, simpler to process, less prone to false-negative comparisons, and more easily cached. To that end, unless there is a compelling reason to do otherwise, XRI references SHOULD be provided in a form in which:

- The optional “xri://” prefix is included,
- The scheme is specified in lowercase,
- The authority component is specified in lowercase,
- Percent-encoding uses uppercase A through F,
- If optional, the leading star in xri-segments is omitted,
- Unnecessary percent-encoding is not present,
- /./ and /../ are absent in absolute XRIs, and
- Cross-references are reasonably canonical with respect to their schemes.

Table 2 illustrates the application of these rules. Although the XRIs in the first and second columns are equivalent, the form in the second column is recommended.

Avoid	Recommended	Comment
@example	xri://@example	Add optional “xri://”
XRI://@example	xri://@example	Lowercase “xri”
xri://@Example	xri://@example	Lowercase authority
xri://@example%2f	xri://@example%2F	Uppercase percent-encoding
xri://@example/*abc	xri://@example/abc	Remove optional leading star
xri://@ex%61mple	xri://@example	Remove unnecessary percent-encoding
xri://@example/./abc	xri://@example/abc	Avoid /./ and /../ in absolute XRIs

Table 2: Examples of XRI canonicalization recommendations.

3 Security and Data Protection Considerations

To a great extent, XRI syntax has the same security considerations as **[IRI]** and **[URI]**. In particular the material in **[URI]**, section 7, *Security Considerations*, includes a discussion of the following topics:

- Reliability and Consistency
- Malicious Construction
- Back-End Transcoding
- Rare IP Address Formats
- Sensitive Information
- Semantic Attacks

This material notes that “a URI does not in itself pose a direct security threat.” In the case of XRIs, this statement remains true only in legacy environments. As noted below, it may not be true for new infrastructure that builds on the extensibility of XRI architecture. In particular the following features of XRIs deserve special mention.

3.1 Cross-References

Since cross-references in an XRI can reference other URI schemes, implementation must carefully consider the relevant security considerations for those referenced schemes.

3.2 XRI Metadata

The use of cross-references employing the GCS “$” symbol for encoding XRI metadata in an XRI (section 2.1.4.1) may involve other security and data protection considerations that are outside the scope of this specification. These considerations are addressed in *Extensible Resource Identifier (XRI) Metadata V2.0* **[XRIMetadata]**.

3.3 Spoofing and Homographic Attacks

One particularly important security consideration is spoofing, covered first in **[URI]** and more thoroughly in **[IRI]** Section 7.5. Spoofing is a semantic attack in which an identifier is deliberately constructed to deceive the user into believing it represents one resource when in fact it represents another. With IRIs in particular, a common example of such an attack is using “homographic” characters (characters from different scripts whose visual appearance is nearly or perfectly identical, e.g., the Latin "A", the Greek "Alpha", and the Cyrillic "A".)

Spoofing has already been used extensively in email "phishing" attacks. As more browsers add support for Internationalized Domain Names (IDN), it is also beginning to appear in online Web links ("pharming"). Not only are some users less suspicious of URIs on the Web, but the attacker may even obtain a corresponding SSL/TLS certificate for the deceptive URI or IRI to make the fraudulent site look completely secure and legitimate.

To help prevent this problem, XRI registries SHOULD institute policies preventing the registration of deceptive XRIs, and user agents that process XRIs SHOULD incorporate safeguards such as warning users when XRIs contain common homographic characters.

3.4 UTF-8 Attacks

Since XRIs incorporate the use of UTF-8 as specified by **[IRI]**, they can also be subject to UTF-8 parsing attacks as described in section 10 of **[RFC3629]**:

> “Implementers of UTF-8 need to consider the security aspects of how they handle illegal UTF-8 sequences. It is conceivable that in some circumstances an attacker would be able to exploit an incautious UTF-8 parser by sending it an octet sequence that is not permitted by the UTF-8 syntax.”

For more information on these attacks, see section 10 of **[RFC3629]**.

3.5 XRI Usage in Evolving Infrastructure

As XRIs are adopted as abstract identifiers, it is anticipated that new services will be developed that take advantage of their extensibility. In particular, XRIs may enable new solutions to security and data protection problems at the resource identifier level that are not possible using existing URI schemes.

For example, XRI cross-reference syntax permits the inclusion of identifier metadata such as an encrypted or integrity-checked path, query or fragment. Cross-references can also be used to indicate methods of obfuscating, proxying or redirecting resolution to prevent the exposure of private or sensitive data.

A complete discussion of this topic is beyond the scope of this document. However, as a consequence of XRI extensibility, it is not possible to make definitive statements regarding all security and data protection considerations related to XRIs. New XRI-producing or consuming applications should include independent security reviews for the specific contexts in which they will be used.

4 References

4.1 Normative

[IRI] M. Duerst, 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, 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] Duerst, M. and 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. Duerst, *Unicode Normalization Forms*, http://www.unicode.org/unicode/reports/tr15/tr15-23.html, April 17, 2003.

[XRIMetadata] D. Reed, *Extensible Resource Identifier (XRI) Metadata V2.0*, http://docs.oasis-open.org/xri/xri/V2.0/xri-metadata-V2.0-cd-01.pdf, March 2005.

[XRIResolution] G. Wachob, *Extensible Resource Identifier (XRI) Resolution V2.0*, http://docs.oasis-open.org/xri/xri/V2.0/xri-resolution-V2.0-cd-01.pdf, March 2005.

4.2 Informative

[XRIIntro] D. Reed, D. McAlpin, *Introduction to XRIs*, http://docs.oasis-open.org/xri/xri/V2.0/xri-intro-V2.0.pdf, Work-In-Progress, March 2005.

[XRIReqs] G. Wachob, D. Reed, M. Le Maitre, D. McAlpin, D. McPherson, *Extensible Resource Identifier (XRI) Requirements and Glossary v1.0*, http://www.oasis-open.org/apps/org/workgroup/xri/download.php/2523/xri-requirements-and-glossary-v1.0.doc, June 2003.

Appendix A. Collected ABNF for XRI (Normative)

This section contains the complete ABNF for XRI syntax. XRI productions use green shading, while productions inherited from IRI use yellow shading. A valid XRI MUST conform to this ABNF.

```
XRI                = [ "xri://" ] xri-hier-part [ "?" iquery ]
                     [ "#" ifragment ]

xri-hier-part      = ( xri-authority / iauthority ) [ xri-path-absolute ]
                     / ipath-empty

XRI-reference      = XRI
                     / relative-XRI-ref

absolute-XRI       = [ "xri://" ] xri-hier-part [ "?" iquery ]

xri-value          = xri-no-scheme / relative-XRI-ref

xri-no-scheme      = xri-hier-part [ "?" iquery ]
                     [ "#" ifragment ]

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

xri-authority      = gcs-authority
                     / xref-authority

gcs-authority      = pgcs-authority / rgcs-authority

pgcs-authority     = "!" xri-subseg-pt-nz *xri-subseg

rgcs-authority     = rgcs-char xri-segment

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

xref-authority     = xref *xri-subseg

xref               = "(" ( XRI-reference / IRI ) ")"

xri-path           = xri-path-absolute
                     / xri-path-noscheme
                     / ipath-empty

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

xri-path-noscheme = xri-subseg-od-nx *xri-subseg-nc *( "/" xri-segment )

xri-segment        = xri-subseg-od *xri-subseg

xri-segment-nz     = xri-subseg-od-nz *xri-subseg

xri-subseg         = ( "*" / "!" ) (xref / *xri-pchar)

xri-subseg-nc      = ( "*" / "!" ) (xref / *xri-pchar-nc)

xri-subseg-od      = [ "*" / "!" ] (xref / *xri-pchar)
```

```
xri-subseg-od-nz  = [ "*" / "!" ] (xref / 1*xri-pchar)

xri-subseg-od-nx  = [ "*" / "!" ] 1*xri-pchar-nc

xri-subseg-pt-nz  = "!" (xref / 1*xri-pchar)

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

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

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 / ":" / "@"

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

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

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 / "-" / "." / "_" / "~"
```

Appendix B. Transforming HTTP IRIs to XRIs (Non-Normative)

To leverage existing infrastructure, it may sometimes be useful to convert HTTP IRIs into XRIs. Because XRI syntax is, for the most part, a superset of generic IRI syntax, the majority of HTTP IRIs can be converted to valid XRIs simply by replacing the scheme name “http” with “xri”. Generally the authority component of the resulting XRI will be properly interpreted as an IRI authority. There are some cases, however, in which a legal authority component in an IRI will be interpreted as an XRI authority rather than an IRI authority when the IRI is converted to an XRI. For example,

```
http://!!1/example
```

is a legal IRI. Converted to an XRI, it would become

```
xri://!!1/example
```

The authority “!!1” matches both the “xri-authority” and the “iauthority” ABNF productions. It would be interpreted as an XRI authority, however, based on the “first-match-wins” rule used to resolve ambiguities in the ABNF. Section 2.2.1.2 provides other examples of legal IRI authorities that would be interpreted as XRI authorities when used in an XRI. Note, however, that these cases are unlikely to arise in practice, since they typically result in an invalid URI when converted from an IRI.

Special consideration must also be given to HTTP IRIs employing those characters in common between the “sub-delims” production of **[IRI]** and the “xri-gen-delims” production of this specification, namely opening parenthesis (“(“), closing parenthesis (“)”), star (“*”), bang (“!”), dollar sign (“$”), plus sign (“+”) and equals sign (“=”). These characters are reserved as delimiters in HTTP IRIs but have no scheme-specific meaning (i.e., they are only used as delimiters in a manner defined by a local authority). In XRIs, however, these characters do have defined semantics that may or may not match the meaning intended by an IRI author. Conversion of such IRIs to XRIs must be handled on a case-by-case basis.

Appendix C. Glossary

The following definitions are common to this specification, the *XRI Resolution* specification **[XRIResolution]**, and the *XRI Metadata* specification **[XRIMetadata]**.) Note that this glossary supercedes the glossary in **[XRIReqs]**.

Absolute Identifier

An identifier that refers to a resource independent of the current context, i.e., one that establishes a global context. Mutually exclusive with "Relative Identifier."

Abstract Identifier

An identifier that is not directly resolvable to a resource, but is either:

a) a self-reference, because it completely represents a non-network resource and is not further resolvable (see "Self-Reference"), or

b) an indirect reference to a resource, because it must first be resolved to another identifier (either another abstract identifier or a concrete identifier).

A URN as described in **[RFC2141]** is one kind of abstract identifier. Compared to concrete identifiers, abstract identifiers permit additional levels of indirection in referencing resources, which can be useful for a variety of purposes, including persistence, equivalence, human-friendliness, and data protection.

Authority (or Identifier Authority)

In the context of identifiers, an authority is a resource that assigns identifiers to other resources. Note that in URI syntax as defined in **[URI]**, the "authority" production refers explicitly to the top-level authority identified by the segment beginning with "//". Since XRI syntax supports unlimited federation, the term "authority" can technically refer to an identifier authority at any level. However, in the "xri-authority" and "iauthority" productions (section 2.2.1), it explicitly refers to the top-level identifier authority. See also "IRI Authority" and "XRI Authority"

In the context of identifier resolution, an authority is a resource (typically a server) that responds to resolution requests from another resource (typically a client). From this perspective, each sub-segment in the authority segment of an XRI identifies a separate authority.

Base Identifier

An absolute identifier that identifies a context for a relative identifier. Changing the base identifier changes the context of the relative identifier. See "Relative Identifier."

Canonical Form

The form of an identifier after applying transformation rules for the purpose of determining equivalence. See also "Normal Form".

Community (or Identifier Community)

A set of resources that share a common identifier authority, often (but not always) a common root authority. Technically, a set of resources whose identifiers form a directed graph or tree.

Concrete Identifier

An identifier that can be directly resolved to a resource or resource representation, rather than to another identifier. Examples include the MAC address of a networked computer and a phone number that rings directly to a specific device. All concrete identifiers are intended to be resolvable. Contrast with "Abstract Identifier."

Context (or Identifier Context)

The resource of which an identifier is an attribute. For example, in the string of identifiers “a/b/c”, the context of the identifier “b” is the resource identified by “a/”, and the context of the identifier “c” is the resource identified by “a/b/”. Since multiple resources may assign an identifier for a target resource, the resource can be said to be identified in multiple contexts. For absolute identifiers, the context is global, i.e., there is a known starting point, or root. For relative identifiers, the context is implicit. See also “Base Identifier.”

Cross-reference

An identifier assigned in one context that is reused in another context. Cross-references enable the expression of polyarchical relationships (relationships that cross multiple hierarchies – see “Polyarchy”.) Cross-references can be used to identify logically equivalent resources in different domains, authorities, or physical locations. For example, a cross-reference may be used to identify the same logical invoice stored in two accounting systems (the originating system and the receiving system), the same logical Web document stored on multiple proxy servers, the same logical datatype used in multiple databases or XML schemas, or the same logical concept used in multiple taxonomies or ontologies.

In XRI syntax, cross-references are syntactically delimited by enclosing them in parentheses. This is analogous to enclosing a word or phrase in quotation marks in a natural language, such as English, to indicate that the author is referring to it independent of the current context. For example, the phrase “love bird” is quoted in this sentence to indicate that we are *mentioning,* rather than *using,* the phrase - that is, we are referring to it independent of the context of this glossary.

Delegated Identifier

A multi-segment identifier in which segments are assigned by more than one identifier authority. Namespace authority is delegated from one identifier authority to the next. Mutually exclusive with “Local Identifier.”

Federated Identifier

A delegated identifier that spans multiple independent identifier authorities. See also “Delegated Identifier.”

Global Context Symbol (GCS)

A reserved character used at the start of the authority segment of an XRI to establish the global context of an XRI authority. XRI 2.0 defines four Global Context Symbols, which are used to represent persons, organizations, the public, and standards specifications. See section 2.2.1.2.

Hierarchy

A branching tree structure in which all primary relationships are parent-child. (Sibling relationships in a hierarchy are secondary, derived from the parent-child relationships.) URI and IRI syntax has explicit support for hierarchical paths. XRI syntax supports both hierarchical and polyarchical paths. See “Polyarchy” and “Cross-reference.”

Human-Friendly Identifier (HFI)

An identifier containing words or phrases intended to convey meaning in a specific human language which is therefore easy for people to remember and use. Contrast with "Machine-Friendly Identifier."

Identifier

Per **[URI]**, anything that “embodies the information required to distinguish what is being identified from all other things within its scope of identification.” In UML terms, an identifier is an attribute of a resource (the identifier context) that forms an association with another resource (the identifier target). The general term “identifier” does not specify whether the identifier is abstract or concrete, absolute or relative, persistent or reassignable, human-friendly or machine-friendly, delegated or local, hierarchical or polyarchical, or resolvable or self-referential.

I-name

An informal term used to refer to a reassignable XRI; more specifically, an XRI in which at least one sub-segment is reassignable.

I-number

An informal term used to refer to a persistent XRI; more specifically, an XRI in which all sub-segments are persistent. Note that a persistent XRI is not required to be numeric - it may be any text string meeting the XRI ABNF requirements.

IRI (Internationalized Resource Identifier)

IRI is a specification for internationalized URIs developed by the W3C. IRIs specify how to include characters from the Universal Character Set (Unicode/ISO10646) in URIs. The IRI specification **[IRI]** provides a mapping from IRIs to URIs, which allows IRIs to be used instead of URIs where appropriate. This XRI specification defines a similar transformation from XRIs to IRIs for the same reason.

IRI Authority

An identifier authority (see “Authority”) represented by the authority segment of an XRI that does not match the “xri-authority” production but matches the “iauthority” production from **[IRI]**. See section 2.2.1.3. Mutually exclusive with “XRI Authority”.

Local Identifier

Any identifier, or any set of segments in a multi-segment identifier, that is assigned by the same identifier authority. Each of these segments is local to that authority. Mutually exclusive with “Delegated Identifier.”

Machine-Friendly Identifier (MFI)

An identifier containing digits, hexadecimal values, or other character sequences optimized for efficient machine indexing, searching, routing, caching, and resolvability. MFIs generally do not contain human semantics. Compare with "Human-Friendly Identifier."

Normal Form

The character-by-character format of an identifier after encoding, escaping, or other character transformation rules have been applied in order to satisfy syntactic requirements. Three normal forms are defined for XRIs—XRI-normal form, IRI-normal form, and URI-normal form. See section 2.3.1 for details. See also “Canonical Form”.

Path

The relationships between resources defined by a multi-segment identifier. In less strict contexts, the word “path” often refers to the multi-segment identifier itself, or to the resources it represents (such as filesystem directories).

Persistent Identifier

An identifier that is permanently assigned to a resource and intended never to be reassigned to another resource - even if the original resource goes off the network, is terminated, or ceases to exist. A URN as described in **[RFC2141]** is an example of a persistent identifier. Persistent identifiers tend to be machine-friendly identifiers, since human-friendly identifiers often reflect human semantic relationships that may change over time. Mutually exclusive with “Reassignable Identifier.”

Polyarchy

A treelike structure composed of multiple intersecting hierarchies in which primary relationships can cross hierarchies. A polyarchy allows one member to be connected or linked to any other, although, in contrast to a web, the overall structure tends to remain strongly hierarchical. XRIs support polyarchic paths through the use of cross-references. See also “Cross-reference” and “Hierarchy”.

Reassignable Identifier

An identifier that may be reassigned from one resource to another. Example: the domain name “example.com” may be reassigned from ABC Company to XYZ Company, or the email address “mary@example.com” may be reassigned from Mary Smith to Mary Jones. Reassignable identifiers tend to be human-friendly because they often represent the potentially transitory mapping of human semantic relationships onto network resources or resource representations. Mutually exclusive with “Persistent Identifier.”

Relative Identifier

An identifier that refers to a resource only in relationship to a particular context (for example, the current community, the current document, or the current position in a delegated identifier). If the context changes, the identifier’s meaning also changes. A relative identifier can be converted into an absolute identifier by combining it with a base identifier (an absolute identifier that is used to identify a context). See “Base Identifier”. Mutually exclusive with “Absolute Identifier.”

Resolvable Identifier

An identifier that references a network resource or resource representation and that can be resolved into a network endpoint for communicating with the target resource. Mutually exclusive with “Self-Reference.”

Resource

Per **[URI]**, “anything that can be named or described.” Resources are of two types: network resources (those that are network-addressable) and non-network resources (those that exist entirely independent of a network). Network resources are themselves of two types: physical resources (resources physically attached to or operating on the network) or resource representations (see “Resource Representation”).

Resource Representation

A network resource that represents the attributes of another resource. A resource representation may represent either another network resource (such as a machine, service, or application) or a non-network resource (such as a person, organization, or concept).

Segment (or Identifier Segment)

Any syntactically delimited component of an identifier. In generic URI syntax, all segments after the authority portion are delimited by forward slashes (“/segment1/segment2/…”). In XRI syntax, slash segments can be further subdivided into sub-segments called *star segments* (for reassignable identifiers) and *bang segments* (for persistent identifiers). See section 2.2.3. XRI also supports another type of segment called a cross-reference, which is enclosed in parentheses. See “Cross-Reference”.

Self-Reference (or Self-Referential Identifier)

An identifier which is itself the representation of the resource it references. Self-references are typically used to represent non-network resources (e.g., “love”, “Paris”, “the planet Jupiter”) in contexts where this identifier is not intended to be resolved to a separate network representation of that resource. The primary purpose of self-references is to establish equivalence across contexts (see “Cross-References”). Mutually exclusive with “Resolvable Identifier.”

Sub-segment

A syntactically delimited component of an identifier segment (see “Segment”). While URI and IRI syntax define only segments, XRI syntax defines both segments and sub-segments. XRI sub-segments are used to distinguish among persistent identifiers, reassignable identifiers, and cross-references. See sections 2.2.2 and 2.2.3.

Synonym (or Identifier Synonym)

An identifier that is asserted by an identifier authority to be equivalent to another identifier not because of strict literal equivalence, but because it resolves to the same resource.

Target (or Identifier Target)

The resource referenced by an identifier. A target may be either a network resource (including a resource representation) or a non-network resource.

URI (Uniform Resource Identifier)

The standard identifier used in World Wide Web architecture. Starting in 1998, RFC 2396 has been the authoritative specification for URI syntax. In January 2005 it was superseded by RFC 3986 **[URI]**.

XDI (XRI Data Interchange)

A generalized, extensible service for sharing, linking, and synchronizing XML data and metadata associated with XRI-identified resources. XDI is being developed by the OASIS XDI Technical Committee (http://www.oasis-open.org/committees/xdi).

XRI Authority

An identifier authority (see “Authority”) represented by the authority segment of an XRI that begins with either a global context symbol or a cross-reference. See section 2.2.1.1. Mutually exclusive with “IRI Authority.”

XRI Descriptor (XRID)

An XML document returned by an authority in the process of XRI resolution as defined in section 2.2.2 of the *XRI Resolution* specification **[XRIResolution]**.

XRI Reference

A term that includes both absolute and relative XRIs. Used in the same way as “URI reference” and “IRI reference.” Note that to transform an XRI reference into an XRI, it must first be converted to absolute form, which in the case of a relative XRI requires the use of a base XRI to establish context.

Appendix D. Acknowledgments

The editors would like to acknowledge the contributions of the OASIS XRI Technical Committee, whose voting members at the time of publication were:

- Geoffrey Strongin, Advanced Micro Devices
- Ajay Madhok, AmSoft Systems
- Jean-Jacques Dubray, Attachmate
- William Barnhill, Booz Allen and Hamilton
- Drummond Reed, Cordance Corporation
- Marc Le Maitre, Cordance Corporation
- Dave McAlpin, Epok
- Loren West, Epok
- Peter Davis, NeuStar
- Masaki Nishitani, Nomura Research
- Nat Sakimura, Nomura Research
- Tetsu Watanabe, Nomura Research
- Owen Davis, PlaNetwork
- Victor Grey, PlaNetwork
- Fen Labalme, PlaNetwork
- Mike Lindelsee, Visa International
- Gabriel Wachob, Visa International
- Dave Wentker, Visa International
- Bill Washburn, XDI.ORG

The editors also would like to acknowledge the following people for their contributions to previous versions of the OASIS XRI specifications (affiliations listed for OASIS members):

Thomas Bikeev, EAN International; Krishna Sankar, Cisco; Winston Bumpus, Dell; Joseph Moeller, EDS; Steve Green, Epok; Lance Hood, Epok; Adarbad Master, Epok; Davis McPherson, Epok; Chetan Sabnis, Epok; Phillipe LeBlanc, GemPlus; Jim Schreckengast, Gemplus; Xavier Serret, Gemplus; John McGarvey, IBM; Reva Modi, Infosys; Krishnan Rajagopalan, Novell; Tomonori Seki, NRI; James Bryce Clark, OASIS; Marc Stephenson, TSO; Mike Mealling, Verisign; Rajeev Maria, Visa International; Terence Spielman, Visa International; John Veizades, Visa International; Lark Allen, Wave Systems; Michael Willett, Wave Systems; Matthew Dovey; Eamonn Neylon; Mary Nishikawa; Lars Marius Garshol; Norman Paskin; Bernard Vatant.

A special acknowledgement to Jerry Kindall (Epok) for a full editorial review.

Also, the authors of and contributors to the following documents and specifications are acknowledged for the intellectual foundations of the XRI specification:

- RFC 1737
- RFC 2616
- RFC 2718
- RFC 3986 (STD 66) and its predecessor, RFC 2396
- RFC 3987
- XNS

Appendix E. Notices

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's procedures with respect to rights in OASIS specifications can be found at 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 implementors or users of this specification, can be obtained from the OASIS President.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS President.