Extensible Resource Identifier (XRI)  
Generic Syntax and Resolution Specification  

Working Draft, 19 January 2005  

Document identifier:  
xri-syntax-resolution-2.0-wd-02  

Location:  
http://www.oasis-open.org/committees/xri/xri-syntax-resolution-2.0-wd  

Editors:  
Gabe Wachob, Visa International <gwachob@visa.com>  
Drummond Reed, OneName <drummond.reed@onename.com>  
Dave McAlpin, Epok <dave.mcalpin@epok.net>  
Mike Lindelsee, Visa International <mlindels@visa.com>  
Peter Davis, Neustar <peter.davis@neustar.biz>  
Nat Sakimura, NRI <n-sakimura@nri.co.jp>  

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 the "XRI Primer" at  

Status:  
This document is a Working Draft.  
Committee members should send comments on this specification to the xri@lists.oasis-open.org list. Others should subscribe to and send comments to the xri-comment@lists.oasis-open.org list. To subscribe, send an email message to xri-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.  
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 XRI TC web page (http://www.oasis-open.org/committees/xri/).  
The errata page for this specification is at http://www.oasis-open.org/committees/xri/xri-syntax-resolution-2.0-errata.
## Table of Contents

34

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table of Contents</strong></td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>Introduction</td>
</tr>
<tr>
<td>36</td>
<td>1.1 Overview of XRIs</td>
</tr>
<tr>
<td>37</td>
<td>1.1.1 Generic Syntax</td>
</tr>
<tr>
<td>38</td>
<td>1.1.2 Usage Examples</td>
</tr>
<tr>
<td>39</td>
<td>1.1.3 URI, URL, URN, and XRI</td>
</tr>
<tr>
<td>40</td>
<td>1.2 Design Considerations</td>
</tr>
<tr>
<td>41</td>
<td>1.2.1 Abstraction and Independence</td>
</tr>
<tr>
<td>42</td>
<td>1.2.2 Persistence and Reassignability</td>
</tr>
<tr>
<td>43</td>
<td>1.2.3 Human-Friendliness and Machine-Friendliness</td>
</tr>
<tr>
<td>44</td>
<td>1.2.4 Internationalization</td>
</tr>
<tr>
<td>45</td>
<td>1.2.5 Cross-Context Identification</td>
</tr>
<tr>
<td>46</td>
<td>1.2.6 Authority, Delegation, and Federation</td>
</tr>
<tr>
<td>47</td>
<td>1.2.7 Security and Privacy</td>
</tr>
<tr>
<td>48</td>
<td>1.2.8 Extensibility</td>
</tr>
<tr>
<td>49</td>
<td>1.3 Terminology and Notation</td>
</tr>
<tr>
<td>50</td>
<td>1.3.1 Keywords</td>
</tr>
<tr>
<td>51</td>
<td>1.3.2 Syntax Notation</td>
</tr>
<tr>
<td>52</td>
<td>1.3.3 Glossary</td>
</tr>
<tr>
<td>53</td>
<td>2 Syntax</td>
</tr>
<tr>
<td>54</td>
<td>2.1 Syntax Notation</td>
</tr>
<tr>
<td>55</td>
<td>2.2 Characters</td>
</tr>
<tr>
<td>56</td>
<td>2.2.1 Character Encoding</td>
</tr>
<tr>
<td>57</td>
<td>2.2.2 Reserved Characters</td>
</tr>
<tr>
<td>58</td>
<td>2.2.3 Unreserved Characters</td>
</tr>
<tr>
<td>59</td>
<td>2.2.4 Percent-encoded Characters</td>
</tr>
<tr>
<td>60</td>
<td>2.2.4.1 Encoding XRI Metadata</td>
</tr>
<tr>
<td>61</td>
<td>2.2.5 Excluded Characters</td>
</tr>
<tr>
<td>62</td>
<td>2.3 Syntax Components</td>
</tr>
<tr>
<td>63</td>
<td>2.3.1 Authority</td>
</tr>
<tr>
<td>64</td>
<td>2.3.1.1 XRI Authority</td>
</tr>
<tr>
<td>65</td>
<td>2.3.1.2 IRI Authority</td>
</tr>
<tr>
<td>66</td>
<td>2.3.1.3 Global Context Symbols (GCS)</td>
</tr>
<tr>
<td>67</td>
<td>2.3.1.4 Cross-References</td>
</tr>
<tr>
<td>68</td>
<td>2.3.1.5 Self-References</td>
</tr>
<tr>
<td>69</td>
<td>2.3.2 Path</td>
</tr>
<tr>
<td>70</td>
<td>2.3.3 Query</td>
</tr>
<tr>
<td>71</td>
<td>2.3.4 Fragment</td>
</tr>
<tr>
<td>72</td>
<td>2.4 Transformations</td>
</tr>
<tr>
<td>73</td>
<td>2.4.1 Transforming XRIs into IRIs and URIs</td>
</tr>
<tr>
<td>74</td>
<td>2.4.2 Escaping Rules for XRI Syntax</td>
</tr>
<tr>
<td>75</td>
<td>2.4.3 Transforming IRIs Back into XRIs</td>
</tr>
<tr>
<td>76</td>
<td>2.5 Relative XRI References .................................................................................................................. 23</td>
</tr>
<tr>
<td>77</td>
<td>2.5.1 Reference Resolution .................................................................................................................. 23</td>
</tr>
<tr>
<td>78</td>
<td>2.5.2 Reference Resolution Examples ............................................................................................... 23</td>
</tr>
<tr>
<td>79</td>
<td>2.5.3 Normal Examples ......................................................................................................................... 23</td>
</tr>
<tr>
<td>80</td>
<td>2.5.4 Abnormal Examples ..................................................................................................................... 24</td>
</tr>
<tr>
<td>81</td>
<td>2.5.5 Leading Segments Containing a Colon ...................................................................................... 25</td>
</tr>
<tr>
<td>82</td>
<td>2.6 Normalization and Comparison ..................................................................................................... 25</td>
</tr>
<tr>
<td>83</td>
<td>2.6.1 Case ........................................................................................................................................ 25</td>
</tr>
<tr>
<td>84</td>
<td>2.6.2 Encoding, Percent-encoding, and Transformations ...................................................................... 25</td>
</tr>
<tr>
<td>85</td>
<td>2.6.3 Optional Syntax .......................................................................................................................... 25</td>
</tr>
<tr>
<td>86</td>
<td>2.6.4 Cross-References ........................................................................................................................ 26</td>
</tr>
<tr>
<td>87</td>
<td>2.6.5 Canonicalization .......................................................................................................................... 26</td>
</tr>
<tr>
<td>88</td>
<td>3 Security and Data Protection ............................................................................................................. 28</td>
</tr>
<tr>
<td>89</td>
<td>3.1 Secure Resolution .......................................................................................................................... 28</td>
</tr>
<tr>
<td>90</td>
<td>3.2 XRI Metadata .................................................................................................................................. 28</td>
</tr>
<tr>
<td>91</td>
<td>3.3 XRI Usage in Legacy Infrastructure .............................................................................................. 28</td>
</tr>
<tr>
<td>92</td>
<td>3.4 XRI Usage in Evolving Infrastructure ............................................................................................ 28</td>
</tr>
<tr>
<td>93</td>
<td>4 References ..................................................................................................................................... 29</td>
</tr>
<tr>
<td>94</td>
<td>4.1 Normative .................................................................................................................................... 29</td>
</tr>
<tr>
<td>95</td>
<td>4.2 Informative .................................................................................................................................. 30</td>
</tr>
<tr>
<td>96</td>
<td>Appendix A. Collected ABNF for XRI (Normative) ............................................................................ 31</td>
</tr>
<tr>
<td>97</td>
<td>Appendix B. Transforming HTTP URIs to XRIs (Non-Normative) ...................................................... 34</td>
</tr>
<tr>
<td>98</td>
<td>Appendix C. Acknowledgments ............................................................................................................ 35</td>
</tr>
<tr>
<td>99</td>
<td>Appendix D. Revision History ............................................................................................................. 36</td>
</tr>
<tr>
<td>100</td>
<td>Appendix E. Notices ............................................................................................................................ 37</td>
</tr>
</tbody>
</table>
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 top of 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]

In the same way IRIs added syntactic elements and extended the unreserved character set to include characters beyond those allowed in generic URIs, XRIs extend the syntactic elements (but not the character set) allowed in IRIs. To accommodate applications that expect IRIs or generic URIs, this specification defines rules for transforming an XRI into a syntactically legal IRI or URI.

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 ABNF for the XRI scheme and associated normalization and equivalence rules. Three additional specifications complete the XRI 2.0 suite:

- **XRI 2.0 Resolution** 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 2.0 Metadata** specifies a small set of standard metadata identifiers registered under the XRI global context symbol "$" which may be used to describe the contents of an XRI.
- **XRI 2.0 Primer** provides an introduction to XRI 2.0 syntax, resolution, and metadata via a set of practical examples.

1.1.1 Generic Syntax

XRI syntax is designed to be as simple and extensible as IRI and URI syntax. A fully-qualified XRI consists of the scheme name “xri:” followed by the same four optional components as a generic 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, which inherit directly from generic URI syntax, can be transformed to valid XRIs simply by changing the scheme from "http" to "xri". The rules for this transformation are summarized in Appendix B, "Transforming HTTP URIs to XRIs".

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

1. **Persistent and reassignable segments.** Generic URI syntax does not distinguish between persistent and reassignable identifiers. XRI syntax allows the internal components of an XRI to be explicitly designated as either persistent or reassignable.

2. **Cross-references.** Generic URI syntax does not provide a way to share identifiers across contexts ("polyarchy"). This capability is particularly useful with abstract identifiers, for example, to share the generic name of a resource, or to share standardized identifier metadata such as version indicators. For this reason, XRI syntax allows XRIs (and URIs) to be shared across contexts by means of parenthetical nesting.

3. **Self-references.** Generic URI syntax does not provide a way to indicate whether or not a URI is intended for resolution. Since an XRI may itself be the full representation of an abstract non-network resource (for example, concepts like "love," "honesty," or "user-friendliness"), XRI syntax provides a way to express self-reference.

4. **Global context symbols.** While XRI syntax supports the same generic syntax used in URIs for DNS and IP authorities, it also provides shorthand symbols for establishing the abstract global context of an identifier.

5. **Standardized federation.** In this context, federated identifiers are identifiers 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 the syntax for federation of both persistent and reassignable identifiers at any level of the path.

### 1.1.2 Usage Examples

The following scenarios illustrate several ways in which XRIs might be used. For more examples, see the XRI Primer.

An individual, Mary Doe, chooses to register a human-friendly XRI to use as a context-independent identifier for many different types of online interactions. If registered in a personal context, this XRI might look like:

```plaintext
=Mary.Doe
```

Alternatively, if registered as a delegated identifier in the organizational context of her XRI service provider, Mary's XRI might look like:

```plaintext
@Service.Provider*Mary.Doe
```

(Note that due to the starting symbols used in these human-friendly XRIs, the "xri://" prefix is not required, similar to the way a prefix is not typically required to recognize an Internet email address.)

In either case, Mary can now use these XRIs to identify herself during an e-commerce transaction at a website that supports XRIs. The website can then use XRI resolution to identify a service offered by Mary's XRI service provider for requesting data Mary has associated with her XRI, such as her shipping address, email address, or telephone number. Separating this concrete data from the abstract XRI allows Mary (or her delegate) to better control its access and usage.
An XRI resolution request against either of the XRIs above might return the following XML document as an XRID (XRI Descriptor). From this XRID, the website can request data from Mary's XRI service provider via any of the listed services.

```
<XRID>
  insert example 2.0-compliant XRID for *Mary.Smith here>
</XRID>
```

For its part, a website may also wish to register one or more XRIs in order to establish a persistent identity that is not dependent on its domain name, legal name, or any other semantic identifier. For example, if Example.com is bought by Big-Site.net and changes its name, the web links to Example.com will stop working. With XRIs, Example.com could register one or more reassignable XRIs in an organizational context such as:

```
@Example
@Example.com
```

But instead of having these XRIs resolve directly to a DNS name or IP address, Example.com could have them resolve to a persistent XRI such as:

```
xri://!!1000/(@!(1234!5678!ABCD))
```

This persistent XRI could then resolve to a concrete URI, such as “http://example.com.” Now, when Big-Site.net buys Example.com, Big-Site.net needs only change the resolution value for Example.com’s persistent XRI to point to “http://big-site.net.” All links that use this persistent XRI will continue to function perfectly even though the semantic names have changed completely.

Furthermore, since the generic XRI resolution protocol is HTTP-based, existing web software can use a server-side HTTP resolver for these XRIs. For example, the following three HTTP URIs could resolve to the same target resource as the three XRIs above:

```
http://public.resolver.net/@Example
http://public.resolver.net/@Example.com
http://public.resolver.net/!!1000/(@!(1234!5678!ABCD))
```

XRIs can also help Example.com solve a different problem: how to easily identify resources on its website or network in a standard way so that users or software programs can locate them automatically. This can be done with XRIs that express a general context such as:

```
+technical.support
+customer.service
+press.releases
+investor.relations
```

Example.com can reuse these generic identifiers in the context of its own website using XRI cross-reference syntax as shown below:

```
xri://@example.com/(+technical.support)
xri://@example.com/(+customer.service)
xri://@example.com/(+press.releases)
xri://@example.com/(+investor.relations)
```
Of course the same resource may have generic identifiers in many different languages, and even multiple generic identifiers in the same language (e.g., "+technical.support", "+tech.support", "+help.desk"). A general context service could use mappings of reassignable XRIs to persistent XRIs to help solve this problem. For example, it could establish the following as XRI "synonyms" for the same resource:

```plaintext
xri://@example.com/(+technical.support)
xri://@example.com/(+tech.support)
xri://@example.com/(+support.technicale)
xri://@example.com/(+!!(7E5!188C)
```

Browsers, websites, search engines, or other agents could then use XRI general context services to assist users in locating desired resources much the same way dictionary or thesaurus programs assist users in locating desired words or phrases.

### 1.1.3 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 need to be classified as being just one of ‘name’ or ‘locators’. 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 any quality of the scheme.”

The XRI scheme explicitly implements this philosophy. XRIs can be used either as direct "names" or "locators" for resources, including other XRIs. The XRI scheme also includes syntax for distinguishing whether an XRI is intended only for identification or also for resolution (see 2.3.1.5, Self-References).

### 1.2 Design Considerations

The full set of requirements for XRI syntax and resolution is documented in “XRI Requirements and Glossary v1.0. [XRIReqs] A synopsis of the major design considerations is included here.

#### 1.2.1 Abstraction and Independence

The overarching requirement of the XRI design is that XRI syntax be fully abstract (i.e., independent of resource location, network, application, transport protocol, type, or security method). Although XRI syntax may be extended for specific uses, the generic XRI syntax is designed to represent logical associations between resources and therefore to be portable across all networks, directories, domains, and applications.
1.2.2 Persistence and Reassignability

XRI syntax and resolution is designed to express and resolve fully persistent identifiers, fully reassignable identifiers, or any combination of persistent and reassignable identifier segments.

1.2.3 Human-Friendliness and Machine-Friendliness

XRI syntax and resolution is designed to support both human-friendly identifiers (HFI—those optimized for human readability, memorability, and usability) and machine-friendly identifiers (MFI—those optimized for machine processing and network efficiency). XRI syntax allows any combination of HFI and MFI components within a single XRI.

1.2.4 Internationalization

XRIs are designed to be rendered in the natural language of the intended user. They therefore employ the Unicode character set [Unicode] and provide syntactical support for expressing optional language-dependent context metadata. As a result, XRIs extend the virtues of human readability, memorability and usability to users of all human languages.

1.2.5 Cross-Context Identification

XRI syntax and resolution is designed to allow one identifier to be used in the context of another identifier (i.e., for an XRI or a URI to be contained within another XRI). These embedded identifiers, called cross-references, allow XRIs to represent polyarchies (relationships that cross hierarchies). They are also the key to XRI extensibility.

1.2.6 Authority, Delegation, and Federation

XRI syntax and resolution are designed to allow any resource to serve as an identifier authority, and for any authority to delegate to any other authority at any level of the path. Thus XRI design imposes no specific delegation model, network topology, or federation structure.

1.2.7 Security and Privacy

XRI syntax and resolution is designed to be adapted to any security model, method, or infrastructure, as well as to any privacy policy or framework. XRIs never require sensitive data, such as passwords or account numbers, to be included in an identifier. If a particular application ever needs to include such data in an XRI, the syntax permits encryption and obfuscation of identifier segments for enhanced security and privacy.

1.2.8 Extensibility

The XRI scheme is designed to provide the same interoperable extensibility for identifiers that XML provides for markup languages. In other words, by design, the XRI scheme should be able to be extended and specialized by various identifier authorities, and these extensions and specializations should be interoperable.

1.3 Terminology and Notation

1.3.1 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.
1.3.2 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 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 URI 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.

1.3.3 Glossary

The following definitions are central to this specification. 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 an example of an abstract identifier. Abstract identifiers provide 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)

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 2.3.1), it explicitly refers to the top-level identifier authority.

Base Identifier

An absolute identifier that identifies the current context for a relative identifier. See “Relative Identifier.”
**Canonical Form**

The state of an identifier after applying transformation rules for the purpose of determining equivalence. See also “Normal Form”.

**Community (or Identifier Community)**

The set of resources that share a common identifier authority, often (but not always) a common root authority. Technically, the 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, a phone number that rings directly to a specific device, and a postal address that is not a forwarding address. All concrete identifiers are intended to be resolvable identifiers. 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. For relative identifiers, the context is implicit.

**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 page 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. In the English language this is is analogous to enclosing a word or phrase in quotes 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 its meaning independent of this context.

**Delegated Identifier**

A multi-segment identifier in which some segments are assigned by different identifier authorities. Mutually exclusive with “Local Identifier.”

**Federated Identifier**

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

**Hierarchy**

A branching tree structure in which all relationships are parent/child. URI and IRI syntax has explicit support for hierarchical paths. XRIs 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 and thus be 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

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

I-number

A general term used to refer to a persistent XRI; more specifically, an XRI in which all subsegments are persistent. Note that a persistent XRI is not required to be a numeric value - 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) into URIs. The IRI specification [IRI] provides a mapping from IRIs to URIs, which allows IRIs to be used instead of URIs where appropriate.

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, hex 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—percent-encoded normal form, IRI normal form and URI normal form. See section 2.4.1 for details. See also "Canonical Form".

Path

The set of relationships between resources defined by a multi-segment identifier.

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 no longer exists. A URN as described in [link] is an example of a persistent identifier. Persistent identifiers tend to be machine-friendly identifiers, since human-friendly identifiers typically reflect human semantic relationships that may change over time. Mutually exclusive with "Reassignable Identifier."
Polyarchy

A structure in which relationships can cross multiple hierarchies, vs. the strictly parent/child relationships supported by hierarchies. XRI supports polyarchic paths through the use of cross-references. See “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 “john@example.com” may be reassigned from John Smith to John Jones. Reassignable identifiers tend to be human-friendly identifiers 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 the current context (for example, the current community, the current document, or the current position in a delegated identifier). A relative identifier can be converted into an absolute identifier by combining it with a base identifier (an absolute identifier that identifies the current context of the relative identifier.) 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

Any syntactically delimited portion 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 2.3.2. 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.”
Synonym
An identifier that is asserted by an identifier authority to be equivalent to another identifier not because of character-by-character equivalence, but because it resolves to an equivalent 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. Since 1998, RFC 2396 has been the authoritative specification for URI syntax. In [month, year] it was superceded by [RFC XXXX]. XRIs build on URI syntax. See section [link].

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. See [link XDI].

XRI Reference
A term that includes both absolute and relative XRIs. Used the same way as "URI reference" and "IRI reference". Note that to transform an XRI reference into an XRI, it must be converted into its absolute form.
2 Syntax

2.1 Syntax Notation

This specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC2234], including the following core ABNF syntax rules defined by that specification: ALPHA (letters), DIGIT (decimal digits), and HEXDIG (hexadecimal digits). The complete XRI syntax is collected in Appendix A.

2.2 Characters

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

2.2.1 Character Encoding

The basic character encoding of XRI is UTF-8, as recommended by [RFC2718]. When an XRI is presented as a human readable identifier, the representation of the XRI in the underlying document should use the character encoding of the underlying document. However, this string must be converted to UTF-8 before any processing external to the underlying document.

Note that not all ASCII sequences can be derived from UTF-8 sequences. A valid XRI character sequence MUST be derivable by unescaping an equivalent UTF-8 sequence. For example, the ASCII sequence '%FC', which would represent U+00FC LATIN SMALL LETTER U WITH DIAERESIS in an iso-8859-1 encoding, when unescaped will not result in a valid UTF-8 sequence.

2.2.2 Reserved Characters

The XRI reserved character set is the combination of xri-gen-delims and xri-sub-delims and is the same as the reserved character set defined by [URI] and [IRI]. 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.

```plaintext
xri-reserved = xri-gen-delims / xri-sub-delims
xri-gen-delims = ";" / ";" / ";" / ";" / "[" / "]" / "(" / ")" / "*" / gcs-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 2.2.4, “Percent-Encoded Characters”. XRIs that differ in the percent-encoding of a reserved character are not equivalent.

2.2.3 Unreserved Characters

The characters that are allowed in XRIs and do not have a reserved purpose are called unreserved. XRI has the same set of unreserved characters as [IRI] (iunreserved).

```plaintext
iunreserved = ALPHA / DIGIT / ";" / ";" / ";" / ";" / "*" / ucschar
```
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 2.6, “Normalization and Comparison”), so unreserved characters should not be percent-encoded except as required in URI-normal form as described in section 2.4.1.

### 2.2.4 Percent-encoded Characters

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

An XRI thus percent-encoded is said to be in percent-encoded normal form. This does not imply that it is necessarily a valid IRI or URI. Rules for converting an XRI into a valid IRI or URI are discussed in section 2.4.1. An XRI is in percent-encoded normal form if it is minimally percent-encoded and matches the ABNF provided in this document, but it is a valid IRI or URI only after it is percent-encoded according to the transformation described in section 2.4.1.

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

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

Note that the % symbol used by itself in an XRI (i.e. as data and not to introduce a percent-encoded triplet) must be percent-encoded as described in section 2.2.5.

### 2.2.4.1 Encoding XRI Metadata

In some cases, the transformation from an identifier in its native language and display format into an XRI in percent-encoded 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 could cause the resulting XRI to be ambiguous.

Another case is when the normalization or canonicalization rules of a particular identifier authority do not permit the inclusion of mixed case letters, or certain punctuation in an XRI segment even when percent-encoded, yet the authority would like to retain this metadata for purposes of presentation. XRI syntax offers an option for encoding this metadata using a cross-reference beginning with the GCS “$” symbol. As defined in section 2.3.1.3, the top level authority for these identifiers is the “XRI Metadata Specification” [XRIMetadata]. It defines special identifiers for UTF-8 metadata, presentation metadata, and other standard types of identifier metadata together with the rules governing their interpretation.
2.2.5 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. Infrastructure responsible for accepting or presenting XRIs may choose to deal with excluded characters, percent-encoding them on input and/or un-percent-encoding them prior to rendering as described in section 2.2.4. A string that contains these characters in an un-percent-encoded form, however, is not a valid XRI.

Note that presenting "space" or other whitespace characters in un-percent-encoded form may be risky 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, so un-percent-encoded spaces or whitespace characters can make it difficult or impossible to determine where the identifier ends. Finally, un-percent-encoded spaces or whitespace can be used to maliciously construct subtly different identifiers intended to mislead the reader. For these reasons, un-percent-encoded spaces or whitespace characters SHOULD be avoided in presentation.

[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.

2.3 Syntax Components

Generic 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 does not define a new URI scheme. Instead, it follows the example of [IRI] and defines a new identifier scheme, along with a specification for transforming XRIs into generic URIs or IRIs for applications that expect them (see section 2.4.1, "Transforming XRIs into IRIs and URIs").

As with URIs, an XRI may be either absolute or relative.

XRI-reference = XRI / relative-XRI
XRI = [ "xri://" ] xri-hier-part [ "?" xri-query ] [ "#" xri-fragment ]
absolute-XRI = [ "xri://" ] xri-hier-part [ "?" xri-query ]
relative-XRI = xri-path [ "?" xri-query ] [ "#" xri-fragment ]
xri-hier-part = ( xri-authority / iauthority ) [ xri-path-absolute ] / ipath-empty

An absolute XRI consists of the scheme name "xri:" followed by the same set of hierarchical components as an absolute URI – authority, path, and query. A relative XRI consists of either a local path or a relative path.
Finally, in certain contexts such as cross-references (section 2.3.1.4), the “xri:” scheme name is redundant. These contexts can use the xri-value production, which includes all levels of XRI paths.

### 2.3.1 Authority

XRI syntax supports the same types of authorities as generic IRI syntax, called *IRI authorities*. In addition, it supports *XRI authorities* that provide two other mechanisms for specifying the global context of an identifier.

#### 2.3.1.1 XRI Authority

In addition to the authorities supported in generic URI syntax, XRIs support two other mechanisms for specifying the global context of an identifier. The first is the global context symbol (GCS), and the second is the cross-reference (abbreviated in the ABNF as *xref*).

\[
xri-authority = xref-authority / gcs-authority
\]

\[
xref-authority = xref *xri-subseg
\]

\[
gcs-authority = gcs-char xri-segment
\]

#### 2.3.1.2 IRI Authority

In the context of an XRI, a URI authority is distinguished by an initial double slash (“/”).

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

The syntax following this starting delimiter is inherited directly from [IRI]. First, the “iuserinfo” sub-component permits identifying a user in the context of a host. Next, the “ihost” sub-component has three options for identifying the host: a domain name, an IPv4 address, or an IPv6 literal.

Note that the host identifier may be omitted. This is because in generic URI syntax, a default may be defined by the semantics of a particular URI scheme. No default is specified for the XRI scheme; this allows a default to be inherited from the particular protocol used to resolve the XRI.

A host identifier can be followed by an optional port number. Because XRIs are abstract identifiers, the XRI syntax specification does not define a default port. It is expected that the default port will be inherited from the resolution protocol, such as the HTTP/HTTPS protocol specified in [LINK TO RESOLUTION SPEC]. Therefore, if the port is omitted in an XRI, it is undefined.

#### 2.3.1.3 Global Context Symbols (GCS)

To support the abstraction and human-friendly identifier (HFI) requirements, XRIs offer a simple, compact syntax for indicating the logical global context of an identifier: a single prefix character.

\[
gcs-char = $= / @ / + / $! / ":"
\]

The global context symbol characters were selected from the set of symbol characters that are valid in a URI under [URI] to represent the global contexts shown in Table 1 below:

<table>
<thead>
<tr>
<th>Symbol Character</th>
<th>Authority Type</th>
<th>Establishes global context for</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>=</td>
<td></td>
<td></td>
</tr>
<tr>
<td>@</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;:&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>General public: Identifiers for generic concepts for which there is no specific authority, i.e., that are established by public convention. (In the English language, for example, these would be the generic nouns.)</td>
<td></td>
</tr>
<tr>
<td>=</td>
<td>Person: Identifiers that represent an individual person.</td>
<td></td>
</tr>
<tr>
<td>@</td>
<td>Organization: Identifiers that represent an organization of any kind.</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>OASIS XRI Metadata Specification: Special identifiers established by the “XRI Metadata Specification”[XRIMetadata] for interoperable identifier metadata (e.g., language, version, type, query syntax, etc.).</td>
<td></td>
</tr>
<tr>
<td>![folder]</td>
<td>User-relative: Identifiers for which the authority is relative to the current user (“shortcut XRIs”). See section Error! Reference source not found..</td>
<td></td>
</tr>
<tr>
<td>![folder]</td>
<td>XRI author: Identifiers used only for human-readable annotations of XRIs (ignored by machine processing.)</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: XRI global context symbols.

Note that because a global context symbol may precede an xri-segment, and an xri-segment may start with a cross-reference (below), a global context symbol can be used to express the abstract logical context of a conventional URI authority. For example:

```
xri://=(http://www.my-website.com)/favorites.html
```

--expresses that this resource represents an individual

### 2.3.1.4 Cross-References

Cross-references are the primary extensibility mechanism in XRIs. They allow an identifier assigned in one context to be reused in another context, permitting identifiers to be shared across contexts to simplify identifying logically equivalent resources. To syntactically delimit a cross-reference, it is enclosed 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 value 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 XRIs to avoid ambiguity, as discussed in 2.5.5.

A cross-reference may appear at any node of any XRI except within an IRI authority segment. The use of cross-references as the very first segment in an XRI enables any globally-unique identifier in any URI scheme (e.g., an HTTP URI, mailto URI, URN, etc.) to specify a global authority.

```
xri://(mailto:john.doe@example.com)/favorites/home
```

--example of using a URI as an XRI global authority
2.3.1.5 Self-References

Cross-reference syntax is also the means by which an XRI can express that it is not intended for resolution, but only for the purpose of establishing equivalence across contexts. Such an XRI is called a self-reference. To express a self-reference, the entire XRI value is enclosed in parentheses—in essence, it becomes a global cross-reference. This is the XRI equivalent of the English language convention of putting a word or phrase in quotes to express that the author is referring to the word or phrase itself and not to its normal meaning. (In linguistics and philosophy, this is called the “use-mention distinction.”) For example:

The term "user-friendly" is used frequently in computing.
--English-language usage of a quoted term
(xri://+user-friendly)
--XRI syntax for expressing a self-reference

2.3.2 Path

As with URIs, 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. The key difference is that while a URI 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: star segments and bang segments after their leading characters ("*" and "!").

Star 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. Bang 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 star segment, so no leading star is required if this is the first (or only) sub-segment.

An XRI path segment can 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, it will be
interpreted as a 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 2.2.4, “Percent-encoded Characters”.

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

With the exception of star and bang sub-segments, an XRI path segment is considered opaque
by generic XRI syntax. As with IRIs in general, XRI extensions or generating applications may
define special meanings for other IRI reserved characters for the purpose of delimiting extension-
specific or generator-specific sub-components.

### 2.3.3 Query

The XRI query component is identical to the URI query component as described in section 3.4 of
[URI], except that it allows the full XRI character range.

```plaintext
xri-query         = *( xri-pchar / iprivate / "/" / "?"
                     / "@" / "(" / ")" / "!" / """ / "+" / "=" )
```

### 2.3.4 Fragment

XRI syntax also supports fragments as described in section 3.5 of [URI], except that it allows the
full XRI character range.

```plaintext
xri-fragment      = *( xri-pchar / "/" / "?"
                     / "@" / "(" / ")" / "!" / """ / "+" / "=" )
```

Since XRI syntax can directly address attributes or secondary representations of a primary
resource 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 2.3.1.4

### 2.4 Transformations

#### 2.4.1 Transforming XRIs into IRIs and URIs

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

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

Applications MUST transform XRIs to IRIs using the following steps (or an equivalent process
that achieves exactly the same result). Before applying these steps, the XRI must be in percent-
encoded normal form as defined in section 2.2.4.
1. If the XRI is not encoded in UTF-8, convert the XRI to a sequence of characters encoded in UTF-8, normalized according to Normalization Form C (NFC) as defined in [UTR15].

2. Optionally add XRI metadata using cross-references as defined in section 2.2.4.1. Note that the addition of XRI metadata may change the resulting IRI or URI for the purposes of comparison. The significance or insignificance of specific types of XRI metadata is defined in the “XRI Metadata Specification” [XRIMetadata].

3. Apply the XRI escaping rules defined in section 2.4.2. Note that this step is not idempotent (i.e., each time this step is applied, it may yield different results), 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 this step, the percent-encoded XRI may be used as an IRI. An XRI 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 2.

In general, an application SHOULD use the least percent-encoded version appropriate for the context in which the identifier appears. For example, if the context allows an XRI directly, the identifier SHOULD be an XRI in percent-encoded normal form as described in section 2.2.4. If the context allows an IRI but not an XRI, the identifier SHOULD be in IRI-normal form, and so on.

2.4.2 Escaping Rules for XRI Syntax

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

The first rule deals with cross-references as explained in section 2.3.1.4. Since a cross-reference contains either an IRI or an XRI reference (which itself may contain further nested IRIs or XRIs), it may include characters that, if not escaped, would cause misinterpretation when the XRI is used in a context that expects IRIs or URIs. 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?id=1)
query = <undefined>
```

To avoid this type of misinterpretation, certain characters in a cross-reference must be percent-encoded when transforming an XRI 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:
In addition, the slash "/" character in a cross-reference may also be misinterpreted by a non-XRI aware processor. 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 in percent-encoded 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 percent-encoded 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 literally and not as a delimiter. To preserve this type of distinction when converting an XRI to an IRI, 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 encoding rules MUST be applied during Step 3 of section 2.4.1. Before applying these rules, the XRI MUST be in percent-encoded 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 across the entire XRI.
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”.

xri-syntax-resolution-2.0-wd-02  19 January 2005
Copyright © OASIS Open 2005. All Rights Reserved.
Page 22 of 37
2.4.3 Transforming IRIs Back into XIRIs

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

1. If the XRI is not encoded in UTF-8, convert the XRI 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., each time this process is applied, it may yield different results), so it is very important that implementers apply this process appropriately to avoid changing the semantics of the identifier.

2.5 Relative XRI References

2.5.1 Reference Resolution

For XIRIs 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, in URI-normal form in contexts that expect URIs and in IRI-normal form in contexts that expect IRIs, since the algorithms described in [IRI] and [URI] may produce incorrect results when applied to XRI references in percent-encoded normal form, particularly when those XRI references contain cross-references.

In contexts that allow a native XRI (i.e. an XRI in percent-encoded 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 XIRIs in percent-encoded normal form provided that the processor a) treats the characters additionally allowed in IRI references the same way as unreserved characters in URI references, as required by section 5 of [IRI] and b) treats all characters within all cross-references as unreserved characters in URI references (i.e. treats cross-references as opaque with respect to relative reference resolution).

2.5.2 Reference Resolution Examples

The following are examples of resolving relative XRI references. These examples are very similar to the examples for resolving relative references in [URI]. Staring with a base XRI of:

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

a relative reference is transformed to its target URI as follows.

2.5.3 Normal Examples

```
"g:h"  =  "g:h"
!/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/"
```
2.5.4 Abnormal Examples

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

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

XRI processors, like URI and IRI processors, must be prepared for unnecessary or nonsensical forms of "." and "..".

XRI processors, like URI and IRI processors, 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.
### 2.5.5 Leading Segments Containing a Colon

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

- 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 forward slash (“/”) character in a relative XRI reference contains a colon, the relative XRI reference must be rewritten to begin either with a “*” or a “../”. Thus, “a:b” becomes either “*a:b” or “../a:b”.

### 2.6 Normalization and Comparison

In general, the normalization and comparison rules for generic 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 XRIs 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).

#### 2.6.1 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 2.3.1) is case-insensitive as defined in [IRI].
- As specified in section 2.2.4, comparison of percent-encoded characters is case-insensitive for the hexadecimal digits “A” through “F”.

#### 2.6.2 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 percent-encoded unreserved characters SHOULD be considered equivalent. If one XRI percent-encodes one or more unreserved characters, and another XRI is different 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 2.4.1 SHOULD be considered equivalent, assuming the same XRI metadata is inserted as described in section 2.4.1.

#### 2.6.3 Optional Syntax

- An xri-segment (section 2.3.2) that omits the optional leading star (“*”) is equivalent to the same xri-segment prefixed with the leading star. For example the segment “/foo*bar” is equivalent to the segment “/foo*bar”.

---

xri-syntax-resolution-2.0-wd-02  19 January 2005
Copyright © OASIS Open 2005. All Rights Reserved.
• A cross-reference (section 2.3.1.4) that begins with the GCS symbol for annotations ("!") AND
  the delimiter that precedes the cross-reference SHOULD be ignored entirely for purposes of
  comparison. For example, "xri:@:A6B4.(www.example.org):5E32" is equivalent to
  "xri:@:A6B4:5E32". Note that because XRI annotations are explicitly designed to be ignored
  by XRI processors, failure to observe this rule will cause XRIs that are intended to be
  equivalent to be incorrectly evaluated.

2.6.4 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 “XRI Metadata
  Specification” [XRIMetadata]. See section 2.2.4.1.

2.6.5 Canonicalization

In general, XRIs do not have a single canonical form. This is particularly true for XRIs 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 may define its own
rules with respect to case-sensitivity, optional or implicit syntax, etc., making canonicalization of
those segments outside the scope of this specification.

It is nevertheless useful to define guidelines for making XRIs reasonably canonical. XRIs 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, XRIs should be provided in a form in which:

- The optional xri scheme is added,
- The scheme is provided in lowercase,
- The authority component is provided in lowercase,
- Percent-encoding uses uppercase A through F,
- If optional, the leading star in xri-segments is omitted,
- Unnecessary percent-encoding is removed,
- ../ 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.

<table>
<thead>
<tr>
<th>Avoid</th>
<th>Recommended</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>@example</td>
<td>xri://@example</td>
<td>Add optional scheme</td>
</tr>
<tr>
<td>XRI://@example</td>
<td>xri://@example</td>
<td>Lowercase scheme</td>
</tr>
<tr>
<td>xri://@Example</td>
<td>xri://@example</td>
<td>Lowercase authority</td>
</tr>
</tbody>
</table>
Table 2: Examples of XRI canonicalization recommendations.

<table>
<thead>
<tr>
<th>XRI</th>
<th>XRI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xri://@example%2f</td>
<td>xri://@example%2F</td>
<td>Uppercase percent-encoding</td>
</tr>
<tr>
<td>xri://@example/*abc</td>
<td>xri://@example/abc</td>
<td>Remove optional leading star</td>
</tr>
<tr>
<td>xri://@ex%61mple</td>
<td>xri://@example</td>
<td>Remove unnecessary percent-encoding</td>
</tr>
<tr>
<td>xri://@example/./abc</td>
<td>xri://@example/abc</td>
<td>Avoid ./ and ../ in absolute XRIs</td>
</tr>
</tbody>
</table>
3 Security and Data Protection

3.1 Secure Resolution
The resolution protocol described in section 3 is not intrinsically trustworthy. It is expected that, in practice, some combination of DNSSEC, SSL, TLS, and other existing technologies will be employed to increase the security of the resolution process.
While such enhancements are outside the scope of this specification, an XRI Secure Resolution Specification is a future deliverable of the OASIS XRI TC. Additional follow-on work is also expected to define best practices and facilitate interoperability.

3.2 XRI Metadata
The use of cross-references employing the GCS “$” symbol for encoding XRI metadata in an XRI (section Error! Reference source not found.) may involve other security and data protection considerations that are outside the scope of this specification. These considerations are addressed in the “XRI Metadata Specification” [XRIMetadata].

3.3 XRI Usage in Legacy Infrastructure
Where XRIs are used within the Internet and other computing infrastructure, the security and data protection considerations are similar to those of other URI schemes. In this context the material in [URI], section 7, Security Considerations, is particularly informative. It includes a discussion of the following topics:
- Reliability and Consistency
- Malicious Construction
- 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. Such applications must be developed with independent security reviews for the specific scenarios in which XRIs are used.

3.4 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 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. These capabilities may enable new security and data protection features at the fundamental level of resource identifiers.
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 security and data protection considerations relating to XRIs.
4 References

4.1 Normative


4.2 Informative


Appendix A. Collected ABNF for XRI (Normative)

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

```
XRI               = [ "xri://" ] xri-hier-part [ "?" xri-query ]
                    [ "#" xri-fragment ]

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

XRI-reference     = XRI
                    / relative-XRI

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

relative-XRI      = xri-path [ "?" xri-query ] [ "#" xri-fragment ]

xri-authority     = xref-authority
                    / gcs-authority

xref-authority    = xref *xri-subseg

gcs-authority     = gcs-char xri-segment

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-nz *( "/" 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

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

gcs-char          = ":" / ":" / ":;++" / ":" / ":!" / ":" / ":$" / ":!"

xri-pchar         = xri-unreserved / pct-encoded / xri-sub-delims / ":;"

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

xri-query         = *( xri-pchar / iprivate / "/" / "?"
                    / ":" / ":" / "!" / "!" / "!" / "!" / "!" / "!" / "!" / "!" / "!" )

xri-fragment      = *( xri-pchar / "/" / "?"
                    / ":" / ":" / "!" / "!" / "!" / "!" / "!" / "!" / "!" / "!" / "!" / "!"
```
xri-reserved = xri-gen-delims / xri-sub-delims

xri-gen-delims = "：" / "／" / "？" / "＃" / "【" / "】" / "（" / "）" / "＊" / gcs-char

xri-sub-delims = "⑤" / "；" / "、" / "’"

xri-unreserved = unreserved / ucschar

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
/ 6:5( h16 ":" ) ls32
/ [ h16 ] 6:4( h16 ":" ) ls32
/ [ *1( h16 ":" ) h16 ] 6:3( h16 ":" ) ls32
/ [ *2( h16 ":" ) h16 ] 6:2( h16 ":" ) ls32
/ [ *3( h16 ":" ) h16 ] 6:1( h16 ":" ) ls32
/ [ *4( h16 ":" ) h16 ] 6:0( h16 ":" ) ls32
/ [ *5( h16 ":" ) h16 ] 6:0( h16 ":" ) h16
/ [ *6( h16 ":" ) h16 ] 6:0( 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 / %x100000-10FFFF
ifragment = *(( ipchar / "/" / "?" )
ipchar = iunreserved / pct-encoded / sub-delims / ":" / "@"
iunreserved = ALPHA / DIGIT / "-" / "." / "_" / "_" / ucschar
pct-encoded = "%" HEXDIG HEXDIG
ucschar = %xA0-D7FF / %xF900-FDCF / %x10000-D7FF
/ %x10000-1FFFF / %x20000-2FFFF / %x30000-3FFFF
/ %x40000-4FFFF / %x50000-5FFFF / %x60000-6FFFF
/ %x70000-7FFFF / %x80000-8FFFF / %x90000-9FFFF
/ %xA0000-AFFFF / %xB0000-BFFFF / %xC0000-CFFFF
/ %xD0000-DFFFF / %xE1000-EFFFF
reserved = gen-delims / sub-delims
gen-delims = ":" / "/" / "?" / "#" / "[" / "]" / "@"
sub-delims = "!" / "$" / "&" / "'" / "(" / ")"
/ "$" / "+" / "," / ";" / "="
unreserved = ALPHA / DIGIT / "-" / "." / "_" / "_"
Appendix B. Transforming HTTP URIs to XRIs
(Non-Normative)

To leverage existing infrastructure, it may sometimes be useful to convert HTTP URIs into XRIs.
Because XRI syntax is, for the most part, a superset of generic URI syntax, the majority of HTTP
URIs can be converted to valid XRIs simply by replacing the scheme “http” with “xri”. Special
consideration, however, must be given to HTTP URIs employing the characters in the “xri-
reserved” production of this specification that differ from those in the “reserved” production of
[URI] (as amended by [RFC2732]). These include opening parenthesis (“(“), closing parenthesis
(“)”), dot (“.”), asterisk (“*”), and exclamation point (“!”).
Typically, characters in the “reserved” production of [URI] that appear in an HTTP URI as normal
characters (i.e. not as syntactic delimiters) are percent-encoded. However, this is not required in
all cases. [URI] says

“Characters in the ‘reserved’ set are not reserved in all contexts. The set of characters
actually reserved within any given URI component is defined by that component. In general, a
color character is reserved if the semantics of the URI changes if the character is replaced with its
escaped US-ASCII encoding.”

Characters in the “xri-reserved” set that are properly left un-escaped in an HTTP URI may be
semantically significant when the HTTP URI is converted to an XRI. For example,

http://www.example.com/example1:example2

is a valid HTTP URI even though it contains an unescaped reserved character – a colon (”:“) –
because section 3.3 of [URI] explicitly omits this character from the reserved set for “path”
components. The same unescaped character in an XRI, however, will be interpreted as a
delimiter. If the colon character should not be understood as a delimiter in the resulting XRI, it
must be escaped during conversion. The same applies to the other characters mentioned above.

Generally, any character not in the “xri-pchar” set that appears in the “abs_path”, “query”, or
“fragment” components of the HTTP URI will need to be escaped when converting to an XRI. This
avoids misinterpretation in the resulting XRI following the guidance in section 2.2.4 of this
specification.
Exceptions are possible. For example, if the author of the above HTTP URI intended the colon
character to be interpreted as described in this specification, or if its use would not be
misinterpreted, then it may be left in its unescaped form.
In addition, it may be beneficial to escape other characters like the percent (“%”) character,
particularly if it may be necessary to convert the resulting XRI back to an HTTP URI. Whether
such additional escaping is desirable or not depends on the intended use of the resulting XRI, the
context in which it will appear, how it is intended to be resolved, etc.
It is worth noting that some rare forms of HTTP URIs can result in XRIs that are misleading to the
reader. For example, the following unusual HTTP URI is valid per [URI].

http://@example.com/example1

When converted to an XRI, as

xri://@example.com/example1

a casual reader could easily misinterpret the “uri-authority” component as an “xri-authority”.
Similarly, a URI with an authority segment like

http://=bob@example.com/example1

could be similarly misinterpreted.
Appendix C. Acknowledgments

The editors would also like to thank the following people who participated in the XRI TC and/or provided input and review of this specification (affiliations listed for OASIS members):

Thomas Bikeev (EAN International), Winston Bumpus (formerly of Novell), James Bryce Clark (OASIS), Matthey Dovey (Individual), Lars Marius Garshol, Steve Green (Epok), Lance Hood (Epok), Phillip LeBlanc (GemPlus), Marc LeMaitre (OneName), Rajeev Maria (Visa International), Adarbad Master (Epok), John McGarvey (IBM), Davis McPherson (Epok), Mike Mealling (Verisign), Reva Modi (Infosys), Joseph Moeller (EDS), Brian Nimmo (Epok), Mary Nishikawa (Individual), Eamonn Neylon (Individual), Masaki Nishitani (NRI), Norman Paskin, Krishnan Rajagopalan (Novell), Chetal Sabinis (Epok), Jim Schreckengast (formerly of Gemplus), Tomonori Seki (NRI), Xavier Serret (Gemplus), Terence Spielman (Visa International), Marc Stephenson (TSO), Geoffrey Strongin (AMD), Bernard Vatant, John Veizades (Visa International), Bill Washburn (XNSORG), Tetsu Watanabe (NRI), Dave Wentker (Visa International), Loren West (Epok), and Michael Willett (Wave Systems).

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 2396 (and RFC 2396bis)
- RFC 2616
- RFC 2718
- RFC 3401-3405 (DDDS)
- REST Architecture
- IRI – Internationalized Resource Identifiers draft
- XNS
## Appendix D. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0-wd-01</td>
<td>2005-1-18</td>
<td>Dave McAlpin</td>
<td>Initial editor's draft</td>
</tr>
<tr>
<td>2.0-wd-02</td>
<td>2005-1-19</td>
<td>Dave McAlpin</td>
<td>Editorial pass</td>
</tr>
</tbody>
</table>
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 Executive Director.

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 Executive Director.

Copyright © OASIS Open 2005. All Rights Reserved.

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 paragraph are included on all such copies and derivative works.

However, this document itself does not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document 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 RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.