An Introduction to XRIs

Working Draft 04, 14 March 2005

Document identifier :

xri-intro-V2.0-wd-04

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:

Fen Labalme, PlaNetwork <fen@idcommons.org>
Mike Lindelsee, Visa International <mlindels@visa.com>
Gabe Wachob, Visa International <gwachob@visa.com>

Abstract:

This document is a non-normative introduction to the uses and features of XRIs. It is intended to accompany the XRI 2.0 suite of specifications including *Extensible Resource Identifier (XRI) Syntax 2.0* **[XRISyntax]**, *Extensible Resource Identifier (XRI) Resolution 2.0* **[XRIResolution]**, and *Extensible Resource Identifier (XRI) Metadata 2.0* **[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

1. Introduction 3
   1.1 What are XRIs? 3
   1.2 Why a Uniform Abstract Identification Layer? 4
2. The Types of Problems XRIs are Designed to Solve 5
   2.1 The Broken Links Problem (Persistent Identification) 5
   2.2 The Multiple Authority Problem (Federated Identification) 5
   2.3 The N-Squared Mapping Problem (Shared Identification) 6
   2.4 The Exploding Addresses Problem (Simplified Identification) 6
   2.5 The Bootstrap Discovery Problem (Metadata Identification) 6
   2.6 The Public Identifier Problem (Privacy-Protected Identification) 7
   2.7 The Future-Proofing Problem (Extensible Identification) 7
3. How XRIs Help Solve These Problems: Example Usage Scenarios 8
   3.1 Persistent Identification 8
   3.2 Federated Identification 10
   3.3 Shared Identification 12
   3.4 Simplified Identification 14
   3.5 Metadata Identification 15
   3.6 Privacy-Protected Identification 16
   3.7 Extensible Identification 17
4. A Brief Guide to the Normative XRI V2.0 Specifications 19
   4.1 XRI Syntax 19
   4.2 XRI Resolution 19
   4.3 XRI Metadata 19
5. References 21
Appendix A. Example XRIs 22
Appendix B. Acknowledgments 23
Appendix C. Revision History 24
Appendix D. Notices 25

1. Introduction

1.1 What are XRIs?

XRIs provide a standard syntax and resolution protocol for abstract identifiers—identifiers that are independent of a specific location, domain, application, or protocol. The XRI specifications are built directly on top of the foundation provided by the URI (Uniform Resource Identifier) and IRI (Internationalized Resource Identifier) specifications from IETF and W3C, as shown in Figure 1.

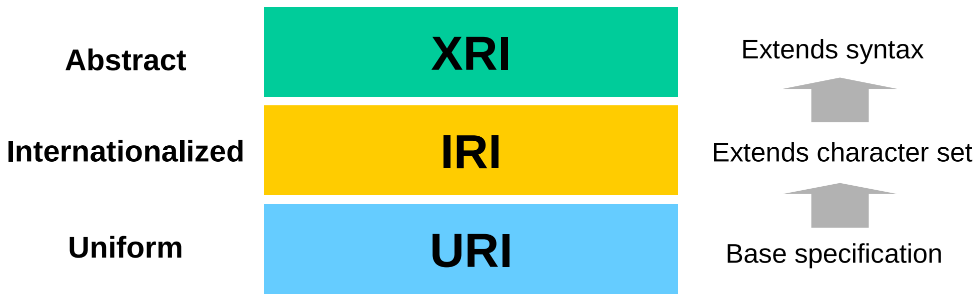

Figure 1: The relationship of XRIs, IRIs, and URIs

URIs introduced a standard means of identifying resources across distributed networks that in only 15 years has become the most successful identifier scheme in history. IRIs subsequently extended the generic URI scheme, which supports only the ASCII character set, to include the full UCS (Unicode Character Set).

XRIs take a third step by adding additional syntax and resolution features that enable XRIs to solve problems of abstract identification that are not easily addressed by conventional URI or IRI syntax or resolution. One of these problems—persistence—has been addressed by other URI schemes for abstract identifiers, such as the URN (Uniform Resource Name) scheme **[RFC2141]**. While XRIs fulfill the requirements for URNs as specified in **[RFC1737]**, they go on to address a much wider range of issues in abstract identification. As shown in more detail in section 2:

- *XRIs provide a uniform syntax and resolution protocol for both persistent and reassignable abstract identifiers.*
- *XRIs provide a uniform syntax for delegating abstract identifiers between authorities at any level or context.*
- *XRIs provide a uniform syntax (called “cross-references”) for sharing abstract identifiers across all contexts.*
- *XRIs provide both a generic and a trusted protocol for resolving a single abstract identifier into any number of concrete identifiers.*
- *XRIs provide a simple, standard means of discovering URIs that may be associated with a resource, including those needed for additional metadata discovery.*
- *XRIs provide a standard means of protecting privacy by revealing the least possible information in an identifier and allowing an authority to control further access.*
- *XRIs provide a standard means of extension without sacrificing interoperability.*

1.2 Why a Uniform Abstract Identification Layer?

Not all resources will need the additional layer of indirection that XRIs can provide, just as not all resources require URIs or IRIs today. When this additional layer is needed, however, the purpose of XRIs is to make it as interoperable across all sites and systems as URIs and IRIs are today. In essence, the goal of XRI architecture is to enable the same *uniform identification layer* for abstract identifiers that URIs and IRIs provide for concrete identifiers today.

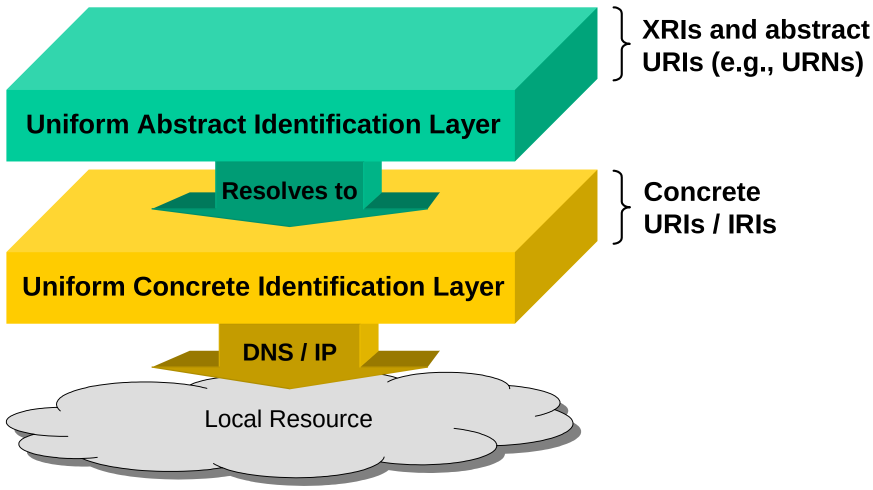

Figure 2: The goal of XRI is to provide a uniform abstract identification layer.

The key difference between the concrete identification layer and the abstract identification layer is that the latter enables identification of resources independent of a specific context, i.e.:

- *Independent of a specific network location* where a resource may exist at a particular point in time.
- *Independent of a specific directory, database, or repository* that might store a resource.
- *Independent of a specific application* that may be responsible for creating, maintaining, or processing a resource.
- *Independent of a specific domain, authority, or owner* of a resource at a particular point in time.
- *Independent of a specific transport or communications protocol* that may be used to interact with a resource.
- *Independent of a specific semantic label or descriptor* that may be associated with a resource at a particular point in time.

While it has long been recognized that persistent identification is one of the key problems an abstract identification layer can solve, the recent growth of distributed XML architecture, Web services, and digital identity infrastructure has brought to light a number of other key problems that a uniform abstract identification layer can help address **[XRIReqs]**.

The purpose of this document is to first identify these other types of problems (section 2), then show how XRIs can be used to solve them through a series of typical usage scenarios (section 3). Finally it provides a brief introduction to the normative XRI 2.0 specifications in section 4.

2. The Types of Problems XRIs are Designed to Solve

2.1 The Broken Links Problem (Persistent Identification)

Probably the best known use case for abstract identifiers is the need to maintain a persistent reference to a resource when it moves or changes locations on the network. As explained in the U.S. Government report *Persistent Identification: A Key Component of an E-Government Infrastructure* **[CENDI]**:

> *The current addressing structure for the Web is based on the Uniform Resource Locator (URL). This technology uses a physical location (IP address/server/path/file name) to identify and locate digital objects. While the URL provides direct, efficient access, URL-only naming fails whenever the resources are moved or reorganized. In addition, the URL may stay the same, but the object addressed by the URL may change significantly or be replaced with completely different content. Thus, while the URL permits interoperability in assigning an initial address for an object, it offers no assurance that this address will follow the object as it moves among locations. The lack of persistence or "linkrot" leads to 404 errors (file not found), inhibiting access to digital objects and causing problems when archiving material for long-term preservation and permanent access.*

This is the fundamental rationale for URNs (Uniform Resource Names), the URI scheme for persistent, location-independent resource identifiers **[RFC2141]**. From this standpoint, the most significant difference between URN and XRI architecture is that URNs deal with only with persistent identifiers, whereas XRIs deal with all types of abstract identifiers—both persistent and reassignable.

> *XRIs provide a uniform syntax and resolution protocol for both persistent and reassignable abstract identifiers.*

2.2 The Multiple Authority Problem (Federated Identification)

URIs, particularly HTTP URIs (commonly called URLs), achieved their success largely because they solved the problem of how to locate and retrieve a resource anywhere across the global Internet—no matter what domain or authority hosted it. They did this by building on top of two federated addressing systems: DNS naming and IP addressing, both of which allow multiple independent authorities to cooperate in resolving different segments of an identifier.

However with DNS and IP, the "edge" of the federation network is an IP-addressable resource (a server, a device, a network application). There is no standard syntax for continuing delegation between resources beyond this endpoint. So, for example, if a corporate directory system wanted to delegate authority for home contact data to employees, or a government agency wanted to delegate responsibility for portions of a spreadsheet it to different departments (who in turn wanted to delegate parts of it to different employees), XRIs provide the syntax necessary to express this in a uniform manner.

> *XRIs provide a uniform syntax for delegating abstract identifiers between authorities at any level or context.*

2.3 The N-Squared Mapping Problem (Shared Identification)

One of the most pervasive problems in large-scale systems integration is the cross-context mapping of resources and data. As long as a resource is only identified in a local context, then each additional system with which it must be shared requires an additional mapping between the two contexts. This pairwise mapping problem grows exponentially with the number of resources and the number of systems involved.

While security and privacy requirements may require pairwise mappings to be used in certain situations, in most other cases establishing a shared abstract identifier collapses the n^2 mapping problem into a much easier n-1 mapping problem. This is particularly true of resources whose primary purpose is to establish shared semantics across systems, e.g., data interchange schemas, dictionaries, taxonomies, ontologies, etc.

> *XRIs provide a uniform syntax (called "cross-references") for sharing abstract identifiers across all contexts.*

2.4 The Exploding Addresses Problem (Simplified Identification)

The constant progression of Internet, wireless, and other communications technologies has begun to produce an explosion of addresses. This problem applies both to people—the exploding business card problem—and increasingly to other resources (smart phones, networked applications) that can also be addressable via multiple networks and interfaces.

Jamie Lewis, CEO of the Burton Group, summarized this problem succinctly in a recent press article **[Lewis]**:

> *Today, we use a wide variety of different mechanisms for identification, including e-mail addresses, IP addresses, phone numbers, and object identifiers. But most of these are specific to one means of interaction. None of them is persistent across the many different ways that people, applications, and devices can communicate, and so they don't function well as identifiers in the long run.*

A uniform abstract identification layer can solve this problem both for the addressees (who will not longer need to constantly change their published addresses/interfaces), as well as for the addressors, who will no longer need to constantly update their local references.

> *XRI provide both a standard and a trusted protocol for resolving a single abstract identifier into any number of concrete identifiers.*

2.5 The Bootstrap Discovery Problem (Metadata Identification)

Many URI schemes are specific to a particular means of interaction with a resource (http, https, ftp, mailto, telnet, etc.) Binding the identifier for a resource to the protocol used to access that resource makes it easy to interact with a resource via that particular protocol, but it does not facilitate discovery of other means of interaction that might be available. Nor do most of these protocols standardize a means for obtaining additional metadata about a resource, e.g., its version, media type, encoding, author, rights management, etc.

In fact, with the rapidly growing number of ways to describe a resource, its attributes, and relationships, there may in fact be multiple sources and formats of this metadata, each with their own interaction-specific URIs identifying them. But how does one discover these? This is the "bootstrap" discovery problem, and one perfectly suited to a uniform abstract identification layer.

> *XRIs provide a simple, standard means of discovering URIs that may be associated with a resource, including those needed for additional metadata discovery.*

2.6 The Public Identifier Problem (Privacy-Protected Identification)

Another ramification of identifiers that are specific to a particular means of interaction is the privacy issues they create, particularly those that need to be used publicly. For example, the most convenient way for users to register at many websites today is using an email address. However many Internet users are reluctant to share their email address for fear of spam.

As public addresses, abstract identifiers have the virtue of not needing to reveal a specific means of interaction, yet serving being able to serve as a key by which access requests can be made. By applying policy to these access requests (for example, requiring authentication), XRI authorities can increase their privacy and obtain greater control over the sharing of contact and other sensitive information.

> *XRIs provide a standard means of protecting privacy by revealing the least possible information in an identifier and allowing an authority to control further access.*

2.7 The Future-Proofing Problem (Extensible Identification)

Many pre-Web identifier schemes have been adapted to the Web through the development of special URI schemes (a good example is the “mailto:” scheme for Internet email addresses). Countless other resources in legacy systems have also been made URI-addressable through authority-specific mapping algorithms.

However the introduction of new URI schemes, resolution protocols, and mapping algorithms can require expensive changes to existing infrastructure and can increase, rather than decrease, integration complexity. XRIs help ameliorate this problem both by providing an easy way to “bind” new URI schemes to existing infrastructure. They also provide the alternative simply defining new XRI namespaces that can be cross-referenced by all other XRI namespaces. Either way, providing this “extensibility by design” for identifiers was one of the key motivations for XRI just as it was for XML.

> *XRIs provide a standard means of extension without sacrificing interoperability.*

3. How XRIs Help Solve These Problems: Example Usage Scenarios

This section will use a set of scenarios to show how XRI architecture can solve the types of problems described in section 2. For continuity, the overall context will be a set of issues a government might face in sharing resources via the Internet. XRIs can be used to identify any type of resource – a person, a network device, a Web page, a web service, an XML namespace, a particular instance of a physical book, etc. This set of examples focuses primarily on documents available via the Web, but the principles apply to any type of XRI-identified resource.

3.1 Persistent Identification

> *XRIs provide a uniform syntax and resolution protocol for both persistent and reassignable abstract identifiers.*

As highlighted in the previous **[CENDI]** quote, one of the most frustrating aspects of the Internet today is the lack of persistent identification for Web-accessible resources. In most cases the root cause is encoding a resource's location and the access protocol in its URI or IRI. Such an identifier is valid only as long as the resource doesn't change locations or retrieval mechanisms. In practical terms, this means when a web site is reorganized, links break.

As an example, say a department of a government agency publishes a document named `govdoc.pdf` via the following URI:

```
ftp://department.agency.example.org/docs/govdoc.pdf
```

The document is easily located via DNS and retrieved via FTP. However, when the resource's access protocol changes from FTP to HTTP, the resource's identifier changes to:

```
http://department.agency.example.org/docs/govdoc.pdf
```

Typically, external links to the former address would now be broken. Notice that the resource itself didn't change – `govdoc.pdf` is still exactly the same – but its identifier is different because the protocol used to retrieve the resource changed.

Next, imagine the department undergoes a reorganization. Its current domain name, `department.agency.example.org`, is changed to `newdept.agency.example.org`. Now the resource's identifier changes to:

```
http://newdept.agency.example.org/docs/govdoc.pdf
```

Finally, the web site hosting `govdoc.pdf` is reorganized, and `docs` is renamed `documents`. The document's identifier changes yet again, this time to:

```
http://newdept.agency.example.org/documents/govdoc.pdf
```

In each case, the resource stayed the same, but its identifier changed. Because the identifier was intrinsically associated with the resource's location and retrieval mechanism, external links to `govdoc.pdf` are broken with every change.

The first step in persistently identifying a resource, then, is assigning an identifier that doesn't depend on the resource's network location and access mechanism, since both will inevitably change over time. This type of identifier is called "abstract" because it doesn't resolve directly to the resource via a specific protocol, but instead resolves into other concrete identifier(s) and access protocol(s) (or metadata that contains this information.) XRIs fulfill this requirement because they don't encode concrete location or protocol information in the identifier.

If the government agency department in the previous example used XRIs, in the first step it might have assigned the resource `govdoc.pdf` an XRI such as:

```
xri://@example.org*agency*department/docs/govdoc.pdf
```

Note that stars ("*") are used instead of dots (".") as delegation characters in XRI syntax, so the dot that appears in `@example.org` is simply treated as another data character. Also note the leading character "@" in the XRI authority segment—this is called a *global context symbol*. It is used to indicate the abstract global context of an authority—in this case, an organization.

XRI resolution would then resolve this abstract identifier into the concrete URI `ftp://department.agency.example.org/docs/govdoc.pdf`. When the access protocol changed from FTP to HTTP, the XRI would not break. Only the resolution result would change—it would return `http://department.agency.example.org/docs/govdoc.pdf` instead of `ftp://department.agency.example.org/docs/govdoc.pdf`.

However we still have a problem when the name of the government department is changed from `department` to `newdept`. In this respect the XRI above has the same problem as the original domain name-based URI—it was constructed from semantic names that are subject to change. We *could* try to solve this by creating a new XRI that reflects the department's new name, i.e.:

```
xri://@example.org*agency*newdept/docs/govdoc.pdf
```

We could then use XRI resolution to map this name to the previous XRI—`xri://@example.org*agency*department/docs/govdoc.pdf`. This XRI would finally resolve to the concrete URI `http://department.agency.example.org/docs/govdoc.pdf`. However such an approach would eventually lead to a spaghetti code of new-to-old XRI mappings. It also has the drawback of preventing reassignment of the identifier "`department`" for another purpose.

A much better solution would be to assign the resource "`govdoc.pdf`" an identifier that never needs to change or be reassigned. This can be accomplished using a fully persistent XRI such as the following:

```
xri://@!9990!A58F!1C3D/!2495
```

Notice the bang ("!") symbols that appear throughout this XRI. Bangs are the delimiter used in XRI syntax to indicate that the identifier that follows (called a *sub-segment*) is intended to be persistent (i.e., it fills the role of a URN as defined in **[RFC1737]**).[1] Compare this to the star ("*") delimiter in the previous example, which indicates that the sub-segment that follows is intended to be *reassignable* (i.e., it fills the same role as a domain name or filename).

Once "`govdoc.pdf`" has been assigned this persistent XRI, the department now has two options for referencing this resource abstractly as shown in Figure 3.

[1] As with URNs, the issue of whether a persistent sub-segment is in fact permanent (never reassigned) is a matter of operational policy for the assigning authority. XRIs can't help with the operational issue, but XRI syntax allows the authority to express its intent.

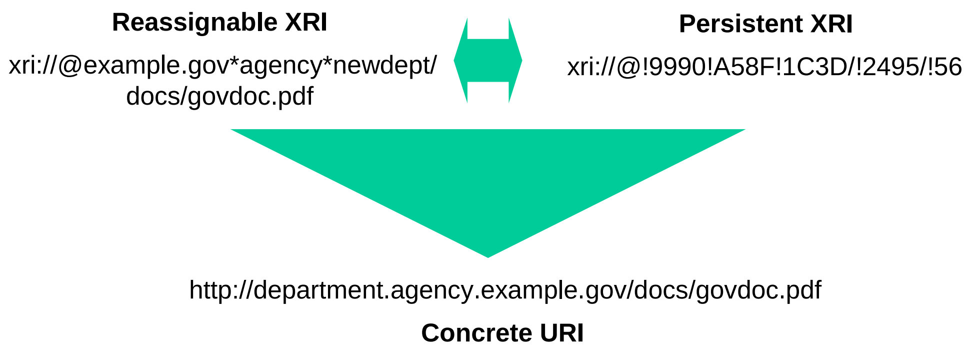

Figure 3: The “resolution triangle” of reassignable XRIs, persistent XRIs, and concrete URIs.

The two XRIs at the top of the triangle are called *XRI synonyms* because they both resolve to the same resource. An XRI resolution request for either XRI (see section 3.4) can return the other XRI synonyms for referencing the resource, as well as the concrete URIs for accessing the resource, all in one step. So an XRI-consuming application can use human-friendly reassignable XRIs to transparently discover and “memorize” the persistent XRI synonyms for a resource without any extra overhead, leading to steadily increasing efficiency and resilience in distributed systems rather than today’s slowing decaying “link rot”.

XRI architecture recognizes that not all identifier authorities may be capable of providing the infrastructure or management necessary to assign persistent identifiers. It is possible for the government agency department to have a persistent XRI, for example, but for its internal document repository not to have this capability. XRI syntax supports this use case by allowing XRIs to contain a mixture of persistent and reassignable sub-segments, as shown below:[2]

```
xri://@!9990!A58F!1C3D/docs/govdoc.pdf
```

This flexibility allows XRIs to be optimized for the context in which they appear: descriptive and memorable for the side of a bus, for example, or dense and opaque for persistence. XRI resolution then allows identifier authorities to map reassignable XRIs to persistent XRIs or to concrete URIs as needed to support both human readability and persistent linking.

3.2 Federated Identification

XRIs provide a uniform syntax for delegating identifiers between authorities at any level or context.

As discussed in section 2.2, federated naming and addressing systems are fundamental to Internet and Web architecture. By allowing multiple independent authorities to cooperate in naming and resolving different segments of an identifier, they provide both scalability and distributed management.

The best known example of a federated naming system is DNS. Let’s look again at the original URI for `govdoc.pdf`:

```
ftp://department.agency.example.org/docs/govdoc.pdf
```

[2] To our knowledge, this feature is unique to XRIs. URNs **[RFC2141]** and other location-independent identifier systems support only persistent identifiers.

The first portion of this URI between the "//" and the first "/" is called the authority segment. This URI uses the DNS name `department.agency.example.org` to specify the authority. DNS resolves each dot-delimited sub-segment right-to-left, i.e., the `gov` nameserver is queried for `example`, the `example` nameserver is queried for `agency`, and the `agency` nameserver is queried for `department`.

XRIs work the same way, except federation works from left-to-right (the same direction as the rest of the path). For example, in our original XRI:

```
xri://@example.org*agency*department/docs/govdoc.pdf
```

The authority segment is `@example.org*agency*department`. To resolve this using XRI resolution, the `@` authority is queried for `*example.org`, the `*example.org` authority is queried for `*agency`, and the `*agency` authority is queried for `*department`.[3]

Notice that each of the sub-segments includes the preceding star ("*").[4] This allows an XRI authority to distinguish between reassignable and persistent sub-segments. For example, say the government agency department also assigned `govdoc.pdf` the following persistent XRI:

```
xri://@!9990!A58F!1C3D/!2495
```

The authority segment here is `@!9990!A58F!1C3D`. To resolve this using XRI resolution, the `@` authority is queried for `!9990`, the `!9990` authority is queried for `!A58F`, and the `!A58F` authority is queried for `!1C3D`.

However from the standpoint of federation the most significant difference between DNS and XRI infrastructure is not just that XRIs support both reassignable and persistent identifiers or that delegation is processed from left to right, but that XRIs provide consistent federation syntax throughout the identifier, not just in the XRI Authority portion. Consider the following XRI:

```
xri://@example.org*agency*department/offices*east*library/govdoc.pdf
```

In this case, delegation continues in the path portion of the XRI. Once we have reached the public XRI authority `@example.org*agency*department`, it can switch to internal delegation (using local XRI resolution or another resolution protocol). The `/offices` authority can delegate to a particular office, `*east`, which in turn can delegate to its internal library, `*library`. Each of these authorities can also maintain corresponding persistent XRIs to support persistent linking if required.

A publication could continue delegation internally. For example, `govdoc.pdf` could itself be a compilation of subdocuments that are delegated by the `govdoc.pdf` author. This ability to delegate and federate both reassignable and persistent identifiers at any level is another unique feature of XRIs that has many applications in referencing, linking, and sharing resources across distributed, multi-authority systems.

---

[3] It should also be noted that XRI resolution supports standard URI or IRI authorities. In other words, the XRI `xri://department.agency.example.org/docs/govdoc.pdf` can be resolved using DNS and HTTP content negotiation to an XRI Descriptor. See section 2.3 of **[XRIResolution]**.

[4] Note that no star appears between "@" and "example.org" in the XRI itself. To make XRI syntax as simple and human-friendly as possible, a star is optional when a reassignable sub-segment follows a global context symbol or a forward slash. However the star is always made explicit when performing XRI resolution. By contrast a bang is never optional—it is always required as the prefix for a persistent XRI sub-segment.

3.3 Shared Identification

XRIs provide a uniform syntax (called "cross-references") for sharing identifiers across all contexts.

Imagine that `govdoc.pdf` goes into wide circulation and the department responsible for it obtains an International Standard Book Number, or ISBN, of 0-395-36341-1. This can be represented as a URN as:

```
urn:ISBN:0-395-36341-1
```

This URN represents a logical resource, not a physical resource. In other words, it identifies the registered identity of the book, not a particular physical copy of the book. If the URN were resolvable, it might reference some information about the book maintained by an ISBN registry, but we wouldn't expect it to resolve to a digital version of the book itself.

On the other hand, a reference to a physical resource *can* be constructed by using this ISBN in a specific context. For example, we might refer to, "The National Library's copy of the book with the ISBN 0-395-36341-1." Notice that the ISBN number *in the context of* the National Library represents a different resource than the ISBN number by itself – a particular copy of the book as opposed to the notion of the book. XRIs provide an elegant way to express the concept of context-based identification. Let's say the National Library has an XRI of:

```
xri://@example.org*national.library
```

We can now create an XRI that expresses the concept of "The National Library's copy of the book with the ISBN 0-395-36341-1":

```
xri://@example.org*national.library/!(urn:ISBN:0-395-36341-1)
```

We can also create both reassignable and persistent way to express the concept of, "The government agency department's representation of the book with the ISBN 0-395-36341-1":

```
xri://@example.org*agency*department/!(urn:ISBN:0-395-36341-1)
xri://@!9990!A58F!1C3D/!(urn:ISBN:0-395-36341-1)
```

The same ISBN number in the context of a book store would represent yet another resource:

```
xri://@example.bookstore/!(urn:ISBN:0-395-36341-1)
```

XRIs allowed us to reuse the identifier `urn:ISBN:0-395-36341-1` consistently across each of these contexts by expressing it as a cross-reference, syntactically distinguished by parentheses. In essence, all of the authorities represented above are able to share the same identifier for the same logical resource (the publication we know as `govdoc.pdf`) in a manner that is understandable to both people and machines, yet allows of them to designate the "meaning" (i.e., resolution) in their own context.

The ability to share identifiers across contexts using compound identifiers is very powerful. It allows us to express ideas like, "The agency's employee with the email address 'bob@example.com'":

```
xri://@!9990!A58F!1C3D/employees*(mailto:bob@agency.example.org)
```

Of course XRIs themselves can also be used in the context of other XRIs. For example, say Bob had registered the following personal XRI under the XRI global context symbol, "=" (used to represent individual persons as authorities.)

```
xri://=example.bob
```

Now Bob has the option to use this personal identifier in the various contexts in which he might be known or have a relationship. For example, if Bob was an employee of the government agency, both of the following XRIs could express this relationship:

```
xri://@example.org*agency*department/employees*(=example.bob)
xri://@!9990!A58F!1C3D/!24*(=example.bob)
```

Another use of cross-references for shared identification is identifier metadata, such as date/time stamps and version identifiers. The global context symbol "$" is reserved for XRIs specified by standards bodies for this purpose. The first two such specifications are **[XRIResolution]**, which establishes the top-level namespace `$res` for identifiers needed in XRI resolution, and **[XRIMetadata]**, which establishes $ namespaces for XRI language tags, date/time stamps, versioning, and annotations. Following are several examples from **[XRIMetadata]**; see the specification for more.

```
xri://@France*($l/fr)/pays)*place
xri://@example/resource*($d/1994-11-05T08:15:30-05:00Z)
xri://@example/resource*($v/3.04.2)
xri://!!1000*($-Example%20Corp.)!5678*($-West%20Coast)/resource
```

Cross-references are also the key to XRI extensibility, further discussed in section 3.7.

3.4 Simplified Identification

> *XRI provide both a standard and a trusted protocol for resolving a single abstract identifier into any number of concrete identifiers.*

XRIs permit the abstract identification of a resource independent of location and interaction protocol. That means a single XRI can be used to represent the same resource stored in multiple locations and accessed through multiple protocols. Discovering those concrete endpoints and protocols is the job of XRI resolution, which leverages both the broadly deployed HTTP and HTTPS infrastructure and the flexibility of XML.

XRI resolution is a two phase process. The first phase, *authority resolution*, resolves to the XRI authority responsible for the resource. The second phase, *local access*, uses URIs and metadata from the authority to interact with the identified resource. During authority resolution each sub-segment in the authority segment is resolved to an XML document that lists concrete locators (such as HTTP URIs) for the named resource.

Let’s look again at

```
xri://@example.org*agency*department/docs/govdoc.pdf
```

The first phase of resolution operates on the XRI Authority portion of the XRI, in this case, `@example.org*agency*department`. The mechanics of XRI resolution are covered in **[XRIResolution]**, but the final result is an XML document called an XRI Descriptor (often abbreviated as XRID) that describes the XRI Authority. The XRID for `@example.org*agency*department` might look something like this:

```
<xrid:XRIDescriptor xmlns=”…”>
   …
  <xrid:Service>
    <xrid:Type>
      xri://$res*local.access/X2R
    </xrid:Type>
    <xrid:URI>
      http://department.agency.example.org/
    </xrid:URI>
    <xrid:URI>
      https://department.agency.example.org/
    </xrid:URI>
  </xrid:Service>
  …
</xrid:XRIDescriptor>
```

The `xrid:Service` element describes a service offering of type `xri://$res*local.access/X2R`. This is a very simple service defined by the XRI resolution protocol that allows a resource to be accessed using the semantics defined by HTTP. In this case, `govdoc.pdf` may be retrieved by choosing one of the listed URIs, appending the path portion of the XRI and performing an HTTP GET on the result. The path portion of original XRI was:

```
/docs/govdoc.pdf
```

Appending this to one of the listed URIs in the `xrid:Service` element above would produce:

```
http://department.agency.example.com/docs/govdoc.pdf
```

This is one URI that can be used to perform HTTP access on `govdoc.pdf`. Since the XRID also indicates that HTTPS access is supported, a client application can calculate a second URI that can be used for secure access:

```
https://department.agency.example.com/docs/govdoc.pdf
```

When the location(s) or retrieval protocol(s) supported by this government agency department change, only its XRI Descriptor needs to change. In addition, the XRID can list multiple locations, access protocols, media types etc. supported by this XRI authority.

This XML-mediated abstraction layer can significantly simplify the process of identifying and interacting with any resource maintained by an XRI authority, whether this authority is an organization, a person, a community, etc.

3.5 Metadata Identification

> *XRIs provide a simple, standard means of discovering URIs that may be associated with a resource, including those needed for additional metadata discovery.*

XRI Descriptors are not limited simply to providing URIs. They can also publish additional metadata about the types of resources available from an authority. To facilitate metadata discovery, XRIDs natively support a `MediaType` element as shown in the following example:

```
<xrid:XRIDescriptor xmlns="…">
   …
  <xrid:Service>
    <xrid:Type>
      xri://$res*local.access/X2R
    </xrid:Type>
    <xrid:URI>
      http://department.agency.example.org/
    </xrid:URI>
    <xrid:URI>
      https://department.agency.example.org/
    </xrid:URI>
    <MediaType>
      application/pdf
    </MediaType>
    <MediaType>
      application/rdf+xml
    </MediaType>
  </xrid:Service>
  …
</xrid:XRIDescriptor>
```

This element would, for example, allow a client application to discover that an RDF description of `govdoc.pdf` may be available.

XRIDs are also not limited to descriptions of local access services—the XML schema is extensible and can include elements from other namespaces. For example, in XRI trusted resolution, where the goal of resolution is to obtain a response with some assurance of authenticity, XRIDs contain SAML assertions issued by the XRI authority producing the response. See section 3.3.1 of **[XRIResolution]** for more details.

Following this extension model, XRIDs can also include any other metadata the authority chooses to publish. For example, the XRI Descriptor above might have included the following:

```
<xrid:XRIDescriptor xmlns="…">
   …
  <other:AuthorityDetails>
    <other:TechnicalContact>
      mailto:bob@agency.example.org
    </other:TechnicalContact>
    …
  </other:AuthorityDetails>
  …
</xrid:XRIDescriptor>
```

The `AuthorityDetails` element is defined in an external schema identified by the `other` namespace. Resolvers that understand this element can process it, while those that don't are free to ignore it.

It is important to note that generic XRI resolution, like DNS resolution, is public, i.e. it does not require client authentication and it does not return different results for different requestors. When using generic XRI resolution, XRI Descriptors should contain only information that is appropriate to share with the public.

3.6 Privacy-Protected Identification

XRIs provide a standard means of protecting privacy by revealing the least possible information in an identifier and allowing an authority to control further access.

The proceeding example of including an email address as a public identifier highlights another benefit of abstract identifiers: they can be constructed to not reveal private information while still providing a means to access such information with the proper authorization.

For instance, the XRI Descriptor in the previous example might have contained the following XRI rather than an explicit email address:

```
<xrid:XRIDescriptor xmlns="…">
   …
  <other:AuthorityDetails>
    <other:TechnicalContact>
      xri://@example.org*agency*department*contact/(=example.bob)
    </other:TechnicalContact>
    …
  </other:AuthorityDetails>
  …
</xrid:XRIDescriptor>
```

This XRI will resolve to an XRID describing how the individual represented by =example.bob might be contacted. For example, it could contain an `xrid:Service` element that specified an HTTP or HTTPS endpoint for submitting a contact request that can be validated according to a local authentication policy.

XRI syntax can also be used to express the complex identifiers that often emerge in federated identity scenarios such as those in SAML, Liberty Alliance, WS-Federation, etc. For example, it's possible to express:

The identity `A` calls `Bob`.

```
xri://@a/Bob
```

The identity in `B's` namespace that `A` calls `Bob`.

```
Xri://@b/(@a/Bob)
```

The identity in `B’s` namespace known by the public identifier `mailto:bob@example.com`.

```
xri://@b/(mailto:bob@example.com)
```

The identity in `B’s` namespace known by the public identifier `=bob`.

```
xri://@b/(=bob)
```

The identity in `B’s` namespace that `A` calls `123`. In this case, `123` might be a pseudononymous identifier, i.e. an identifier used only when A talks to B about that identity.

```
xri://@b/(@a/123)
```

The identity in `B’s` namespace that `A` calls `456` when `A` talks to `C` about that identity. This might be appropriate when `C` is facilitating an introduction between `A` and `B`. In that case, `456` might be used only once, during the introduction, and then discarded.

```
xri://@b/(@c/(@a/456))
```

XRI authority and cross-reference syntax is flexible to enough to accommodate all these use cases in a manner that is interoperable across all these protocols and frameworks.

3.7 Extensible Identification

XRIs provide a standard means of extension without sacrificing interoperability.

Because there will always be new and better ways to identify resources, the “X” in “XRI” reflects the same design principle as it does in “XML”: extensibility-by-design. This is accomplished using XRI cross-reference syntax in two ways.

The first is using cross-references to incorporate identifiers created in new URI schemes. For example, say our fictitious government agency wanted to create a database of all the country’s jurisdictions for referencing in government legislation. It might decide the best way to make these jurisdictional identifiers Web-compatible is to create a new URI scheme. This new `juris:` scheme might be partitioned into cities, counties, and provinces as shown below:

```
juris:city:example.city
juris:county:example.county
juris:province:example.province
```

Despite the fact that this is a brand new URI scheme, perhaps with its own resolution protocol, it is immediately interoperable with XRI syntax because a `juris:` reference can be incorporated into an XRI as a cross-reference. For example, our department could specify one of the provinces over which it has authority as:

```
xri://@example.org*agency*department/(juris:province:example.province)
```

However, if the government is already using XRIs for abstract identification, it also has the option of simply creating a new XRI namespace. One XRI global context symbol, “+”, is specifically designated for generic identifiers—the online equivalent of generic nouns. Any XRI authority can create identifiers in this space that can be cross-referenced by any other XRI authority.

So the government could create a new XRI namespace for jurisdictional references such as the following:

```
xri://+juris
```

This new XRI namespace immediately enjoys all the other standard benefits of XRI syntax and resolution.[5] For example, the government could partition this namespace (and even delegate these partitions to separate authorities) the same way it could with a new URI scheme, but without the need to create a new resolution protocol.

```
xri://+juris*city/example.city
xri://+juris*county/example.county
xri://+juris*province/example.province
```

Now other XRI authorities, such as our example agency department, can reference this namespace using cross-references. For example, the department could specify one of the provinces over which it has authority as follows.

```
xri://@example.org*agency*department/(+juris*province/example.province)
```

Again, the ability of cross-references to share identifiers across contexts, when combined the ability for global context symbols to establish the abstract global context of an identifier, enables XRIs to provide the same interoperable richness for identifiers that XML provides for data.

[5] Note that XRIs in the + namespace may only locally resolvable using XRI “dictionary” authorities. Multiple dictionary authorities may then cooperate to establish common synonyms, eventually evolving broadly adopted generic XRIs the same way generic nouns evolve in human language.

4. A Brief Guide to the Normative XRI V2.0 Specifications

4.1 XRI Syntax

[XRISyntax] serves the same role for XRIs that RFC 3986 **[URI]** serves for URIs and RFC 3987 **[IRI]** serves for IRIs. It includes the following major sections:

- *Syntax Components* defines the normative ABNF grammar.
- *Transformations* defines the rules for transforming a valid XRI into a valid IRI and a valid URI and back again (using these transformations, an XRI may be used any place an IRI or URI may be used).
- *Relative References* covers the rules for dealing with relative XRIs.
- *Normalization and Comparison* covers XRI equivalence.
- *Security and Data Protection* discusses the ways in which compound identifiers may be subject to attack or information leakage.

It also includes an appendix on transforming HTTP URIs into XRIs and a Glossary that serves all three normative XRI V2.0 specifications.

4.2 XRI Resolution

[XRIResolution] defines an optional HTTP/HTTPS-based resolution protocol for XRIs. It includes the following major sections:

- *Generic Resolution* defines the basic two-phase resolution protocol, including the structure of XRI Descriptors (XRIDs), the construction of HTTP(S) URIs for XRI authorities, the use of XRI synonyms for redirects, IRI authorities, local access service, HTTP headers, and HTTP caching.
- *Trusted Resolution* specifies how generic resolution is extended to create a chain of trust between the participating authorities using SAML assertions. It defines the additional XRID elements needed, the client and server behavior, and the special requirements of authorities offering trusted resolution.
- *Extensibility and Versioning* explains the extensibility points built into the protocol and how future versioning of the specification and XML schema will be handled.
- *Security and Data Protection* covers special considerations for the use of XRI resolution.

Appendix A includes the normative XML schema for XRIDs, and appendix B includes the non-normative RelaxNG compact syntax schema.

4.3 XRI Metadata

[XRIMetadata] defines four namespaces under the XRI global context symbol “$” for common types of identifier metadata:

- *Language Metadata* ($l) permits expressing the human language in which an XRI, or an XRI sub-segment, is intended to be understood, interpreted, or pronounced.
- *Date/time Metadata* ($d) provides a standard means of representing the date and time that an XRI was assigned to a resource.
- *Version Metadata* ($v) defines a standard means of representing the version of an XRI or XRI sub-segment.

- *Annotation Metadata* ($-) enables XRI producers to provide human-readable annotations in an XRI, or an XRI sub-segment, without affecting resolution or comparison.

The specification also covers normalization and comparison rules for XRI metadata and explains how these and other forms of XRI metadata can be extended by any XRI authority using cross-references.

5. References

[CENDI] Persistent Identification: A Key Component of an E-Government Infrastructure. http://cendi.dtic.mil/publications/04-2persist_id.pdf, March, 2004.

[IRI] M. Duerst, M. Suignard, *Internationalized Resource Identifiers (IRIs)*, http://www.ietf.org/rfc/rfc3987.txt, RFC 3987, January 2005.

[Lewis] BetaNews, http://www.betanews.com/article/1075980052, February 5, 2004

[XRISyntax] D. Reed, D. McAlpin, *Extensible Resource Identifier (XRI) Syntax V2.0*, http://docs.oasis-open.org/xri/xri/V2.0/xri-syntax-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.

[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.

[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.

[RFC3066] H. Alvestrand, *Tags for the Identification of Languages,* http://www.ietf.org/rfc/rfc3066.txt, RFC 3066, January, 2001.

[SAML] S. Cantor, J. Kemp, R. Philpott, E. Maler, *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0,* http://www.oasis-open.org/committees/security, March 2005.

[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.

[XMLSchema2] P. Biron, A. Malhotra, *XML Schema Part 2: Datatypes Second Edition W3C Recommendation,* http://www.w3.org/TR/xmlschema-2/, October 2004.

[RFC3066bis] A. Phillips, M. Davis, *Tags for Identifying Languages,* http://www.ietf.org/internet-drafts/draft-phillips-langtags-09.txt, Work-In-Progress, January, 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. Example XRIs

The purpose of this appendix is to provide a consolidated set of XRIs that help illustrate key concepts of XRI syntax. (Note the heavy use of “example” and “!1000” for authority names in order to prevent the use of real authorities.)

Fully Persistent XRIs

`xri://@!1000!123!0-395-36341-1`

`xri://@!1000!1234/!rfc!2141`

This XRI includes a persistent sub-segment that is “semantically reflective”, i.e. it has meaning that is apparent to the reader. This is supported in XRI syntax, but should generally be avoided.

`xri://!!1000!34F2!A98E!B8FC/!3283`

This XRI is rooted on the bang (“!”) global context symbol, the context for persistent identification of physical resources, independent of a person or organization.

Fully Reassignable XRIs

`xri://@example.gov*agency*department/docs/govdoc.pdf`

`xri://@example/john.doe`

`xri://=example.john.doe`

This XRI is rooted on the equals (“=”) global context symbol, the context for identification of individual persons.

`xri://=example.john.doe*home`

`xri://=example.john.doe*work`

`xri://=example.john.doe*work/(+phone)`

This XRI contains a cross-refererence rooted on the plus (“+”) global context symbol, the context for generic identifiers. The interpretation of these identifiers is established by convention and/or shared XRI dictionaries.

`xri://xri.example.com/john.doe`

This XRI is rooted on a DNS name.

`xri://(http://example.com)/john.doe`

This XRI is rooted on a cross-reference, which allows any local community to establish a root authority without registering with a central authority.

XRIs with both persistent and reassignable sub-segments

`xri://@example.bookstore/!1E2F!34A2`

`xri://@!1000!3432!2397/documents/file.txt`

`xri://@!1000!3432!2397/documents/example.xml#34234`

XRIs support the same query and fragment syntax as URIs and IRIs

`xri://@example.gov*national.library/!(urn:ISBN:0-395-36341-1)`

This XRI has a persistent cross-reference (URNs satisfy the requirements for persistent XRI segments).

Appendix B. 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 Inc
- Victor Grey, PlaNetwork Inc
- Fen Labalme, PlaNetwork Inc
- 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.

Appendix C. Revision History

Rev	Date	By Whom	What
01	2/24/2005	Drummond Reed	Initial document.
02	3/3/2005	Drummond Reed, Dave McAlpin	Revised sections 1 and 2, scenarios developed for section 3
03	3/14/2005	Dave McAlpin, Drummond Reed	Editorial revisions to sections 1 and 2, rough draft of almost all of section 3
04	3/14//2005	Dave McAlpin, Drummond Reed	Complete draft including appendix A

Appendix D. Notices