By Steven Churchill
Nov. 25, 2007
steven.churchll@ootao.com

Introduction

I voted yes on the ballot to promote XRI Resolution WD11 to OASIS committee draft because I feel that the specification (the "spec") captures enough of the underlying resolution model that conformance among implementations can be achieved in practice. However I am in disagreement with the spec in some key architectural areas, and I thus present this document as a way to capture these disagreements.

My positions on these matters are well known among the spec editors, and this document does not attempt to formalize arguments. Rather, it serves as a rough compilation of arguments I have presented earlier to the TC. This is done in the event that it may help others to formalize and improve the spec architecturally as it moves through standardization process.

This document is structured as a simple list of two primary "disagreements" followed by a set of appendixes that contain versions of arguments previously submitted to the TC.

Disagreements

1. The Resolver functional interface is overloaded by URI resolution.

Until recently, the functional interface for XRI resolution (the XRI *Resolver's* functional interface) defined a mapping from an XRI (identifier) to a node in the XRI authority graph. This interface has now been overloaded with the new functionality of mapping a URI to … well, to some other abstraction. (See section 5.1 of the spec.)

The XRI resolution spec should be about—just that—XRI resolution. Functional interfaces for URI resolution should be left to other specifications. (Note that an XRI resolver, such as the OpenXRI resolver implemation, may indeed implement multiple interfaces. It seems that we have lost this basic architectural consideration.)

This overloading has led to the relatively recent introduction of constructs EquivID and CanonicalEquivID whose genesis evolved from requirements in the URI resolution space (specifically, the OpenID "recycling" problem.) Since XRI resolution should not be overloaded by URI resolution in the first place, these constructs also represent a fundamental overloading and should be removed. XRI Resoution already has a means of traversing polyarchical edges in an authority graph, and that is the Ref construct. (See appendix C below.) This construct is sufficient.

2. The resolution model's fundamental abstractions are misrepresented.

XRI resolution has an elegant underlying model, and that model is captured and enforced by existing reference implementations such as the OpenXRI and Barx resolvers. Ironically, the model is not represented well by the spec.

I discuss these abstractions in appendix A below. There is an important distinction between (1) the mapping of an identifier "to an XRD" and (2) the mapping of an identifier to an entity defined within a formalized model—such as the XRI authority node. The spec consistently represents XRI resolution as the former, and this leads to erroneous concepts such as "distributed XRDS management" being used throughout the text (section 12.)

Appendix B below discusses why the term "synonym" should be used in the context of identifiers, such as: *I want to treat identifier A to be synonymous with identifier B because they both refer to the same identity.* The spec incorrectly tries to use the term synonym to categorize attributes of the identity itself (sections 5, 14, etc.)

Finally, appendix C introduces the XRI authority graph itself and explains the notions of polyarchical relationships and CanonicalID as an attribute that can be used to distinguish identity. These fundamentation concepts are not captured in the spec.

Appendix A: On Identifiers and Identity

This is from an email I sent to the XRI list August 21, 2007. I have edited it here for context.

Introduction

Much of the continuing delays and confusion in the XRI Resolution Specification stem from a fundamental lack of understanding of the underlying identity model. Much of this arises from the failure to make the important distinction between the notions of identifier and identity.

The truly amazing thing is that despite these attempts, the XRI Resolution Specification today still stands strong upon the foundation of its identity model. This makes XRI resolution an elegant and exceptionally strong framework for solving many real-world problems for years to come.

In this email, I will lay out the core constructs of the identity model that underlies XRI resolution.

Identifiers and identity

The identity model of which I speak is quite simple: **identifiers** have a **means of associating** them with some entity, where **identity** is the set of characteristics which distinguish one entity from another.

For example, the two **identifiers** "Steven Churchill" and "Steve Churchill" can be associated with the entity (the human being) who is typing out this email. My **identity** is the set of characteristics which distinguish one human being from another (my DNA, my "soul"). The **means of associating** my names to my identity might simply be a social convention where people refer to me using these names (my social identity model), or it might me a more formal means, such as that used by my bank (the bank's identity model.)

XRI has an identity model illustrated in this example: the two **identifiers** =steven.churchill and @ootao*steven can be associated with the XRI authority that has the CanonicalID value of `=!C5FB.53B6.6E94.824`. The **identity** under this model is determined by the value of the CanonicalID—this is the characteristic of the XRI authority that distinguishes it from all other XRI authorities. The **means of associating** the identifiers to the identity is that of XRI Resolution—or more specifically, the act of performing XRI resolution with a given set of input parameters.

For example, if you click on the first two links below, you will be using XRI resolution as the means of associating the two identifiers to an identity (an XRI authority.) You can see here that the two identifiers =steven.churchill and @ootao*steven have been associated with an XRI authority with the CanonicalID value (identity) of `=!C5FB.53B6.6E94.824`. The third link is to show that, indeed, the means of association is tied to the resolution input parameters—clicking on it renders a different identity than does clicking on the second link. Same identifier, different identity.

http://beta.xri.net/=steven.churchill?_xrd_t=http://openid.net/signon/1.0&&_xrd_r=application/xrd+xml;sep=true;
http://beta.xri.net/@ootao*steven?_xrd_t=http://openid.net/signon/1.0&&_xrd_r=application/xrd+xml;sep=true;
http://beta.xri.net/@ootao*steven?_xrd_t=xri://+iservice*(+contact)*($v*1.0)&&_xrd_r=application/xrd+xml;sep=true;

(See the article at http://dev.inames.net/wiki/XRI_CanonicalID_Verification for more information about why this is so.)

So what?

The important thing to take away from this discussion is this "precious distinction" between identifier and identity. "=steven.churchill" and "@ootao*steven" are both **identifiers**. XRI resolution provides a **means of associating** them with an entity that we call an XRI authority. Each XRI authority has an **identity** which is

the characteristic that separates it from all other XRI authorities—in our case, it is the value of its CanonicalID.

I have been told that this "precious distinction" does not exist. Here's how Drummond puts it: "but a CanonicalID is itself an identifier… thus there is really not a distinction between identifier and identity." I would advise not to allow yourself to fall prey to this thinking. The fact that the CanonicalID also happens to be used as an identifier in some contexts *does not in any way alter the fact that the CanonicalID is the characteristic that distinguishes one XRI authority from another*. The model has a clear and real notion of identity and thus a clear and real distinction between identifier and identity. [I know that Drummond still strenuously disagrees with me on this fundamental precept.]

By formalizing the CanonicalID (and its verification), XRI Resolution can be said to **formally support only a single identity model.** This is the identity model that uses CanonicalIDs to distinguish the identity of XRI authorities. It should be stressed, however, that this identity model (where CanonicalID determines identity) is only one of a multitude of identity models allowed (although not formally supported) by XRI resolution. XRI clients need not ever care about CanonicalIDs, and conformant authority resolution services may not ever return a CanonicalID—yet this does not prevent a client from using other quite valuable and meaningful ways of associating XRI identifiers with some notion of identity.

Authorities, XRDs, Resources, Registries

Unfortunately, the XRI TC still suffers confusion regarding these concepts.

What is an XRI authority? It is the entity in the above XRI identity model to which we ascribe identity. It is the real-world entity (living in a real namespace registry) encapsulating the data for local identifiers, Refs, service endpoints, and so forth—those things that often appear as metadata in an XRD. It is the record in Les' "global database" and the node in my hierarchical graph. (These are metaphors discussed in previous emails.) Whereas the XRI authority encapsulates the actual data, metadata about the XRI authority is described using an XRD as explained next.

What is an XRD? It is a bit of XML used to describe metadata about some "resource". Under XRI Resolution, the XRD provides metadata describing the actual data of the XRI authority. (So I guess this makes an XRI authority a type of resource.) But an XRD is also used to describe other types of resources: consider its usage under YADIS.

So when I go to **provision** the data for my XRI authority that is distinguished by the CanonicalID `=!C5FB.53B6.6E94.824` (that entity for which I shell out a few bucks every year) I know that my provisioning will show up as metadata in an XRD used to describe my authority under XRI resolution. (To be sure, not all my authority's data will show up in the metadata. For example if I provision my authority with a new local identifier *steve.churchill, and if I obtain an XRD by resolving =steven.churchill, then I will not see that new local identifier, even though that is a real part of my authority's data.)

Like failing to make the distinction between identifier and identity, the TC also commonly fails to make the clear distinction between an authority's actual data and an XRD's metadata.

A final important construct is what I refer to as the **XRI namespace registry**. Each non-leaf node in my hierarchical graph represents such a registry—a namespace for its child authorities. Les' global database metaphor can be thought of as being partitioned into these namespace registries. For example, the = and @ namespace registries are hosted at NeuStar. The following diagram explains the role of the namespace registry in relation to the other model constructs already presented above.

XRI Resolution:

Understanding the difference between identifier and identity

The land of identifiers

XRI Resolution (via XRI Resolver)

- Implements the association between an XRI identifier (given a set of input parameters) and an XRI Authority.
- Note that the only identity model *formally* supported by XRI resolution is that where the authority's identity is distinguished by its CanonicalID.

XRI Resolution Client

- Invokes the Resolver to establish an association between a given identifier and an XRI authority.
- May wish to use the "supported" identity model—that is, the one where CanonicalID distinguishes identity.

  For example, it could treat XRI identifiers that resolve to a given CanonicalID as *synonyms* to the XRI authority using that CanonicalID to distinguish its identity.
- May instead wish to use another model of identity and synonymity. May in fact wish not to support the notion of identifier synonymity at all.

The land of identity

XRI Authority

- Contains provisionable data, such as service endpoints, Refs, local identifiers (for example, *steve and *steven), contact agent (for GRS global authorities), etc.
- Contains non-provisionable data, such as the CanonicalID. (This determines the authority's identity!)
- Is managed by a namespace registry.

XRI Namespace Registry

- Stores XRI authorities within the same XRI namespace. This can be modeled as a node in the hierarchical authority graph.
- Assigns (local) CanonicalIDs to newly created XRI authorities. Well-behaved registries will never re-assign a CanonicalID to a second authority.
- Provisions XRI authorities—that is, allows the data associated with the authority to be modified.
- Provides the authority resolution service for the namespace: given a local identifier for an authority, this returns a metadata description of the authority in the form of an XRD.

Appendix B: Use of term “synonym”

This is from another email I sent to the XRI list August 21, 2007. I have edited it here for context.

Appendix A explains that our identity model contains a *means of associating an **identifier** with an entity, where **identity** is the set of characteristics that distinguish one entity from another.* In XRI land, the entity is the XRI authority, where the single characteristic that distinguishes one authority from another is the CanonicalID. The identifier is the XRI that is passed to an XRI resolver as the means for associating that identifier with its authority.

If you examine the diagram above, you will notice that the concept of synonymity appears only on the left side of the diagram (in identifier land.) This is because the concept exists only within the framework—the identity model—of the XRI client application. For example, if a client gets back the same CanonicalID when resolving =steven.churchill and @ootao*steven then the client is free to use an identity model that considers these identifiers as **synonyms** for a single identity. (And thus, for example, the client is free to store the CanonicalID as the PK for its user account.) But many client applications might consider this notion absurd and instead have different notions of identity and synonymity—or none at all.

The Resolution Specification makes the conceptual error of using the term “synonym” in regard to the right side of the diagram (in *identity* land). This is in an attempt to categorize a set of the authority’s data elements and call them “synonyms”. The spec incorrectly refers to CanonicalIDs, LocalIDs, Refs, EquivIDs, and the like, as synonyms. The terms use in this context is arbitrary and obfuscating.

Like beauty, synonymity is in the eye of the beholder. Only the client application can determine the synonymity model, and the client application only applies this notion to identifiers.

(Okay, there is one exception to this rule. It is okay to think of a “local identifiers” such as *steve and *steven” as being “local synonyms” within the same registry namespace. This notion occurs on the right side of the diagram.)

Appendix C: The XRI Authority Graph model for CanonicalID verification.

This following is text that I had proposed for the spec in June 2007 that describes the underlying authority model, the meaning of Refs, and polyarchical relationships. For another article on these topics, see http://dev.inames.net/wiki/XRI_CanonicalID_Verification.

1.1 Canonical ID Verification Graph Model

The Canonical ID Verification Graph Model establishes the foundation of identity required for the Canonical ID verification of XRI synonyms.

1.1.1 Synonyms and absolute identity

XRI synonymity is one of the most important features of XRI Resolution. Identifier synonymity cannot exist outside of an abstract model that formalizes the absolute identity of the object for which two identifiers purport to be synonyms. In the Canonical ID Verification Graph Model, this object is the *authority node* within the graph. Its absolute identity is defined by its *canonical identifier path*, which is represented using the `xrd:CanonicalID` element of its XRD. (Definitions of italicized terms such as *canonical identifier path* are given in section 1.1.7.)

Applications that support XRI synonyms may require a mechanism to safely verify that a given XRI is truly synonymous with the object (the authority node) to which it purports synonymity. Otherwise, the XRD returned for the purported synonym may contain illegitimate XRD metadata, such as a spoofed endpoint for an authentication service. Canonical ID verification provides this mechanism for safely binding the XRI synonym to the authority node's canonical identifier path.

XRI Resolution does not preclude alternative models of XRI synonymity and/or object identity, however the XRI Resolution Specification provides no mechanism for synonym verification under such alternative models.

IMPORTANT: The graph model presented in this section describes relationships that validate under Canonical ID verification. The model and its terminology are not intended for use outside of this constraint.

1.1.2 Graph Notation and Conventions

When an XRI is resolved, the authority segment of the XRI represents a path through the Canonical ID Verification Graph. Each XRI subsegment traverses an edge to the next child authority node within the graph as shown in Figure 0-1.

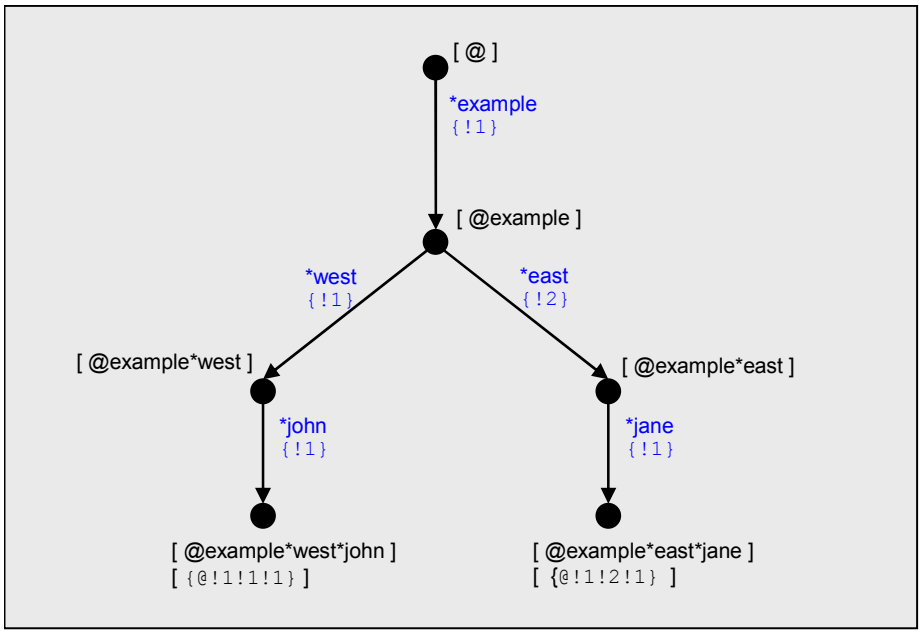

Figure 0-1. Canonical ID Verification Graph diagram.

1.1.3 Edges and local synonyms

The solid-line edges of the graph represent hierarchical parent-child relationships. (Under a hierarchy, a given child node has at most a single parent.) Polyarchical parent-child relationships (where a child node can have multiple parents) are denoted by dashed lines and are covered in section 1.1.5.

The hierarchical edges are labeled with one or more *local synonyms*. Local synonyms establish the naming relationship between a parent authority node (in its context as an Authority Resolution Service) and the child authority nodes within its namespace. Under XRI Resolution, each local synonym is addressed by the names of the subsegments in the authority segment of the XRI providing the path through the graph. For example, resolving the XRI `@example*east*jane` will result in traversing the edges containing the local synonyms `*example`, `*east`, and `*jane`, and will produce the XRD for the node in the lower right corner of Figure 0-1.

In the graph diagram, the labeled local synonyms may include one or more *local canonical identifiers.* These are surrounded in the diagram with curly braces, as in `{!1}`, and they are represented using the highest priority `xrd:LocalId` element(s) of the child node's XRD.

The Resolver "queries" an Authority Resolution Service with the name of the local synonym in order to obtain an XRD describing the given child. The local synonym may be returned in the `xrd:Query` element of the resulting XRD.

1.1.4 Nodes

Nodes in the graph represent the "absolute identity" of the XRI authority, which, as stated above, is a required property for verifying the synonymity of two XRIs. Under the constraints of the Canonical ID Verification Graph Model, the local synonyms of each hierarchical edge must contain at least one local canonical identifier. Thus authority nodes can be absolutely identified by a *canonical identifier path* containing one local identifier subsegment for each edge up to the root. The identifier `{@!1!2!1}` in Figure 0-1 is an example of such a canonical identifier path.

Note that nodes in the diagram may be labeled with square brackets containing XRI synonyms that resolve to the given node.

1.1.5 Polyarchical parent-child relationships

Figure 11-2 shows both hierarchical and polyarchical parent-child relationships. Polyarchical relationships are represented in the diagram with dashed-line edges.

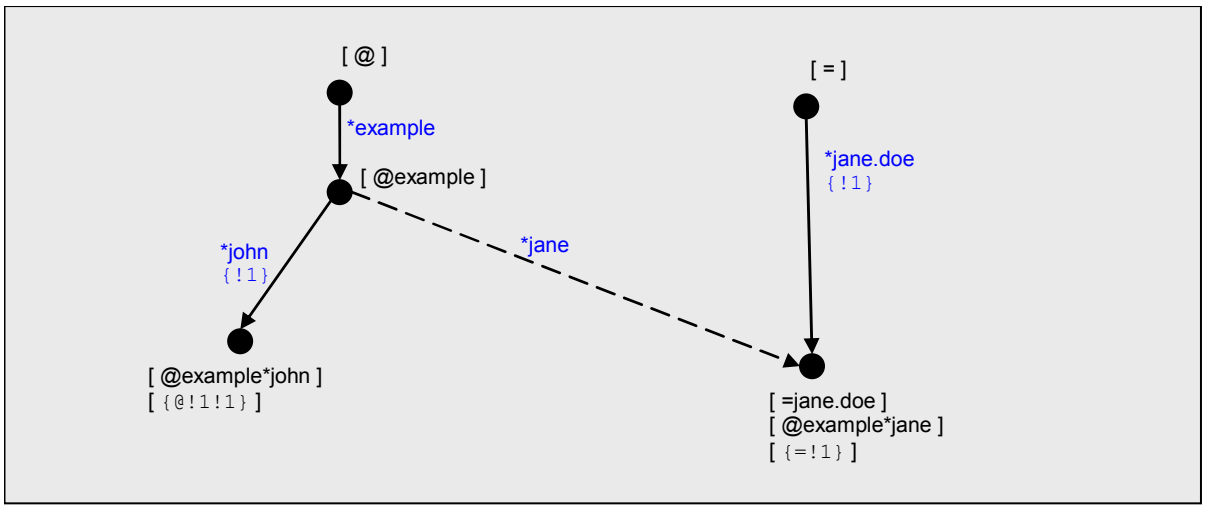

Figure 0-2: Polyarchical parent-child relationship.

This diagram indicates that `@example*jane` and `=jane.doe` are both verifiable synonyms for the authority node with canonical identifier path `{=!1}`.

Because the Canonical ID Verification Graph Model is constrained to those relationships that validate under Canonical ID verification, any mechanism for establishing a polyarchical relationship must be subject to this verification. The only mechanism for defining such verifiable polyarchical parent-child relationships is the `xrd:Ref` element.

An `xrd:Ref` element forms a relationship from one authority node to another, however it is not a parent-child relationship—its semantics are different. In essence, a Ref tells the Resolver, "if you cannot find the service you're looking for in my XRD, then replace my XRD by following this Ref, and look there." *[N.B. these are the "old" Ref semantics. The Ref semantics have changed under the final WD11 with the addition of service-level Refs.]* Whereas these "replacement" semantics are clearly not "parent-child semantics" they effectively *create* a parent-child relationship between the parent of the node containing the Ref and the Ref's target.

Figure 0-3 uses the dotted line to represent the Ref for the polyarchical parent-child relationship "created" by the Ref in Figure 11-2.

Note that the relationship denoted by the dotted line (the Ref relationship) does not exist within the Canonical ID Verification Graph. The Ref is shown in

Figure 0-3 only to illustrate how the polyarchical edge of Figure 11-2 was created. Instead, only parent-child relationships exist within the graph model. (Recall that edges in the graph represent subsegment traversal, as explained in section 1.1.3.)

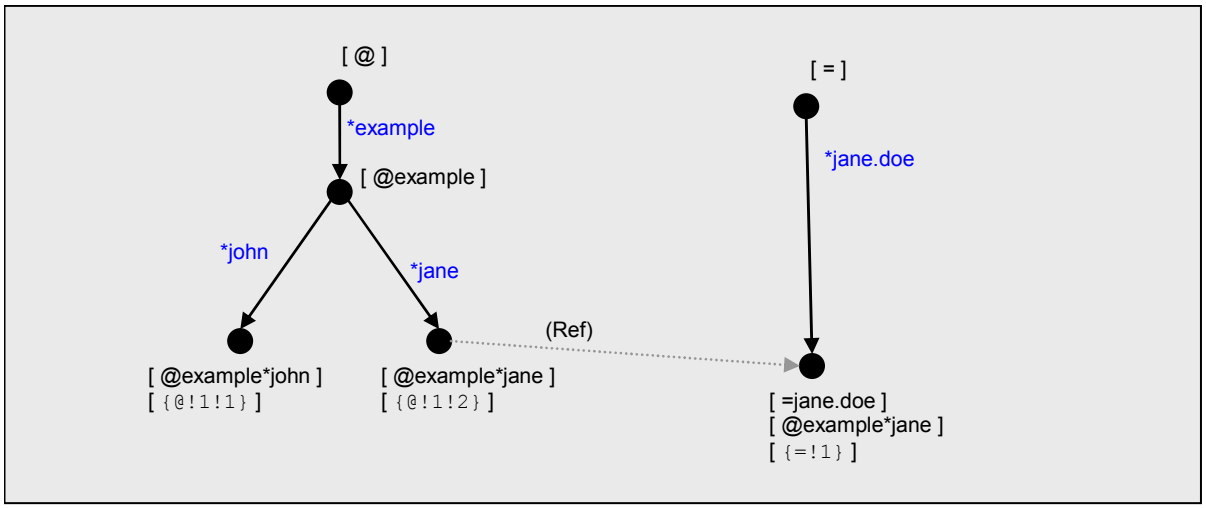

Figure 0-3. Showing the Ref that creates the polyarchical edge of Figure 11-2.

Under CanonicalID verification, a canonical identifier path is not valid unless it satisfies the rules presented in section **Error! Reference source not found.**. During Ref processing, the Resolver applies these verification rules to any XRI constituting the target of a Ref. Therefore, XRI synonyms that contain polyarchical edges as created by Refs are verifiable within the model.

1.1.6 Resolver input parameters

Under XRI resolution, the synonymity of two XRIs is defined only with respect to Resolver input parameters. That is, two XRIs can be synonymous with respect to one set of input parameters whereas they are not synonymous with respect to another set of input parameters. This is because an `xrd:Ref` may be followed under one set of parameters—establishing a given polyarchical parent-child relationship—whereas a different Ref (or no Ref at all) may be followed under a different set of parameters (establishing a different relationship.)

For example, under one set of Resolver input parameters, the XRI `@example*jane` may resolve to the authority node with canonical identifier path `{=!1}` shown in Figure 11-2, whereas under another set of input parameters it may resolve to the authority node with path `{@!1!2}` shown in

Figure 0-3. The difference in input parameters may be something as simple as a difference in the requested service type (xrd_t).

1.1.7 Definitions and Constraints

1.1.7.1 Local synonym

Local synonyms establish the naming relationship between a parent authority node (in its context as an Authority Resolution Service) and the child authority nodes within its namespace. There must be at least one local synonym for each parent-child relationship, and all local synonyms must be unique across the given authority node's children. When the Resolver queries an Authority Resolution Service with a local synonym, the local synonym is returned within the xrd:Query element of the XRD. Local synonyms must be a single XRI subsegment. Local canonical identifiers (section 1.1.7.2) are local synonyms.

1.1.7.2 Local canonical identifier

Local canonical identifiers are types of *local synonyms* that establish the canonical identity relationship between a parent authority node (in its context as an Authority Resolution Service) and the child authority nodes within its namespace. There must be at least one local canonical identifier for each parent-child relationship, and all local canonical identifiers must be unique across the given authority node's children. When the Resolver queries an Authority Resolution Service with any local synonym, the local canonical identifiers are returned in the XRD's highest priority `xrd:LocalID` elements.

1.1.7.3 Canonical identifier path

An authority node's absolute identity is established by one or more *canonical identifier paths.* When the Resolver queries an Authority Resolution Service with any local synonym, the canonical identifier paths are returned within the XRD's highest priority `xrd:CanonicalID` elements. A canonical identifier path is valid only under the rules defined in section **Error! Reference source not found.**.

An authority node may have multiple canonical identifier paths (multiple highest priority `xrd:CanonicalID` elements) but, because the authority node has only one absolute identify, each canonical identifier path must be a *proper synonymous path*, as described in section 1.1.7.5. Lower priority `xrd:CanonicalID` elements represent legacy canonical identifier paths resulting from the merging of two XRI authorities. These are not validated under Canonical ID verification and are thus not represented in the graph.

[N.B. The above is no longer true, since in the final WD11 the cardinality of CanonicalID has been changed from n to 1.]

1.1.7.4 Synonymous paths

Paths that lead to the same to the same node. These paths may include *hierarchical* and/or *polyarchical edges*.

1.1.7.5 Proper synonymous paths

Hierarchical paths with the same number of edges that traverse the same nodes in the same order. These include *hierarchical edges* only.