Internet Engineering Task Force
INTERNET-DRAFT
draft-mealling-oid-dns-00.txt

Michael Mealling
Georgia Institute of Technology
Patrik Faltstrom
Bunyip Information Systems Inc.
Leslie L. Daigle
Bunyip Information Systems Inc.
November 22, 1995

Uniform Resource Names, ISO OIDs and DNS

Abstract

This paper describes a "resolution-mechanism"-independent architecture for Unifrom Resource Name (URN) usage and name space management. This non-monolothic architecture allows different components of the name space to be managed by the appropriate level of network authority. This not only integrates well with traditional Internet models, it allows flexibility in choice of implementation of support for each layer of the name space.

An implementation for the architecture, using OIDs and DNS-based infrastructure, is outlined.

Status of this draft

Introduction

This paper describes a "resolution-mechanism"-independent architecture for Unifrom Resource Name (URN) usage and name space management. This non-monolothic architecture allows different components of the name space to be managed by the appropriate level of network authority. This not only integrates well with traditional Internet models, it allows flexibility in choice of implementation of support for each layer of the name space.

An implementation for the architecture, using OIDs and DNS-based infrastructure, is outlined.

This paper has been revised to conform to the jointly-proposed URN syntax that came out of the Knoxville meeting of URN system architects (October 30, 31, 1995).

Issues Concerning Uniform Resource Names

Apart from the basic requirements of Uniform Resource Naming, some specific issues underly the architecture and implementation proposed in this paper:

1. One architecture to serve several communities

   Historically, different communities have developed specialized naming systems. For example, ISBNs and DNS evolved different architectures and implementations as name spaces for different communities. Ideally, URNs must accommodate name spaces from several different communities, and should do so in a way that requires minimal changes to their name spaces.

2. Persistence vs. transcribability of Uniform Resource Names

   In order to achieve some degree of persistence of URNs over time (i.e., to avoid the kinds of expiration problems that URLs have), the name space has to be optimized for structure. However, arguments have been put forward that suggest that URNs will not be usable if they are so arbitrary and complex as to be meaningless to humans -- increasing the number of transcription errors, etc.

3. Support for multiple URN resolution protocols

   For pragmatic reasons, it is apparent that URN resolution should be

capable of supporting multiple protocols. In order to accomplish this there must be some method to identify the name space and the resolution protocol. Within a URN, the name space is identified by a scheme.

4. Modular vs. monolithic name space maintenance

URN resolution can be treated as a "black box" service -- given a URN, return an information resource, a pointer, or some description of the resource. However, to implement a resolution service in such a monolithic fashion would require the maintainer of the service to accurately track information at a very fine level of granularity. As an alternative, a single name space can be handled in a modular fashion, leaving authors closer to the maintenance of their documents, publishers in charge of the editorial content of their collections, service providers in charge of network details, etc.

One of the difficulties with more transcribable names is the very semantics that are ascribed to the identifier strings. This has already caused problems with domain name registration as people rush to register domains that they expect will be sought after, law suits are launched over the usage of particular phrases, etc. The proposed architecture addresses these issues by separating out URN collection identifiers (which are definitive, and not particularly prone to semantic interpretation by human readers) and collection aliases (which are transmutable). This is described in more detail below.

The Architecture

URN Syntax and Semantics

The proposed URN syntax is as follows:

URN:<collection name>[:<CUI>]

where

URN indicates to the client (human or machine) that the rest of the identifier conforms to URN standards (for distinguishability of <collection name> and <CUI>, etc).

<collection name> identifies the collection in which the resource was originally published. URN syntax places no requirements

on the semantics of this identifier, although it is required (for the sake of interoperability of all URN resolution systems) that this be a hierarchical top-level-first slash-delimited name of the authority that assigned the URN, and the name should, through some transformation algorithm, be a valid DNS name.

: designated separator between collection name and the CUI

<CUI> or "collection-unique identifier" is an identifier of no globally-specified syntax or structure (opaque) that is only required to be unique within the given collection. The CUI is optional so that the collection itself can be identified.

Thus, URNs are globally unique by requiring that collection names are unique across the URN name space. Pragmatically, this means that a collection name is brought into existence once, by a collection publisher. Thereafter, the rights to that collection name may be conferred to another publisher, but only resources published by such authorized publishers will contain that collection name, and these URNs will contain that collection name for all time, irrespective of where the underlying resource migrates. Individual collections are left to select object-identifiers in any way they choose, as long as they are unique for that collection for all time. This facilitates the incorporation of existing object name spaces -- an existing library of information can assign URNs for its material by creating a collection
name and using the resources' existing names as the CUI. In order to be usable as CUI strings, the only requirement placed on an object name space is that it never reuse a name. If this is not inherently characteristic of the object name
space (e.g., as in file systems), the assigner of the name may choose to append a unique suffix -- e.g., time stamp or serial number.

A sample URN might look like:

URN:/com/bunyip:a0n12a3r4b5i6t7r8a9r0y12o3p4a5q6u7e8s9t0r1i2n3g

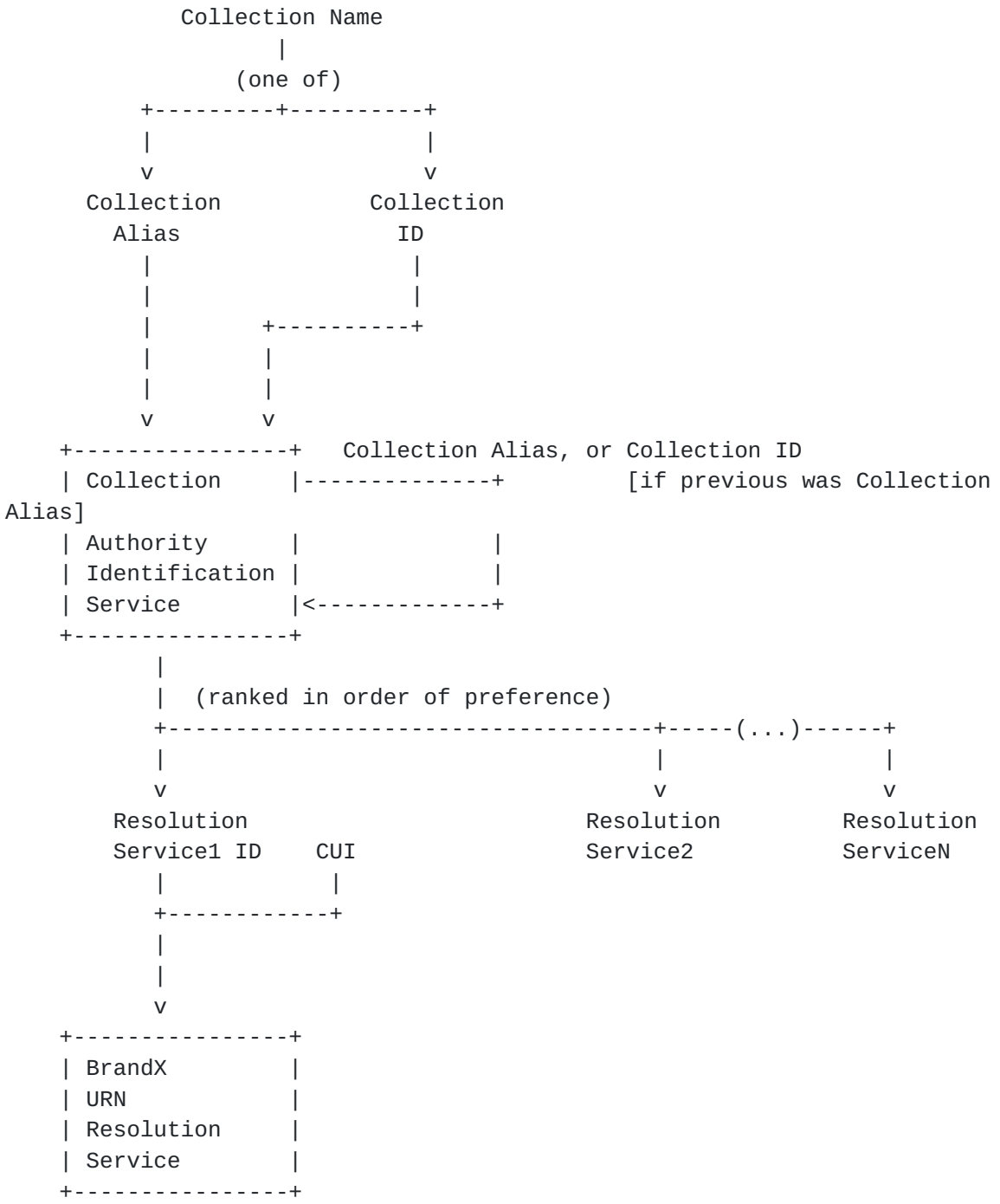
It is proposed that the collection name may be either a Collection Alias (preferred), or a Collection ID. The Collection Alias is a logical name for a collection, while the Collection ID identifies definitively the entity that is responsible for naming authority. Collection Aliases must be resolved into a single Collection ID, although many aliases can exist for a single ID. Collection IDs are associated with one or more resolution services (at different sites, using different protocols, etc).

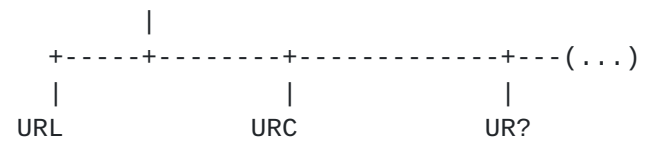
Note that the complete URN identifier consists of the combined collection name and CUI strings. The distinction between the components is provided in the syntax in order to permit the implementation of efficient support mechanisms for partitioning the search space when resolving the URN. That is, the collection's naming authority is the first logical place to try to resolve the CUI. If that is not successful (discussed in more detail below), the URN string remains a valid resource identifier; more

general resolution mechanisms must be applied across a larger portion of the URN name space.

From URN to Resource...

In the ideal case, the process of locating a resource identiried by a URN can be represented as follows:





This process is proposed as a means of handling the URN identifier because it provides modular layers of information management. From the standpoint of the holder of a URN, this means that much of the underlying support for a given URN or collection may change without disrupting the identification capacities of the URN. For example, the resolution service(s) used by a collection may move or change format entirely, without affecting the validity of the URN.

Also, the information maintenance required for each level is isolated to that level. The Collection Authority Identification Service is responsible for maintaining the mapping information from Collection Aliases to Collection IDs, and for Collection IDs to Resolution Service IDs. This can be done straightforwardly by having each collection notify the Collection Authority ID Service of any relevant changes -- which should be relatively infrequent. Each Resolution Service is responsible for maintaining mapping information from the CUI to the URx results. This will require more frequent maintenance, but is localized to a single collection. The holder of the resource at the end of the URx is responsible for that information (e.g., the site holding the document accessible by a URL). As a result of this modularity, each one of these services can be provided by completely independent entities.

From Author to URN...

URNs are assigned through a publishing process. Details of the formality of this process are up to the individual maintainers of collections, and are not relevant beyond the requirement that they yield the unique identifier as described above. Recall that a resource's URN includes the name of the collection under which it was published, irrespective of where the resource migrates over time.

From the author's perspective, a resource is published by submitting it to a publisher. The publisher will store the resource in a collection (either locally, or at some service it subscribes to) and thereafter undertakes to maintain a copy of the resource. A CUI is assigned by the publisher (possibly in conjunction with the collection's resolution service). This resolution service may be run by the publisher, or may also be a service to which the publisher subscribes, and so on up the levels.

The resource may be published under a Collection ID directly, or under a Collection Alias. The latter case is preferred, since any given Collection may use Collection Aliases to further partition and identify its resources.

It is also simpler to confer rights to an entire collection than it is to handle migration of individual resources (discussed below), and it is therefore interesting to keep a fairly fine-grained set of Collection Aliases.

Life of a URN

One of the highly desirable characteristics of URNs is their persistence as identifiers of a resource. That is, the URN should not expire or become unresolvable until or unless the resource to which it was assigned expires or becomes irretrievable (and even then, it would be best of the system could provide an indication of that condition).

The above architecture provides several degrees of freedom in terms of the flexibility of the infrastructure that supports a given URN. For example, if a collection authority changes the underlying resolution service it uses, this simply yields a different reference when the collection name is resolved. Similarly, if a Collection Alias is used in the URN, the rights to the whole collection can be passed to another entity with a completely different collection name authority -- the Collection Alias now resolves to a new Collection ID.

However, this idealized architecture does not handle the case of collections that are split at some point in time. The most obvious example of such an occurrence would be if a single document migrates to another collection. There are some implications of publishing that should reduce the number of such occurrences (e.g., if the act of publishing includes the rights to the material, the author cannot arbitrarily move the document to another collection; arguably, if the author had wanted to maintain that control over the resource, it should have been self-published).

Nevertheless, resources will migrate and collections will be split up. In view of this fact, the Collection Name component of the URN can be viewed as a guide as to where to look first for the resource. The goal of a global Collection Authority Identifier Service is to provide efficient shortcuts to the source of the resource in as many cases as possible. Two things can be done to handle other cases:

1. The resolver can pass back new URNs, when the collection authority remains "friendly" to the migrated resource

2. A global index of "orphaned" URNs can be maintained.

The latter option is to be avoided if at all possible because it carries several inherent difficulties that may be exacerbated over time:

. distance from original author => reduced likelihood of accurate and up to date information regarding resource

. it will only grow...

The first option is proposed because, although it may be assumed that the newer the URN, the less likely it is to be "orphaned", it is not appropriate to assume that old URNs will be accessed only infrequently -- there are definitive versions of some types of resources. Therefore, it would be useful to have the ability to refer enquiries from the original Collection to a newer collection, with the expectation that the client would perform necessary updates and that the number of such referrals would decrease over time.

An Implementation

A Practical Implementation of Collection Identifiers

Since the proposed URN syntax requires that the collection name be algorithmically mappable to a valid DNS name, it is tempting to consider using either the existing DNS name space or one loosely based on it by starting with a new hierarchy but still attempting to use existing names. This may work for small publishers that have very simple name spaces but large organizations with multiple levels of "resource promotion" or movement of authority will have trouble with semantically incorrect names in URNs. For example, if Microsoft purchased Intuit and the rights to all of its URNs, those URNs that have the string "intuit" in the authority portion of the string are semantically incorrect because the entity "Intuit" has ceased to exist.

Nevertheless, DNS is already deployed across the globe, and is recognized as a system that has been designed (and is being improved) to support resolution of global name spaces. It is therefore interesting to consider leveraging its strengths for URNs.

We now consider a semantically unbiased name space that can be mapped into DNS. This name space should have few rules regarding name assignment. In order to have Collection IDs that are free from human-ascribable semantics (as described above) it seems reasonable to assume that the character set must be limited to either all numbers or a few numbers and letters (ala hex). It is also
desired that the rules concerning name assignment be only structured to the extent that sub-authorities can be assigned.

These two preferences allow several solutions, but an existing name space would be preferable since it facilitates acceptance. One existing solution is the International Standards Organization Object Identifier or OID which has been
used by the IETF for several years in identifying MIBs and other objects.

OIDs are based on a hierarchical name space using only numbers delimited by periods with the right most element being most significant. This is how OIDs have been broken down for use by the IETF for various functions:

1	**iso**
1.3	**org**
1.3.6	**dod**
1.3.6.1	**internet**
1.3.6.1.1	**directory**
1.3.6.1.2	**mgmt**
1.3.6.1.2.1	**mib-2**
1.3.6.1.2.1.2.2.1.3	**ifType**
1.3.6.1.2.1.10	**transmission**
1.3.6.1.2.1.10.23	**transmission.ppp**
1.3.6.1.2.1.27	**application**
1.3.6.1.2.1.28	**mta**
1.3.6.1.3	**experimental**
1.3.6.1.4	**private**
1.3.6.1.4.1	**enterprise**
1.3.6.1.5	**security**
1.3.6.1.6	**SNMPv2**
1.3.6.1.7	**mail**

Delegation of authority is unspecified and thus is left up to the owner of a given portion of the hierarchy. For example, 1.3.6.1.4.1.636 is 'owned' by The Georgia Institute of Technology. It could as easily be owned by the Library of Congress. Also, at this time delegation below that is by department. If this organization of authority changes then no change must occur in the name because it is transparent.

Another important feature of OIDs is that they are recognized by ISO and thus are an internationally recognized standard.

DNS as a practical implementation of a Collection Authority ID Service
---

The proposed URN syntax requires that the collection name component be mappable onto a valid DNS name. While this does not mean that the collection name is in any way required to be resolvable to a site name, the DNS infrastructure can be used to support URN resolution. The existing Domain Name System used extensivly on the Internet is by far the largest and most accessible distributed database in the world. The Internet's reliance on it means that most Internet-aware software is already primed with the ability to interact with DNS. For this reason it is a good candidate for one of the possible implementations of a Collection Authority ID Services.

When resolving an OID, we get information about what service to use when resolving other data bound to the OID. You also get the hostname and port number of the server itself. We call this information a Naming Authority

Pointer (NAPTR) record.

Each OID gives us a record with enough data to be know what resolution service that organization, or part of that organization, uses. Note that if one organization uses several resolution services, they can simply delegate OIDs, one for each service, for the different resolution methods.

Collection Aliases, or User Friendly Naming

We also introduce the concept of a UFN ("User Friendly Name" -- since OIDs are not) record in DNS to handle Collection Aliases. (There is no relationship
between this User Friendly Name and a User Friendly Name in X.500).

The UFN itself doesn't have to have the same meaning as the OID, i.e. we can use this aliasing as a mechanism for having the UFN name the publisher, and the OID name the authority which is responsible for the name delegation. This makes it easy to allow publishers to purchase resolution services from entities more suited to the task.

As an example, suppose Acme, Inc. decides to be a publisher of culinary recipes and treatises on obscure points of international law. URNs for the material in these collections could take the form:

URN:/com/acme/recipe:<string particular to the document>

URN:/com/acme/intlaw:<string particular to the document>

The decision to identify an entity as a publisher is distinct from undertaking to maintain an operational URN resolution service. That is, if Acme does decide to run its own service, the UFNs recipe.acme.com and intlaw.acme.com will map to the OID for Acme, Inc. On the other hand, if Acme, Inc. decides that running such a service requires more Internet expertise and/or service than they can supply (e.g., Acme is in fact a cottage industry run out of a cabin in backwoods Vermont), it can pay for resolution services from a provider that specializes in them. In that case, the UFN will map to the provider's OID, and resolution will continue by identifying the types and locations of the provider's resolution services.

For example, the UFN bunyip.com can map to the OID for Bunyip, 1.3.6.1.4.1.1375, but if Bunyip is not interested in managing the OID name space, the UFN bunyip.com can map onto the OID for The Georgia Institute of Technology. This means that when we resolve the UFN bunyip.com, we are referred to 1.3.6.1.4.1.636 for further resolution of other data bound to the bunyip.com UFN.

This flexibility can also facilitate other things like promotion to library grade cataloging and management. It supports persistence of URNs by encapsulating certain maintenance responsibilities at different levels in

the name space. For example, if Acme, Inc. goes out of business, its recipe and international law collections can be moved elsewhere (or made part of another collection), and all that needs to change is the registration that maps the UFNs recipe.acme.com and intlaw.acme.com to an OID. In fact, the two collections can be taken over by 2 separate entities, and the UFNs may now map to distinct OIDs. The semantics understood by users hasn't changed. The name space management has adapted to a very pragmatic approach to supporting the continued existence of a collection.

The UFN record is syntactically (and implementably) identical to the current CNAME resource record except that it is not burdened with CNAMEs restrictive semantics. For example, it is not possible to have both a CNAME record and any other kind of DNS record (e.g., MX record) for one domain name in DNS. (This is one reason why UFN records are proposed, instead of continuing to use CNAME records). A UFN and an MX record can be intermixed with no side effects.

The NAPTR gives the information needed, given an URN collection name, needed for further resolution of the CUI in a URN. The information given is the:

precedence - to allow the client to choose among multiple NAPTR records for a Naming Authority. An unsigned short. Multiple records can mean several possible things:

a) the NA offers multiple resolution methods. Each method gets its own preference as specified the the resolving agent.

b) the NA offers multiple resolving servers around the net as a load sharing mechanism.

c) a combination of a and b

d) an unspecified reason

Client software should choose the record of lowest-precedence that has a 'scheme' (see below) that it recognizes. Beyond that, client developers should not infer any global semantics of precedence numbers.

algorithm - The resolution method used to resolve the end service. Current examples would be 'path','whois++','handle','x-dns-2'. These do not specify the actual protocol used to finally resolve the CUI but instead specify the next step in the resolution process. For example, the client looks up /a/b/c/d and

finds that the method is 'path'. At that point
the client can assume that it can follow the
standard path resolution mechanisms. If instead
it had found 'handle' it could then assume that
it could contact the root and such as prescribed
by the handle method.

host - a DNS compresses hostname to connect to for this service.

port - The port connect to on 'hostname'. An unsigned short.

method-specific-string - a text field that only has meaning when
coupled with the algoirthm from above. For example,
if the algorithm were 'path' then this would be
the place in which a string specifying variable
substitution could be found. If the algorithm were
'whois++' it might contain the whois++ server name
or a template name.

It is important to bear in mind that the Collection Alias concept in the proposed architecture allows mapping from any kind of name to any other kind of name. This can be used by other (non-DNS-space) name spaces to map onto OIDs, or indeed for DNS-space names to map onto other name spaces. To distinguish a Collection Alias from a real Collection ID within a URN, we simply require that any URN authority field must, after 1 or more iterations through the Collection Authority ID Service, map to something that is identified as a bona fide Collection ID.

Examples

Suppose the UFN bunyip.com is resolved by a Whois++ service at The Georgia Institute of Technology, but some Georgia Tech departments use other resolution services. Georgia Tech uses the OID 1.3.6.1.4.1.636.1 for Whois++, **1.3.6.1.4.1.636.2 for DNS etc. The bunyip.com UFN therefore refers to the** OID 1.3.6.1.4.1.636.1, and when looking up information about that OID, we get the following information:

```
OID: 1.3.6.1.4.1.636.1
Preference: 10
Service: WHOIS
Hostname: mordred.gatech.edu
Portnr: 63
```

To be as flexible as possible in supporting multiple resolution services, we also introduce a Preference on the NAPTR record. In this way, a single OID can resolve to several NAPTR records, each with a different metric. This is the same way MX records are handled in DNS. The record with the lowest metric is to be used first, and if that fails, the second lowest is

used etc. It is important to realize that the actual value or original meaning of the of the metric is inaccessible and inherently nonportable. It is therefore not recommended to make assumptions about given reference metrics.

This gives us another method to implement the above possible need for multiple resolution services. Let's say that for redundancy and customer service reasons Bunyip wishes to provide all possible method of URN resolution but that it prefers whois++. It therefore puts several NAPTR records in with different resolution scheme/preference pairs.

If the bunyip.com UFN is mapped to the OID 1.3.6.1.4.1.636.1, that in turn can map into these two NAPTR records:

```
  OID: 1.3.6.1.4.1.636.1
  Preference: 10
  Service: WHOIS
  Hostname: mordred.gatech.edu
  Portnr: 63

  OID: 1.3.6.1.4.1.636.1
  Preference: 20
  Service: HTTP
  Hostname: mordred.gatech.edu
  Portnr: 80
```

We can resolve the collection name into a number of UFN or NAPTR records. If it is a UFN then it is further resolved into a Collection ID name which is then queried for an NAPTR record. Each NAPTR record is then used in the order the Preference states to resolve the CUI in the URN.

The full resolution process is now:

```
   Get the URN
    Look up the Collection Name of the URN and ask for both UFN and NAPTR
        records.
    If there is a valid UFN record for this then
        Look up one or several NAPTR records that the UFN points to;
    else if there is one or more valid NAPTR record for this then
        fall through;
    else die;

    For each NAPTR found above,
        contact the services in the order specified by the Preference,
        Look up a URC using the CUI
    until we have a URC
```

An example DNS entry follows:

```
   bunyip.com. IN UFN  1.3.6.1.4.1.636.1
```

```
   1.636.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          10                   ; metric
          mordred.gatech.edu.  ; hostname
          63                   ; port nr
          "whois"              ; protocol
          "GATECH01"  )        ; auxiliary data for Whois
```

or

```
   gatech.edu. IN UFN  1.3.6.1.4.1.636.1
   oit.gatech.edu. IN UFN 1.3.6.1.4.1.636.5.3

   636.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          10                   ; metric
          mordred.gatech.edu.  ; hostname
          63                   ; port nr
          "whois"              ; protocol
          "GATECH01"  )        ; auxiliary data for Whois

   636.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          20                   ; metric
          mordred.gatech.edu.  ; hostname
          80                   ; port nr
          "http"              ; protocol
          "text/urc"  )        ; auxiliary data for HTTP

   3.5.636.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          10                   ; metric
          fuzzy.oit.gatech.edu. ; hostname
          80                   ; port nr
          "http"              ; protocol
          "text/urc"  )        ; auxiliary data for HTTP
```

Let's say that the very small company Acme Inc has published some materials which they want to give a URN. They have unfortunately not registered a OID yet, so they buy a OID space from the company Publish It Inc. They register this by creating a UFN record which points from acme.com to **1.3.6.1.4.1.4711.13 which is an OID delegated from the Publish It Inc OID** (1.3.6.1.4.1.4711).

But, the company Publish It Inc is not connected to the Internet, so the registration of the different documents is done by the company Big Net. The NAPTR record therefore points to the Whois++ server at the company Big Net, and they actually runs a separate Whois++ server for the documents of Publish It Inc on port 7070.

The Big Net company in turn cooperates with the network provider Small Net. They both mirror each others databases each night to be able to let their customers use the other companies servers if their own have crashed. So,

they run secondary nameserver, secondary MX and secondary NAPTR records for each other.

The UFN and the NAPTR records therefore looks like this:

```
   acme.com. IN UFN  1.3.6.1.4.1.4711.13

   13.4711.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          10                  ; metric
          server.nic.big.net. ; hostname
          7070                ; port nr
          "whois"             ; protocol
          "BIGNET01"  )       ; auxiliary data for Whois

   13.4711.1.4.1.6.3.1.oid.urn.net. IN NAPTR (
          20                  ; metric
          nic.small.net.      ; hostname
          6212                ; port nr
          "whois"             ; protocol
          "SMALLNET01"  )     ; auxiliary data for Whois
```

Final Remarks

The point of determining a common URN syntax at the Knoxville meeting was to promote interoperability of different URN-resolving strategies. This paper presents one such strategy, and proposed implementation structures for URNs. Of course, this strategy is not limited to use by OIDs. Within DNS any name can have an NAPTR record. This facilitates use of this resolution method by any hierarchical, DNS accessible name space. We know the difference between a UFN and a real URN authority by which record is returned for a DNS query.

It is also important to realize that the use of DNS is not necesary for this scheme to work. It is possible to use protocols like Whois++ for each one of the steps as long as you can, given a something like a UFN, get a real authority, and given a real authority get one or several things like an NAPTR record. The decision to use Whois++ over DNS, or vice versa, is in the hands of the client software writer. The important step here is to provide at least _one_ URN-resolution name space management and resolution infrastructure, without precluding the evolution of others.

"The Relentless Pursuit of Perfection
   is a depth-first search of the infinitely deep
      tree of life's experiences."
                                    -- ThinkingCat

Leslie Daigle
leslie@bunyip.com
Montreal, Canada
