URNBIS
Internet-Draft
Obsoletes: 2141, 3406 (if approved)
Intended status: Standards Track
Expires: October 18, 2016

P. Saint-Andre
Filament
J. Klensin

April 16, 2016

Uniform Resource Names (URNs)
draft-ietf-urnbis-rfc2141bis-urn-16

Abstract

A Uniform Resource Name (URN) is a Uniform Resource Identifier (URI) that is assigned under the "urn" scheme and a particular URN namespace, with the intent that the URN will be either a persistent, location-independent resource identifier or in some cases an abstract designator that is persistent but that does not identify a resource. With regard to URN syntax, this document defines the canonical syntax for URNs (in a way that is consistent with URI syntax), specifies methods for determining URN equivalence, and discusses URI conformance. With regard to URN namespaces, this document specifies a method for defining a URN namespace and associating it with a namespace identifier, and describes procedures for registering namespace identifiers with the Internet Assigned Numbers Authority (IANA). This document obsoletes both RFC 2141 and RFC 3406.

Status of This Memo

This Internet-Draft will expire on October 18, 2016.

Copyright Notice

Table of Contents

1. Introduction

A Uniform Resource Name (URN) is a Uniform Resource Identifier (URI) [RFC3986] that is assigned under the "urn" scheme and a particular namespace, with the intent that the URN will be either a persistent, location-independent resource identifier or in some cases an abstract designator that is persistent but that does not identify a resource. A URN namespace is a collection of such identifiers, each of which is (1) unique, (2) assigned in a consistent and managed way, and (3) assigned according to a common definition. (Some URN namespaces

create names that exist only as URNs, whereas others create URNs out of names that already exist in other identifier systems, such as ISBNs [RFC3187] and ISSNs [RFC3044].)

The assignment of URNs is done by an organization (or, in some cases, according to an algorithm or other automated process) that has been formally delegated a namespace within the "urn" scheme (e.g., a URN in the 'example' namespace [RFC6963] might be of the form "urn:example:foo").

This document rests on two key assumptions:

1. Assignment of a URN is a managed process.

2. The space of URN namespaces is itself managed.

While other URI schemes may allow identifiers to be freely chosen and assigned, such is not the case for URNs. The syntactical correctness of a string starting with "urn:" is not sufficient to make it a URN. In order for the string to be a valid URN, the namespace identifier needs to be registered in accordance with the rules defined here and the remaining parts of the assigned-name portion of the URN needs to be generated in accordance with the rules for the registered namespace.

So that information about both URN syntax and URN namespaces is available in one place, this document does the following:

1. Defines the canonical syntax for URNs in general (in a way that is consistent with URI syntax), specifies methods for determining URN equivalence, and discusses URI conformance.

2. Specifies a method for defining a URN namespace and associating it with a namespace identifier, and describes procedures for registering namespace identifiers with the Internet Assigned Numbers Authority (IANA).

For URN syntax and URN namespaces, this document modernizes and replaces the definitions from the original URN syntax [RFC2141] and namespace definition and registration [RFC3406] specifications. These modifications build on the key requirements provided in the original functional description for URN [RFC1737] and many years of experience. In both those original documents and the present one, the intent is to define URNs in a consistent manner so that, wherever practical, the parsing, handling, and resolution of URNs can be independent of the namespace within which a given URN is assigned.

Together with input from several key user communities, the history and experiences dictated expansion of the URN definition to support new functionality, including the use of syntax explicitly reserved for future standardization in RFC 2141. All namespaces and URNs that were valid under the earlier specifications remain valid even though it may be useful to update some of them to take advantage of new features.

Summaries of changes from RFC 2141 and RFC 3406 appear in Appendix B and Appendix C respectively. This document obsoletes both [RFC2141] and [RFC3406]. While it does not explicitly update or replace [RFC1737] or [RFC2276] the reader who references those documents should be aware that the conceptual model of URNs in this document is slightly different from those older specifications.

1.1. Specificity and this Standard

To a degree much greater than when URNs were first considered and their uses outlined (Cf. [RFC1737]) issues of persistent identifiers on the Internet involve fundamental design tradeoffs and research questions that are much broader that URNs or the URN approach. Ideal and comprehensive specifications about what should be done or required across the entire range of URNs would require general agreement about those issues and their resolution. While some of them were introduced by the Internet or computer-age approaches to character encodings and data abstraction, others predate the Internet and computer systems by centuries; there is unlikely to be agreement about comprehensive solutions in the near future.

Among these general issues, one that is specific to URNs is the fairly abstract topic of "resolution", discussed in Section 1.2, Section 2.3.2, and elsewhere below. While it is possible to define the relationships quite precisely for a URN that resolves to a URL that, in turn, resolves (or locates) to a single target document or similar resource, that is only one special case albeit an important one. URNs (either individually or entire namespaces as defined below) that do not resolve to URLs at all or that resolve to metadata or non-Internet objects are among URN use cases explicitly permitted by this specification; each leaves the concept of "resolution" somewhat more abstract and difficult than the simple case of resolution to a URL.

A similar set of issues arises for character sets and encodings. URNs, especially URNs that will be used as user-facing identifiers, should be convenient to use in local languages and writing systems, easily specified with a wide range of keyboards and local conventions, and unambiguous. There are tradeoffs among those goals and it is impossible at present to see how a simple and readily-

understandable set of rules could be developed that would be optimal, or even reasonable, for all URNs. The discussion in Section 2.2 defines an overall framework that should make generalized parsing and processing possible, but also makes recommendations about rules for individual namespaces.

This specification consequently contains some requirements and flexibility that would not be present in a more perfect world but that are necessary in order to allow producing any consensus specification at all rather than just giving up on URNs going forward.

1.2. Terminology

This document uses the terms "resolution" and "resolver" in roughly the sense in which they were used in the original discussion of architectural principles for URNs [RFC2276], i.e., "resolution" is the act of supplying services related to the identified resource, such as translating the persistent name into one or more current locators for the resource, delivering metadata about the resource in an appropriate format, or even delivering a document object from a convenient source without requiring further intermediaries. At the time of this writing, resolution services are described in [RFC2483]. In order to underline the difference between the names and locators, this document uses the term Uniform Resource Locator (URL), rather than the generic term Uniform Resource Identifier (URI), to refer to locators; see also Section 1.1.3 of [RFC3986].

If there are or will be resolution services available for a URN, this document calls the URN a "resource identifier" in roughly the sense that term is used in [RFC3986]. If there is no intention to provide any resolution services, and the distinction is important, this document calls the URN an "abstract designator".

Several other important terms used in this document, including some "normalization" operations that are not part of the Unicode Standard [UNICODE], are defined in the URI specification [RFC3986].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. URN Syntax

As discussed above, the syntax for URNs in this specification allows significantly more functionality than was the case in the earlier specifications. It is also harmonized with the general URI syntax

[RFC3986]. That syntax definition was completed after the earlier URN specifications.

However, this specification does not extend the URN syntax to allow direct use of characters outside the ASCII range [RFC20]. That prohibition implies that any such characters need to be percent-encoded as described in Section 2.1 of the URI specification [RFC3986].

The basic syntax for a URN is defined using the Augmented Backus-Naur Form (ABNF) as specified in [RFC5234]. Rules not defined here (specifically: alphanum, fragment, and pchar) are defined as part of the URI syntax [RFC3986] and used here to point out the syntactic relationship with the terms used there. The definitions of some of the terms used below are not complete; additional restrictions are imposed sections of the document that are specific to those terms.

```
namestring    = assigned-name
                [ rq-components ]
                [ "#" f-component ]
assigned-name = "urn" ":" NID ":" NSS
NID           = (alphanum) 0*30(ldh) (alphanum)
ldh           = alphanum / "-"
NSS           = pchar *(pchar / "/")
rq-components =  ( "?="  q-component
                     [ "?+" r-component ] ) /
                 ( "?+" r-component
                     [ "?="  q-component ] )
q-component   = pchar *( pchar / "/" / "?" )
r-component   = pchar *( pchar / "/" / "?" )
f-component   = fragment
```

The question mark character "?" can be used without percent-encoding inside q-components, r-components, and f-components. Other than inside those components a "?" that is not immediately followed by "=" or "+" is not defined for URNs and SHOULD be treated as a syntax error by URN-specific parsers and other processors.

The following sections provide additional information about the syntactic elements of URNs.

2.1. Namespace Identifier (NID)

Namespace Identifiers (NIDs) are case insensitive (e.g., "ISBN" and "isbn" are equivalent).

Characters outside the ASCII range [RFC20] are not permitted in NIDs, and no encoding mechanism for such characters is supported.

Section 5.1 and Section 5.2 impose additional constraints on the strings that can be used as NIDs, i.e., the syntax shown above is not comprehensive.

2.2. Namespace Specific String (NSS)

The namespace specific string (NSS) is a unique identifier within a namespace that is assigned and managed in a consistent way and that conforms to the definition of the relevant namespace. The combination of the NID (unique across the entire "urn" scheme) and the NSS (unique within the namespace) ensures that the resulting URN is a globally unique URI.

The NSS specified in this document allows characters not permitted by earlier specifications (see Appendix B. In particular, the "/" character, which is now allowed, effectively makes it possible to encapsulate hierarchical identifiers from other naming systems. For instance, consider the hypothetical example of a hierarchical naming system in which the identifiers take the form of a sequence of numbers separated by the "/" character, such as "1/406/47452/2". If the naming authority for such identifiers were to use URNs, it would be natural to place the existing identifiers in the NSS, resulting in URNs such as "urn:example:1/406/47452/2".

The changes to the syntax for the NSS do not modify the encoding rules for URN namespaces that were defined in accordance with [RFC2141]. If any such URN namespace that is used outside of the URN context (i.e., as a standalone, non-embedded, identifier space) also allows the use of "/", "~", or "&" in the native form within that identifier space, then the encoding rules for that namespace are not changed by this specification.

Depending on the rules governing a namespace, strings that are valid in an NSS associated with that namespace might contain characters that are not allowed by the "pchar" production referenced above (e.g., characters outside the ASCII range or, consistent with the restrictions in RFC 3986, the characters "/", "?", "#", "[", and "]"). While such a string might be a valid name, it is not a valid URN until it has been translated into a conformant NSS. In the case of URNs that are formed from names that exist separately in a standalone identifier space, translation of an identifier from its "native" format to URN format is accomplished by using the canonicalization and encoding methods defined for that URN namespace. Software that is not aware of those namespace-specific canonicalization and encoding rules MUST NOT construct URNs from the names in the standalone identifier space.

In particular, with regard to characters outside the ASCII range, URNs that appear in protocols or that are passed between systems MUST use only Unicode characters encoded in UTF-8 and further encoded as required by RFC 3986. To the extent feasible consistent with the requirements of identifiers defined and standardized elsewhere and the principles discussed in Section 1.1, strings SHOULD be restricted to either ASCII letters and digits or to the characters and syntax of some widely-used identifier model such as those of IDNA [RFC5890], PRECIS [RFC7613], or the Unicode Identifier and Pattern Syntax spec [UAX31].

In order to make URNs as stable and persistent as possible when protocols evolve and the environment around them changes, namespaces SHOULD NOT allow characters outside the basic Latin repertoire [RFC20] unless the nature of the particular namespace makes such characters necessary.

2.3. Optional Components

This specification includes three optional components in the URN syntax. They are known as q-component, r-component, and f-component and are described in more detail below. Because this specification focuses almost exclusively on URN syntax, it does not define detailed semantics of these components for URNs in general. However, each of these components has a distinct role that is independent of the URN and its namespace. It is intended that clients will be able to handle these components uniformly for all URNs. These components MAY be used with URNs from existing namespaces, whether or not a namespace explicitly supports them. However, consistent with the approach taken in RFC 3986, the behavior of a URN that contains components that are undefined or meaningless for a particular namespace or resource is not defined. The following sections describe these optional components and their interpretation in greater detail.

2.3.1. q-component

The q-component is intended for passing parameters to either the named resource or a system that can supply the requested service, for interpretation by that resource or system. (By contrast, passing parameters to URN resolution services is handled by r-components as described in the next section.)

The URN q-component has the same syntax as the URI query component, but is introduced by "?=", not the "?" alone. If a URN resolves to a URL, the q-component from the URN is copied verbatim to the query component of the URL. If the URN does not resolve to a URL (i.e., is an abstract designator or resolves directly to an object or a non-URL

resource designator), the interpretation of the q-component is undefined by this specification. Thus, for URNs which may be resolved to a URL, the semantics of q-component are identical to those for queries to the resource located via that URL.

For the sake of consistency with RFC 3986, neither the general syntax nor the semantics of q-components are defined by, or dependent on, the namespace of the URN. In parallel with RFC 3896, specifics of syntax and semantics, e.g., which keywords or terms are meaningful, of course may depend on a particular namespace or even a particular resource.

The sequence "?=" begins the q-component. The q-component ends with a "?+" sequence (which begins an r-component) or a "#" character (number sign, which begins an f-component). If neither of those appear, the q-component continues to the end of the URN. The characters slash ("/") and question mark ("?") may represent data within the q-component. Note that characters outside the ASCII range [RFC20] MUST be percent-encoded using the method defined in Section 2.1 of the generic URI specification [RFC3986].

As described in Section 3, the q-component SHALL NOT be taken into account when determining URN equivalence.

Namespaces and associated information placement in syntax SHOULD be designed to avoid any need for a resolution service to consider the q-component. Namespace-specific and more generic resolution systems MUST NOT require that q-component information be passed to them for processing.

Consider the hypothetical example of passing parameters to an application that returns weather reports from different regions or for different time periods. This could perhaps be accomplished by specifying latitude and longitude coordinates and datetimes in the URN's q-component, resulting in URNs such as the following.

```
   urn:example:weather?=op=map&lat=39.56
      &lon=-104.85&datetime=1969-07-21T02:56:15Z
```

2.3.2. r-component

The r-component is intended for passing parameters to URN resolution services (taken broadly, see Section 1.1) and interpreted by those services. (By contrast, passing parameters to the resources identified by a URN, or to applications that manage such resources, is handled by q-components as described in the previous section.)

The URN r-component has no syntactic equivalent in URLs.

The sequence "?+" begins the r-component. The r-component ends with a "?=" sequence (which begins a q-component) or a "#" character (number sign, which begins an f-component). If neither of those appear, the r-component continues to the end of the URN. Note that characters outside the ASCII range [RFC20] MUST be percent-encoded using the method defined in Section 2.1 of the generic URI specification [RFC3986].

As described under Section 3, the r-component SHALL NOT be taken into account when determining URN equivalence. However, the r-component SHALL be supplied along with the URN when presenting a request to a URN resolution service.

This document defines only the syntax of the r-component and reserves it for future use. The exact semantics of the r-component and its use in URN resolution protocols are a matter for potential standardization in separate specifications, presumably including specifications that define conventions and a registry for resolution service identifiers.

Consider the hypothetical example of passing parameters to a resolution service (say, an ISO alpha-2 country code [ISO3166-1] in order to scope down the preferred country in which to search for a physical copy of a book). This could perhaps be accomplished by specifying the country code in the r-component, resulting in URNs such as:

```
urn:example:foo-bar-baz-qux?+CCResolve: cc=uk
```

While the above should serve as a general explanation and illustration of the intent for r-components, there are many unresolved issues with them, including their relationship to resolution mechanisms associated with the particular NID and namespace at registration time. Thus r-components SHOULD NOT be used for actual URNs until additional development and standardization work is complete, including specification of any necessary registration mechanisms.

2.3.3. f-component

The f-component is intended to be interpreted by the client as a specification for a location within, or region of, the named resource.

The URN f-component has the same syntax as the URI fragment component. If a URN containing an f-component resolves to a single

URL associated with the named resource, the f-component from the URN can be applied (usually by the client) verbatim as the fragment of that URL. If the URN does not resolve to a URL (e.g., is an abstract designator), the interpretation of the f-component is undefined by this specification. Thus, for URNs which may be resolved to a URL, the semantics of f-components are identical to those of fragments for that resource.

For the sake of consistency with RFC 3986, neither the general syntax nor the semantics of f-components are defined by, or dependent on, the namespace of the URN. In parallel with RFC 3896, specifics of syntax and semantics, e.g., which keywords or terms are meaningful, of course may depend on a particular namespace or even a particular resource.

The f-component is indicated by the presence of a number sign ("#") character and terminated by the end of the URI. Any characters outside the ASCII range [RFC20] that appear in the f-component MUST be percent-encoded using the method defined in Section 2.1 of the generic URI specification [RFC3986].

As described under Section 3, the f-component SHALL NOT be taken into account when determining URN equivalence.

Clients SHOULD NOT pass f-components to resolution services unless those services also perform object retrieval and interpretation functions.

The f-component is primarily intended to distinguish the constituent parts of resources named by URNs. Thus, for URNs that resolve to URLs of the named resources, the semantics of an f-component are defined by the media type of those resources, not by the namespace.

Consider the hypothetical example of obtaining resources that are part of a larger entity (say, the chapters of a book). Each part could be specified in the f-component, resulting in URNs such as:

```
urn:example:foo-bar-baz-qux#somepart
```

3. Equivalence of URNs

3.1. Procedure

For various purposes such as caching, it is often desirable to determine if two URNs are "the same". This is done by testing for equivalence (see Section 6.1 of [RFC3986]).

The generic URI specification [RFC3986] is very flexible about equality comparisons, putting the focus on allowing false negatives and avoiding false positives. If comparisons are made in a scheme-independent way, i.e., as URI comparisons only, URNs that this specification considers equal would be rejected. The discussion below applies when the URIs involved are known to be URNs.

Two URNs are equivalent if their <assigned-name> portions are octet-by-octet equal after applying case normalization (as specified in Section 6.2.2.1 of [RFC3986]) to the following constructs:

1. the URI scheme "urn", by conversion to lower case

2. the NID, by conversion to lower case

3. any percent-encoded characters in the NSS (that is, all character triplets that match the <pct-encoding> production found in Section 2.1 of the base URI specification [RFC3986]), by conversion to upper case for the digits A-F.

Percent-encoded characters MUST NOT be decoded, i.e., percent-encoding normalization (as specified in Section 6.2.2.2 of [RFC3986]) MUST NOT be applied as part of the comparison process.

If a q-component, r-component, or f-component (or any combination thereof) are included in a URN, they MUST be ignored for purposes of determining equivalence.

URN namespace definitions MAY include additional rules for equivalence, such as case-insensitivity of the NSS (or parts thereof). Such rules MUST always have the effect of eliminating some of the false negatives obtained by the procedure above and MUST NOT result in treating two URNs as not equivalent if the procedure here says they are equivalent. For related considerations with regard to NID registration, see below.

3.2. Examples

This section shows a variety of URNs (using the "example" NID defined in [RFC6963]) that highlight the equivalence rules.

First, because the scheme and NID are case-insensitive, the following URNs are equivalent to each other:

- urn:example:a123,z456

- URN:example:a123,z456

- urn:EXAMPLE:a123,z456

Second, because the q-component and f-component are not taken into account for purposes of testing equivalence, the following URNs are equivalent to the first three examples above:

- urn:example:a123,z456?=abc

- urn:example:a123,z456#789

- urn:example:a123,z456#abc

Third, because the "/" character (and anything that follows it) in the NSS is taken into account for purposes of equivalence, the following URNs are not equivalent to each other or to the preceding URNs:

- urn:example:a123,z456/foo

- urn:example:a123,z456/bar

- urn:example:a123,z456/baz

Fourth, because of percent-encoding, the following URNs are equivalent only to each other (although %2C is the percent-encoded transformation of "," from the previous examples, such sequences are not decoded for purposes of testing equivalence):

- urn:example:a123%2Cz456

- URN:EXAMPLE:a123%2cz456

Fifth, because characters other than percent-encoded sequences in the NSS are treated in a case-sensitive manner (unless otherwise specified for the namespace in question), the following URNs are not equivalent to the first three URNs:

- urn:example:A123,z456

- urn:example:a123,Z456

Sixth, on casual visual inspection of a URN presented in a human-oriented interface the following URN might appear the same as the first three URNs (because U+0430 CYRILLIC SMALL LETTER A can be confused with U+0061 LATIN SMALL LETTER A), but it is not equivalent:

- urn:example:%D0%B0123,z456

4. URI Conformance

4.1. Use in URI Protocol Slots

Because a URN is, syntactically, a URI under the "urn" scheme, in theory a URN can be placed in any protocol slot that allows for a URI (e.g., the 'href' and 'src' attributes in HTML, the <base/> element in HTML, the 'xml:base' attribute in XML [XML-BASE], and the 'xmlns' attribute in XML for XML namespace names [XML-NAMES]).

However, this does not imply that, semantically, it always makes sense in practice to place a URN in a given URI protocol slot; in particular, because a URN might not specify the location of a resource or even point indirectly to one, it might not be appropriate to place a URN in a URI protocol slot that points to a resource (e.g., the aforementioned 'href' and 'src' attributes).

Ultimately, guidelines regarding when it is appropriate to use URIs under the "urn" scheme (or any other scheme) are the responsibility of specifications for individual URI protocol slots (e.g., the specification for the 'xml:base' attribute in XML might recommend that it is inappropriate to use URNs in that protocol slot). This specification cannot possibly anticipate all of the relevant cases, and it is not the place of this specification to require or restrict usage for individual protocol slots.

4.2. Parsing

In part because of the separation of URN semantics from more general URI syntax [I-D.ietf-urnbis-semantics-clarif], generic URI processors need to pay special attention to the parsing and analysis rules of RFC 3986 and, in particular, must treat the URI as opaque unless the scheme and its requirements are recognized. In the latter case, such processors may be in a position to invoke scheme-appropriate processing such as by a URN resolver. The URN resolver can either be an external resolver that the URI resolver knows of, or it can be functionality built into the URI resolver. Note that this requirement might impose constraints on the contexts in which URNs are appropriately used; see Section 4.1.

4.3. URNs and Relative References

Section 5.2 of [RFC3986] describes an algorithm for converting a URI reference that might be relative to a given base URI into "parsed components" of the target of that reference, which can then be recomposed per RFC 3986 Section 5.3 into a target URI. This algorithm cannot be applied directly to URNs because their syntax does not support the necessary path components. Whenever a URN

resolves to a URL which may be used to access the resource, there is a more specific interpretation of q-component and f-component: the q-component is copied verbatim to the query portion of the URL (if that URL scheme supports query), and the f-component is copied verbatim to the fragment portion of the URL. Even though the notion of a URN as a "persistent", "permanent" identifier does not reconcile easily with relative referencing, resources named with URNs may contain relative references that do not apply to the URN itself.

Given the foregoing, a relative reference SHOULD NOT be evaluated directly with respect to a URN. Instead, a relative reference SHOULD be evaluated indirectly with respect to one of the following:

1. a base URI (other than a URN) declared by the resource itself; or

2. a base URI (other than a URN) obtained through the URN resolution process; or

3. the URL of the resource as obtained through the URN resolution process

(Case 2 permits the resolution process to explicitly supply a base URI if the resource content is supplied directly by the resolution service rather than via an intermediate "location" URI.)

If no such base URI exists, use of a relative reference with respect to a URN is an error. Client behavior in this case is undefined.

Resolution services SHOULD ensure that a base URI is supplied any time they provide resource content directly to a client.

4.4. Transport and Display

When URNs are transported and exchanged, they MUST be represented in the format defined herein. Further, all URN-aware applications MUST offer the option of displaying URNs in this canonical form to allow for direct transcription (for example by cut-and-paste techniques). Such applications might support display of URNs in a more human-friendly form and might use a character set that includes characters that are not permitted in URN syntax as defined in this specification (e.g., when displaying URNs to humans, such applications might replace percent-encoded strings with characters from an extended character repertoire such as Unicode [UNICODE]).

To minimize user confusion, a URI browser SHOULD display the complete URN (including the "urn" scheme and any components) to ensure that there is no confusion between URN namespace identifiers and URI scheme identifiers. For example, a URI beginning with "urn:xmpp:"

[RFC4854] is very different from a URI beginning with "xmpp:" [RFC5122]. Similarly, a potential DOI URI scheme [DOI-URI] is different from, and possibly completely unrelated to, a possible DOI URN namespace.

4.5. URI Design and Ownership

As mentioned, the assignment of URNs is a managed process, as is the assignment of namespaces themselves. Although design of the URNs to be assigned within a given namespace is ceded by this specification to the namespace owner, doing so in a managed way avoids the problems inherent in unmanaged generation of URIs as described in the recommendations regarding URI design and ownership [RFC7320].

5. URN Namespaces

A URN namespace is a collection of identifiers that obey three constraints: each identifier is (1) unique, (2) assigned in a consistent way, and (3) assigned according to a common definition.

1. The "uniqueness" constraint means that an identifier within the namespace is never assigned to more than one resource and never reassigned to a different resource (for the kind of "resource" identified by URNs assigned within the namespace). This holds true even if the identifier itself is deprecated or becomes obsolete.

2. The "consistent assignment" constraint means that an identifier within the namespace is assigned by an organization or created in accordance with a process or algorithm that is always followed.

3. The "common definition" constraint means that there are clear definitions for the syntax of identifiers within the namespace and for the process of assigning or creating them.

A URN namespace is identified by a particular NID in order to ensure the global uniqueness of URNs and, optionally, to provide a cue regarding the structure of URNs assigned within a namespace.

With regard to global uniqueness, using different NIDs for different collections of identifiers ensures that no two URNs will be the same for different resources, since each collection is required to uniquely assign each identifier. However, a single resource MAY have more than one URN assigned to it, either in the same namespace (if the namespace permits it) or in different namespaces, and either for similar purposes or different purposes. (For example, if a publisher assigns an ISBN to an electronic publication and that publication is later incorporated into a digital long term archive operated by a

national library, the library might assign the publication a NBN, resulting in two URNs referring to the same book.) Subject to other constraints, such as those imposed by the URI syntax [RFC3986], the rules of the URN scheme are intended to allow preserving the normal and natural form of identifiers specified elsewhere when they are treated as URN namespaces.

With regard to the structure of URNs assigned within a namespace, the development of an identifier structure (and thereby a collection of identifiers) depends on the requirements of the community defining the identifiers, how the identifiers will be assigned and used, etc. These issues are beyond the scope of URN syntax and the general rules for URN namespaces, because they are specific to the community defining a namespace (e.g., the bibliographic and publishing communities in the case of the 'ISBN' and 'ISSN' namespaces, or the developers of extensions to the Extensible Messaging and Presence Protocol in the case of the 'XMPP' namespace).

URN namespaces inherit certain rights and responsibilities by the nature of URNs, e.g.:

1. They uphold the general principles of a well-managed URN namespace by providing persistent identification of resources and unique assignment of identifier strings in accordance with a common definition.

2. Optionally, they can be registered in global registration services such as those described in [RFC2483].

There are two types of URN namespace: formal and informal. These are distinguished by the expected level of service, the information needed to define the namespace, and the procedures for registration. Because the majority of the namespaces registered so far have been formal, this document concentrates on formal namespaces.

5.1. Formal Namespaces

A formal namespace provides benefit to some subset of users on the Internet. In particular, it would not make sense for a formal namespace to be used only by a community or network that is not connected to the Internet. For example, it would be inappropriate for a NID to effectively force someone to use a proprietary network or service not open to the general Internet user. The intent is that, while the community of those who might actively use the names assigned within that NID might be small, the potential use of identifiers within that NID is open to any user on the Internet. Formal NIDs might be appropriate even when some aspects are not fully open. For example, a namespace might make use of a fee-based,

privately managed, or proprietary registry for assignment of URNs in the namespace. However, it might still benefit some Internet users if the associated services have openly-published identifiers.

An organization that will assign URNs within a formal namespace SHOULD meet the following criteria:

1. Organizational stability and the ability to maintain the URN namespace for a long time; absent such evidence, it ought to be clear how the namespace can remain viable if the organization can no longer maintain the namespace.

2. Competency in name assignment. This will improve the likelihood of persistence (e.g. to minimize the likelihood of conflicts).

3. Commitment to not reassigning existing names and to allowing old names to continue to be valid (e.g., if the assignee of a name is no longer a member or customer of the assigning organization, if various information about the assignee or named entity happens to change, or even if the assignee or the named entity itself is no longer in existence; in all these cases, the name is still valid).

A formal namespace establishes a particular NID, subject to the following constraints (above and beyond the syntax rules already specified):

1. It MUST NOT be an already-registered NID.

2. It MUST NOT start with "urn-" (which is reserved for informal namespaces).

3. It MUST be more than two characters long.

4. It MUST NOT start with ALPHA ALPHA "-", i.e., any string consisting of two letters followed by one hyphen.

5. It MUST NOT start with the string "xn--" or any other string consisting of two letters followed by two hyphens. Such strings are reserved for potential representation of DNS A-labels and similar strings in the future [RFC5890].

6. It MUST NOT start with the string "X-" so that it will not be confused with or conflict any experimental namespace previously permitted by [RFC3406].

All two-letter strings, and all two-letter strings followed by "-" and any sequence of valid NID characters, are reserved for potential

use as NIDs based on ISO alpha-2 country codes [ISO3166-1] for eventual national registrations of URN namespaces. The definition and scoping of rules for allocation of responsibility for such country-code-based namespaces is beyond the scope of this document.

Applicants and reviewers considering new NIDs should also be aware that they may be considered as names with semantic implications and hence a source of conflict. Particular attention should be paid to strings that might be construed as names of, or registered under the authority of, countries (including ISO 3166-1 alpha-3 codes) and to strings that might imply association with existing URI schemes, identifier systems, or trademarks. However, in line with traditional policies, disputes about "ownership" of particular strings are disagreements among the parties involved; neither IANA nor the IETF will become involved in such disputes except in response to orders from a court of competent jurisdiction.

5.2. Informal Namespaces

Informal namespaces are full-fledged URN namespaces, with all the associated rights and responsibilities. Informal namespaces differ from formal namespaces in the process for assigning a NID: for an informal namespace, the registrant does not designate the NID; instead, IANA assigns a NID consisting of the string 'urn-' followed by one or more digits (e.g., "urn-7") where the digits consist of the next available number in the sequence of positive integers assigned to informal namespaces. Thus the syntax of an informal namespace is:

```
    InformalNamespaceName = "urn-" Number
    Number                = DigitNonZero 0*Digit
    DigitNonZero          = "1"/ "2" / "3" / "4"/ "5"
                          / "6" / "7" / "8" / "9"
    Digit                 = "0" / DigitNonZero
```

The only restrictions on <Number> are that it (1) consist strictly of ASCII digits, that it (2) not have leading zeros, and that it (3) not cause the NID to exceed the length limitations defined for the URN syntax.

6. Defining and Registering a URN Namespace

6.1. Overview

Because the space of URN namespaces is itself managed, the definition of a namespace SHOULD pay particular attention to:

1. The purpose of the namespace.

2. The syntax of URNs assigned within the namespace, including the internal syntax and anticipated effects of q-components or r-components. (The syntax and interpretation of f-components are defined in RFC 3986.)

3. The process for assigning URNs within the namespace.

4. The security implications of assigning URNs within the namespace and using the assigned URNs.

5. Any potential interoperability issues with URNs assigned within the namespace.

6. Optionally, the process for resolving URNs issued within the namespace.

The section on completing the template (Section 6.4) explains these matters in greater detail. Although the registration templates are the same in all cases, slightly different procedures are used depending on the source of the registration.

6.2. Registration Policy and Process: Community Registrations

The basic registration policy for URN namespaces is Expert Review as defined in the "IANA Considerations" document [RFC5226]. For namespaces or their definitions that are intended to become standards or normative components of standards, the output of the Expert Review process is intended to be a report, rather than instructions to IANA to take action (see below). The key steps are:

1. Fill out the namespace registration template (see Section 6.4 and Appendix A). This can be done as part of an Internet-Draft or a specification in another series, although that is not necessary.

2. Send the completed template to the urn@ietf.org discussion list for review.

3. If necessary to address comments received, repeat steps 1 and 2.

4. If the designated experts approve the request and no standardization action is involved, the IANA will register the requested NID. If standardization is anticipated, the designated experts will prepare a report and forward it to the appropriate standards approval body (the IESG in the case of the IETF); IANA will register the requested NID only after receiving directions from that body and a copy of the expert review report.

A namespace registration can be revised by updating the registration template, following the same steps outlined above for new registrations. A revised registration MUST describe differences from prior versions and SHOULD make special note of any relevant changes in the underlying technologies or namespace management processes.

Experience to date with namespace registration requests has shown that registrants sometimes do not initially understand some of the subtleties of URN namespaces, and that defining the namespace in the form of a specification enables the registrants to clearly formulate their "contract" with the intended user community. Therefore, although the registration policy for formal namespaces is Expert Review and a specification is not strictly required, registrants SHOULD provide a stable specification documenting the namespace definition and expanding upon the issues described herein.

Because naming can be difficult and contentious, namespace registrants and the designated experts are strongly encouraged to work together in a spirit of good faith and mutual understanding to achieve rough consensus (see [RFC7282]) on handling registration requests. They are also encouraged to bring additional expertise into the discussion if that would be helpful in providing perspective or otherwise resolving issues.

Especially when iterations in the registration process are prolonged, designated experts are expected to take reasonable precautions to avoid "race conditions" on proposed NID names and, if such situations arise, to encourage applicants to work out any conflicts among themselves.

6.3. Registration Policy and Process: Fast Track for Standards Development Organizations, Scientific Societies, and Similar Bodies

The IETF recognizes that situations will arise in which URN namespaces will be created to either embed existing and established standards, particularly identifier standards, or to reflect knowledge, terminology, or methods of organizing information that lie well outside the IETF's scope or the likely subject matter knowledge of its Designated Experts. In situations in which the registration request originates from, or is authorized by, a recognized standards-related organization, scientific society, or similar body, a somewhat different procedure is available at the option of that body:

1. The namespace registration template is filled out and submitted as in steps 1 and 2 above.

2. A specification is required that reflects or points to the needed external standards or specifications. Publication in the RFC Series or through an IETF process (e.g., posting as an Internet Draft) is not expected and would be appropriate only under very unusual circumstances.

3. The reviews on the discussion list and by the designated experts are strictly advisory, with the decisions about what advice to accept and the length of time to allocate to the process strictly under the control of the external body.

4. When that body concludes that the application is sufficiently mature, its represenative(s) will request that IANA complete the registration for the NID, and IANA will do so.

Decisions about whether to recognize the requesting entity as a standards-related organization, scientific society, or similar body are the responsibility of the IESG.

A model similar to this has already been defined for recognized standards-related organizations that wish to register Media Types. The document describing that mechanism [RFC6838] provides somewhat more information about the general approach.

6.4. Completing the Template

A template for defining and registering a URN namespace is provided in Appendix A. This section describes considerations for completing the template.

6.4.1. Purpose

The "Purpose" section of the template describes matters such as:

1. The kinds of resources identified by URNs assigned within the namespace.

2. The scope and applicability of the URNs assigned within the namespace; this might include information about the community of use (e.g., a particular nation, industry, technology, or organization), whether the assigned URNs will be used on public networks or private networks, etc.

3. How the intended community (and the Internet community at large) will benefit from using or resolving the assigned URNs.

4. How the namespace relates to and complements existing URN namespaces, URI schemes, and identifier systems.

5. The kinds of software applications that can use or resolve the assigned URNs (e.g., by differentiating among disparate namespaces, identifying resources in a persistent fashion, or meaningfully resolving and accessing services associated with the namespace).

6. Whether resolution services are available or will be available (and, if so, the nature or identity of the services). Examples of q-component and, when they are standardized, r-component, semantics and syntax are helpful here, even if detailed definitions are provided elsewhere later.

7. Whether the namespace or its definition is expected to become an integral or normative element of a standard being developed in the IETF or some other recognized standards body.

6.4.2. Syntax

The "Syntax" section of the template contains:

1. A description of the structure of URNs within the namespace, in conformance with the fundamental URN syntax. The structure might be described in terms of a formal definition (e.g., using Augmented BNF for Syntax Specifications (ABNF) as specified in [RFC5234]), an algorithm for generating conformant URNs, or a regular expression for parsing the identifier into components; alternatively, the structure might be opaque.

2. Any special character encoding rules for assigned URNs (e.g., which character ought to always be used for quotes).

3. Rules for determining equivalence between two identifiers in the namespace. Such rules ought to always have the effect of eliminating false negatives that might otherwise result from comparison. If it is appropriate and helpful, reference can be made to specific equivalence rules defined in the URI specification [RFC3986]. Examples of equivalence rules include equivalence between uppercase and lowercase characters in the Namespace Specific String, between hyphenated and non-hyphenated groupings in the identifier string, or between single-quotes and double-quotes. There may also be namespace-specific special encoding considerations, especially for URNs that contain embedded forms of other types of identifiers. (Note that these are not normative statements for any kind of best practice related to handling of equivalences between characters in general; such statements are limited to one particular namespace only.)

4. Any special considerations necessary for conforming with the URN syntax. This is particularly applicable in the case of existing naming systems that are used in the context of URNs. For example, if a namespace is used in contexts other than URNs, it might make use of characters that are reserved in the URN syntax. This section ought to note any such characters, and outline necessary mappings to conform to URN syntax. Normally, this will be handled by percent-encoding the character as specified in Section 2.1 of the URI specification [RFC3986].

5. Any special considerations for the meaning of q-components (e.g., keywords) or f-components (e.g., predefined terms) in the context of this namespace.

6.4.3. Assignment

The "Assignment" section of the template describes matters such as:

1. Mechanisms or authorities for assigning URNs to resources. It ought to make clear whether assignment is completely open (e.g., following a particular procedure such as first-come, first-served (FCFS)), completely closed (e.g., for a private organization), or limited in various ways (e.g., delegated to authorities recognized by a particular organization); if limited, it ought to explain how to become an assigner of identifiers or how to request assignment of identifiers from existing assignment authorities.

2. Methods for ensuring that URNs within the namespace are unique. For example, identifiers might be assigned sequentially or in accordance with some well-defined process by a single authority, assignment might be partitioned among delegated authorities that are individually responsible for respecting uniqueness rules, or URNs might be created independently following an algorithm that itself guarantees uniqueness.

6.4.4. Security and Privacy

The "Security and Privacy" section of the template describes any potential issues related to security and privacy with regard to assignment, use, and resolution of identifiers within the namespace. Examples of such issues include:

- The consequences of producing false negatives and false positives during comparison for equivalence (see "Issues in Identifier Comparison for Security Purposes" [RFC6943])

- Leakage of private information when identifiers are communicated on the public Internet

- The potential for directory harvesting

- Various issues discussed in the guidelines for security considerations in RFCs [RFC3552] and the privacy considerations for Internet protocols [RFC6973].

6.4.5. Interoperability

The "Interoperability" section MUST specify any known potential issues related to interoperability. Examples include possible confusion with other URN namespaces or naming systems because of syntax (e.g., percent-encoding of certain characters) or scope (e.g., overlapping areas of interest). If at all possible, concerns that arise during the registration of a URN namespace (e.g., due to the syntax or scope of an identifier system) should be resolved as part of or in parallel to the registration process.

6.4.6. Resolution

The "Resolution" section MUST specify whether resolution mechanisms are intended or anticipated for URNs assigned within the namespace (e.g., URNs within some namespaces are intended to act as abstract designators and thus are not intended to be resolved).

If resolution is intended, then this section SHOULD specify whether the organization that assigns URNs within the namespace intends to operate or recommend any resolution services for URNs within that namespace. In addition, if the assigning organization intends to implement registration for publicly advertised resolution services (for example using a system based on principles similar to those described in [RFC2276] and [RFC2483]), then this section SHOULD list or reference the requirements for being publicly advertised by the assigning organization. In addition, this section SHOULD describe any special considerations for the handilng of r-components in the context of this namespace.

6.4.7. Additional Information

Please include any additional information that would be useful to those trying to understand this registration or its relationship to others, such as comparisons to existing namespaces that might seem to overlap.

This section of the template is optional.

7. IANA Considerations

7.1. URI Scheme

This section updates the registration of the 'urn' URI scheme in the Permanent URI Registry [URI-Registry] .

[Note to RFC Editor: please replace "[this document]" with "RFC" and the number assigned to this document upon publication.]

URI Scheme Name: urn

Status: permanent

URI Scheme Syntax: See Section 2 of [this document].

URI Scheme Semantics: The 'urn' scheme identifies Uniform Resource Names, which are persistent, location-independent resource identifiers.

Encoding Considerations: See Section 2 of [this document].

Applications/Protocols That Use This URI Scheme Name: Uniform Resource Names are used in a wide variety of applications, including bibliographic reference systems and as names for Extensible Markup Language (XML) namespaces.

Interoperability Considerations: See Section 4 of [this document].

Security Considerations: See Section 6.4.4 and Section 8 of [this document].

Contact: URNBIS WG [mailto:urn@ietf.org]

Author/Change Controller: This scheme is registered under the IETF tree. As such, the IETF maintains change control.

References None.

7.2. Registration of URN Namespaces

This document outlines the processes for registering URN namespaces, and has implications for the IANA in terms of registries to be maintained (see especially Section 6). In all cases, the IANA ought to assign the appropriate NID (formal or informal) once the procedures outlined in this document have been completed.

8. Security and Privacy Considerations

The definition of a URN namespace needs to account for potential security and privacy issues related to assignment, use, and resolution of identifiers within the namespace (e.g., some namespace resolvers might assign special meaning to certain characters in the Namespace Specific String); see Section 6.4.4 for further discussion.

In most cases, URN namespaces provide a way to declare public information. Normally, these declarations will have a relatively low security profile, however there is always the danger of "spoofing" and providing misinformation. Information in these declarations ought to be taken as advisory.

9. References

9.1. Normative References

[RFC20] Cerf, V., "ASCII format for network interchange", RFC 20, October 1969.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

9.2. Informative References

[DOI-URI] Paskin, N., Neylon, E., Hammond, T., and S. Sun, "The "doi" URI Scheme for the Digital Object Identifier (DOI)", June 2003, <http://tools.ietf.org/id/draft-paskin-doi-uri-04.txt>.

[I-D.ietf-urnbis-semantics-clarif] Klensin, J., "URN Semantics Clarification", draft-ietf-urnbis-semantics-clarif-03 (work in progress), February 2016.

[ISO3166-1]
ISO, "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", ISO 3166-1:2013, 2013.

[RFC1737] Sollins, K. and L. Masinter, "Functional Requirements for Uniform Resource Names", RFC 1737, December 1994.

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

[RFC2276] Sollins, K., "Architectural Principles of Uniform Resource Name Resolution", RFC 2276, January 1998.

[RFC2483] Mealling, M. and R. Daniel, "URI Resolution Services Necessary for URN Resolution", RFC 2483, January 1999.

[RFC3044] Rozenfeld, S., "Using The ISSN (International Serial Standard Number) as URN (Uniform Resource Names) within an ISSN-URN Namespace", RFC 3044, January 2001.

[RFC3187] Hakala, J. and H. Walravens, "Using International Standard Book Numbers as Uniform Resource Names", RFC 3187, October 2001.

[RFC3406] Daigle, L., van Gulik, D., Iannella, R., and P. Faltstrom, "Uniform Resource Names (URN) Namespace Definition Mechanisms", BCP 66, RFC 3406, October 2002.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

[RFC4854] Saint-Andre, P., "A Uniform Resource Name (URN) Namespace for Extensions to the Extensible Messaging and Presence Protocol (XMPP)", RFC 4854, April 2007.

[RFC5122] Saint-Andre, P., "Internationalized Resource Identifiers (IRIs) and Uniform Resource Identifiers (URIs) for the Extensible Messaging and Presence Protocol (XMPP)", RFC 5122, February 2008.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

[RFC6648] Saint-Andre, P., Crocker, D., and M. Nottingham, "Deprecating the "X-" Prefix and Similar Constructs in Application Protocols", BCP 178, RFC 6648, June 2012.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

[RFC6943] Thaler, D., "Issues in Identifier Comparison for Security Purposes", RFC 6943, May 2013.

[RFC6963] Saint-Andre, P., "A Uniform Resource Name (URN) Namespace for Examples", BCP 183, RFC 6963, May 2013.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, July 2013.

[RFC7282] Resnick, P., "On Consensus and Humming in the IETF", RFC 7282, June 2014.

[RFC7320] Nottingham, M., "URI Design and Ownership", BCP 190, RFC 7320, July 2014.

[RFC7613] Saint-Andre, P. and A. Melnikov, "Preparation, Enforcement, and Comparison of Internationalized Strings Representing Usernames and Passwords", RFC 7613, DOI 10.17487/RFC7613, August 2015, <http://www.rfc-editor.org/info/rfc7613>.

[UAX31] The Unicode Consortium, "Unicode Standard Annex #31: Unicode Identifier and Pattern Syntax", June 2015, <http://unicode.org/reports/tr31/>.

[UNICODE] The Unicode Consortium, "The Unicode Standard", 2015, <http://www.unicode.org/versions/latest/>.

[URI-Registry]
IANA, "Permanent URI Schemes", <http://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml#uri-schemes-1>.

[XML-BASE]
Marsh, J. and R. Tobin, "XML Base (Second Edition)", World Wide Web Consortium Recommendation REC-xmlbase-20090128, January 2009, <http://www.w3.org/TR/2009/REC-xmlbase-20090128>.

[XML-NAMES]
Thompson, H., Hollander, D., Layman, A., Bray, T., and R. Tobin, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

Appendix A. Registration Template

Namespace ID: Requested of IANA (formal) or assigned by IANA (informal).

Version: The version of the registration, starting with 1 and incrementing by 1 with each new version.

Date: The date when the registration is requested of IANA, using the format YYYY-MM-DD.

Registrant: The person or organization that has registered the NID, including the name and address of the registering organization, as well as the name and contact information (email, phone number, or postal address) of the designated contact person. If the registrant is a recognized standards development organization or scientific society requesting the fact track registration procedure (see Section 6.3), that information should be clearly indicated in this section of the template.

Purpose: Described under Section 6.4.1 of this document.

Syntax: Described under Section 6.4.2 of this document. Unless the registration explicitly says otherwise, use of q-components and f-components is not allowed for this namespace.

Assignment: Described under Section 6.4.3 of this document.

Security and Privacy: Described under Section 6.4.4 of this document.

Interoperability: Described under Section 6.4.5 of this document.

Resolution: Described under Section 6.4.6 of this document.

Documentation: A pointer to an RFC, a specification published by another standards development organization, or another stable document that provides further information about the namespace.

Additional Information Described under Section 6.4.7 of this document.

Revision Information: Description of changes from prior version(s). (Applicable only when earlier registrations have been revised.)

Additional Information: Any additional information that would be useful to the reader or those trying to understand the registration, perhaps in context with other work. May be a reference to another document or omitted if not needed.

Appendix B. Changes from RFC 2141

This document makes substantive changes from the syntax and semantics of [RFC2141]:

B.1. Syntax changes from RFC 2141

The syntax of URNs as provided in [RFC2141] was defined before the updated specification of URIs in [RFC3986]. The definition of URN syntax is updated in this document to do the following:

o Ensure consistency with the URI syntax.

o Facilitate the use of URNs with parameters similar to URI queries and fragments.

o Permit parameters influencing URN resolution.

o Ease the use of URNs with naming systems that include the '/' character.

In particular, this specification does the following:

o Extends URN syntax to explicitly allow the characters '/', "?", and "#", which were reserved for future use by RFC 2141. As described below, this change effectively also allows several components of the URI syntax although without necessarily tying those components to URI semantics.

o Defines general syntax for an additional component that can be used in interactions with a URN resolution service.

o Disallows "-" at the end of a NID.

o Allows the "/", "~", and "&" characters in the namespace-specific string (NSS).

o Makes several smaller syntax adjustments.

B.2. Other changes from RFC 2141

- Formally registers 'urn' as a URI scheme.

- Allows what are now called q-components, r-components, and f-components.

In addition, some of the text has been updated to be consistent with the definition of Uniform Resource Identifiers (URIs) [RFC3986] and the processes for registering information with the IANA [RFC5226], as well as more modern guidance with regard to security [RFC3552] and privacy [RFC6973] issues and identifier comparison [RFC6943].

Appendix C. Changes from RFC 3406

This document makes the following substantive changes from [RFC3406]:

1. Relaxes the registration policy for formal namespaces from "IETF Review" to "Expert Review" as discussed in Section 6.2.

2. Removes the category of experimental namespaces, consistent with [RFC6648]. Experimental namespaces were denoted by prefixing the namespace identifier with the string "X-". Because experimental namespaces were never registered, removing the experimental category has no impact on the existing registries. Because they are not registered, strings that refer to experimental namespaces are not valid URNs. Truly experimental usages MAY, of course, employ the 'example' namespace [RFC6963].

3. Adds some information too, but generally simplifies, the registration template.

Appendix D. Contributors

RFC 2141, which provided the basis for the syntax portion of this document, was authored by Ryan Moats.

RFC 3406, which provided the basis for the namespace portion of this document, was authored by Leslie Daigle, Dirk-Willem van Gulik, Renato Iannella, and Patrik Faltstrom.

Their work is gratefully acknowledged.

Appendix E. Acknowledgements

Many thanks to Marc Blanchet, Leslie Daigle, Martin Duerst, Juha Hakala, Ted Hardie, Alfred Hoenes, Paul Jones, Barry Leiba, Sean Leonard, Larry Masinter, Keith Moore, Mark Nottingham, Julian

Reschke, Lars Svensson, Henry S. Thompson, Dale Worley, and other participants in the URNBIS WG for their input. Alfred Hoenes in particular edited an earlier version of this document and served as co-chair of the URNBIS WG.

Juha Hakala deserves special recognition for his dedication to successfully completing this work, as do Andrew Newton and Melinda Shore in their roles as working group co-chairs and Barry Leiba in his role as area director and then as co-chair.

Appendix F. Change log for versions of draft-ietf-urnbis-rfc2141bis-urn

[[RFC Editor: please remove this appendix before publication.]]

F.1. Changes from -08 to -09

- o Altered the text in Section 4 to reflect list discussions about the earlier phrasing. Also added DOI example and citation to that section.

- o Clarified the naming rules for formal namespaces and their relationship to ISO 3166, IDNA, etc., reserved strings.

- o Added an explicit statement about use of URNs in various protocols and contexts to Section 4.

- o Clarified that experimental namespace NIDs, which were explicitly not registered, are not valid URNs (in Section 5.

- o Transformed the partial production in Section 5.2 into valid ABNF.

- o Added more text about p-/q-/f-components and recommendations about use.

- o Added clarifying note about "?" within q-components and f-components.

- o Added explicit requirement that revisions of existing registrations document the changes and added a slot for that description to the template.

- o Many small editorial changes and adjustments including adding additional references and cross-references for clarification.

- o Inserted a placeholder for additional examples.

F.2. Changes from -09 to -10

- Several clarifying editorial changes, most suggested by Ted Hardie and Henry S. Thompson (some of them off-list).

- Added a large number of placeholders that identify issues that require WG consideration and resolution (or WG delegation to the editors).

F.3. Changes from -10 to -11

- Removed most of the placeholders added in -10. Supplied new text as required or suggested by on-list discussion of those issues.

- Replaced the conformance examples Section 3.2 with a more complete collection and discussion.

- Revised and consolidated the registration procedure, and added provisions for NIDs that are the subject of standards and for avoiding race conditions about NID strings.

- In response to independent comments from Ted Hardie and Henry S. Thompson, called attention to the possibility of conflicts between NID strings and various claims of national, corporate, and other perogatives.

- Changed the production for assigned-name as suggested by Lars Svensson.

- Several clarifying editorial changes including correcting a glitch in instructions to the RFC Editor.

F.4. Changes from -11 to -12

- Removed p-components as a standalone construct, and instead folded them into the NSS.

- Defined syntax for r-components as a way to pass information to resolvers, but left the semantics for future standardization efforts.

- Further tuned the discussion of interoperability and related registration issues.

- Made a number of editorial corrections and reorganized the syntax material in Section 2 somewhat to make it internally consistent and keep the relationship to RFC 3986 clear.

F.5. Changes from -12 to -13

- More precisely defined the semantics of the optional components.

- Defined the term "resolution" and clarified several related matters throughout the text.

- Clarified terminological relationship to RFC 3986.

- Further cleansed the document of p-components.

- Corrected several examples to avoid confusion with existing identifier systems.

- Improved text regarding the purpose of namespaces being registered.

F.6. Changes from -13 to -14

- Reverted the ABNF to what had been defined in version -12.

- Added fast-track approval process for standards-related organizations, scientific societies, and similar bodies (similar to RFC 6838 for Media Types).

F.7. Changes from -14 to -15

- Reorganized the Introduction slightly, adding new subsection 1.1 and making Terminology (the former Section 2) Section 1.2.

- Tightened the discussion of "resolution" somewhat to try to mitigate some on-list confusion.

- Added some text about character set choices and repertoires (consistent with the Section 1.1 explanation).

- Moved away from "?" and "??" for q-component and r-component delimiters and went to two-character sequences for each. This includes several changes to the text to remove or modify discussions of string termination and the role of a question mark not followed by one of the new delimiters.

- Redefined r-component to be an ASCII resolver ID and a string. Neither is further defined in this specification and text has been added to say that.

- Several editorial changes to improve clarity, most following up on comments made on the list. These included modifying the table of

contents so that the subsections on optional components now appear there.

F.8. Changes from -15 (2016-02-04) to -16

- Rewrote the introductory material to make the relationship to other specifications more clear and allow removing or altering text that was stated in terms of changes from 2141. The specification is now self-contained with regard to the earlier definitions and descriptions of URNs.

- Removed the parts of Section 2 that were really a description of changes from RFC 2141 to Appendix B, where such changes are enumerated. Similarly, removed most material describing changes from RFC 3406 to Appendix C.

- Replaced one example.

- Rearranged and rewrote text to improve clarity and relationships to other documents and to reduce redundant material.

- Made it more clear that r-components, despite the partial syntax specification, are reserved for future standardization.

- Clarified that there can be URNs that neither resolve to URLs nor are abstract designators.

- Added pointers to make it clear that the Syntax material in Section 2 is not self-contained, e.g., that its subsections and other sections further restrict strings that can be used for NIDs and so on.

- Added an "Additional Information" section to the registration template. See list discussion on and about 2016-03-18.

- Minor editorial/ typographic fixes (per comment from Lars).

Authors' Addresses

Peter Saint-Andre
Filament

Email: peter@filament.com
URI: https://filament.com/

John C Klensin
1770 Massachusetts Ave, Ste 322
Cambridge, MA 02140
USA

Phone: +1 617 245 1457
Email: john-ietf@jck.com