INTERNET-DRAFT
<draft-ietf-urlreg-guide-05.txt>
March 25, 1999

Larry Masinter
Harald T. Alvestrand
Dan Zigmond
Rich Petke

Guidelines for new URL Schemes

Status of this Memo

Copyright Notice

Abstract

A Uniform Resource Locator (URL) is a compact string representation of the location for a resource that is available via the Internet. This document provides guidelines for the definition of new URL schemes.

1. Introduction

A Uniform Resource Locator (URL) is a compact string representation of the location for a resource that is available via the Internet. RFC 2396 [1] defines the general syntax and semantics of URIs, and, by inclusion, URLs. URLs are designated by including a "<scheme>:" and then a "<scheme-specific-part>". Many URL schemes are already defined.

This document provides guidelines for the definition of new URL schemes, for consideration by those who are defining and registering or evaluating those definitions.

The process by which new URL schemes are registered is defined in RFC [URL-PROCESS] [2].

2. Guidelines for new URL schemes

Because new URL schemes potentially complicate client software, new schemes must have demonstrable utility and operability, as well as compatibility with existing URL schemes. This section elaborates these criteria.

2.1 Syntactic compatibility

New URL schemes should follow the same syntactic conventions of existing schemes when appropriate. If a URI scheme that has embedded links in content accessed by that scheme does not share syntax with a different scheme, the same content cannot be served up under different schemes without rewriting the content. This can already be a problem, and with future digital signature schemes, rewriting may not even be possible. Deployment of other schemes in the future could therefore become extremely difficult.

2.1.1 Motivations for syntactic compatibility

Why should new URL schemes share as much of the generic URI syntax (that makes sense to share) as possible? Consider the following:

- If fragment syntax isn't shared between two schemes, (e.g. "<a href="#foo">"), you can't move individual completely self referential documents between schemes without rewriting the embedded references within the document. In the Web, the fragment syntax is a property of the media type, and evaluated by the client.

- If fragment syntax is not shared between different media types of the same capability (e.g. HTML, XML, Word, or image types such as GIF, JPEG, PNG) then you can't have a URI reference that can evolve to superior media types as they become available, or even likely work properly today with content negotiation.

- If relative syntax (to the extent of understanding the URI is relative, and what part of the URI string is relative) isn't shared between two schemes, (e.g. "<a href="foo">"), you can't move sets of documents that are internally self referential between schemes without rewriting the embedded URIs.

- If the ".." syntax as a path component in relative URI's isn't shared between schemes, you can't easily have sets of document sets and refer to them between schemes without rewriting the embedded references.

- If the "/" syntax (to the extent of understanding that the URI refers to a path relative to the current naming authority, see section 2.1.1) isn't shared, you can't have multiple sets of documents easily be moved up or down in a relative hierarchy of names and share a common set of documents between them, without rewriting the content, shared either in that scheme or between schemes. The best example is a site that has a common set of GIF's, JPEG and PNG images, and you want to reorganize the site changing the depth of a subtree from one depth to another, or from one directory to another where the depth isn't the same.

- If naming authority syntax (e.g. what comes after "//" in most URL schemes, see section 2.1.1) and relative path syntax is shared, to the extent of understanding that the URI has a naming authority, and what part of the URI string is the naming authority vs. path), isn't shared between two schemes, you can't share identical name spaces and serve them up via different schemes. (The naming authority syntax is a property of the scheme). The fact that HTTP, and FTP have the same syntax, for example, has often been exploited by sites transitioning from ftp archive service to HTTP archive service so that the URL's can be identical between schemes except for the scheme; the same content can be served via two schemes simultaneously.

2.1.2 Improper use of "//" following "<scheme>:"

Contrary to some examples set in past years, the use of double slashes as the first component of the <scheme-specific-part> of a URL is not simply an artistic indicator that what follows is a URL: Double slashes are used ONLY when the syntax of the URL's <scheme-specific-part> contains a hierarchical structure as described in RFC 2396. In URLs from such schemes, the use of double slashes indicates that what follows is the top hierarchical element for a naming authority. (See section 3 of RFC 2396 for more details.) URL schemes which do not contain a conformant hierarchical structure in their <scheme-specific-part> should not use double slashes following the "<scheme>:" string.

2.1.3 Compatibility with relative URLs

URL schemes should use the generic URL syntax if they are intended to be used with relative URLs. A description of the allowed relative forms should be included in the scheme's definition.

Many applications use relative URLs extensively. Specifically,

- Can the scheme be parsed according to RFC 2396 - that is, if the tokens "//", "/", ";", "?" and "#" are used, do they have the meaning given in RFC 2396?

- Does the scheme make sense to use it in relative URLs like those RFC 2396 specifies?

- If the scheme syntax is designed to be broken into pieces, does the documentation for the scheme's syntax specify what those pieces are, why it should be broken in this way, and why the breaks aren't where RFC 2396 says that they usually should be?

- If the scheme has a hierarchy, does it go left-to-right and with slash separators like RFC 2396? If not, why not?

2.1.4 Compatibility with fragment syntax

Fragment syntax should be shared across URL schemes whenever possible. Fragments indicate a location within a particular document, of a particular media type. As media types evolve, and content negotiation becomes deployed, a shared fragment syntax allows a fragment to point to the correct location within documents of different media types. For example, a named fragment (#foo), should to be able to point to the foo label in either a HTML document or an XML document. Similarly for fragments identifying a location in an image, where the image may want to evolve from GIF, to JPEG, to PNG, the fragment ID should point to the same location.

2.2 Is the scheme well defined?

It is important that the semantics of the "resource" that a URL "locates" be well defined. This might mean different things depending on the nature of the URL scheme.

2.2.1 Clear mapping from other name spaces

In many cases, new URL schemes are defined as ways to translate other protocols and name spaces into the general framework of URLs. The "ftp" URL scheme translates from the FTP protocol, while the "mid" URL scheme translates from the Message-ID field of messages.

In either case, the description of the mapping must be complete, must describe how characters get encoded or not in URLs, must describe exactly how all legal values of the base standard can be represented using the URL scheme, and exactly which modifiers,

alternate forms and other artifacts from the base standards are included or not included. These requirements are elaborated below.

2.2.2 URL schemes associated with network protocols

Most new URL schemes are associated with network resources that have one or several network protocols that can access them. The 'ftp', 'news', and 'http' schemes are of this nature. For such schemes, the specification should completely describe how URLs are translated into protocol actions in sufficient detail to make the access of the network resource unambiguous. If an implementation of the URL scheme requires some configuration, the configuration elements must be clearly identified. (For example, the 'news' scheme, if implemented using NTTP, requires configuration of the NTTP server.)

2.2.3 Definition of non-protocol URL schemes

In some cases, URL schemes do not have particular network protocols associated with them, because their use is limited to contexts where the access method is understood. This is the case, for example, with the "cid" and "mid" URL schemes. For these URL schemes, the specification should describe the notation of the scheme and a complete mapping of the locator from its source.

2.2.4 Definition of URL schemes not associated with data resources

Most URL schemes locate Internet resources that correspond to data objects that can be retrieved or modified. This is the case with "ftp" and "http", for example. However, some URL schemes do not; for example, the "mailto" URL scheme corresponds to an Internet mail address.

If a new URL scheme does not locate resources that are data objects, the properties of names in the new space must be clearly defined.

2.2.5 Character encoding

When describing URL schemes in which (some of) the elements of the URL are actually representations of sequences of characters, care should be taken not to introduce unnecessary variety in the ways in which characters are encoded into octets and then into URL characters. Unless there is some compelling reason for a particular scheme to do otherwise, translating character sequences into UTF-8 (RFC 2279) [3] and then subsequently using the %HH

encoding for unsafe octets is recommended.

2.2.6 Definition of operations

In some contexts (for example, HTML forms) it is possible to specify any one of a list of operations to be performed on a specific URL. (Outside forms, it is generally assumed to be something you GET.)

The URL scheme definition should describe all well-defined operations on the URL identifier, and what they are supposed to do.

Some URL schemes (for example, "telnet") provide location information for hooking onto bi-directional data streams, and don't fit the "infoaccess" paradigm of most URLs very well; this should be documented.

NOTE: It is perfectly valid to say that "no operation apart from GET is defined for this URL". It is also valid to say that "there's only one operation defined for this URL, and it's not very GET-like". The important point is that what is defined on this type is described.

2.3 Demonstrated utility

URL schemes should have demonstrated utility. New URL schemes are expensive things to support. Often they require special code in browsers, proxies, and/or servers. Having a lot of ways to say the same thing needless complicates these programs without adding value to the Internet.

The kinds of things that are useful include:

- Things that cannot be referred to in any other way.

- Things where it is much easier to get at them using this scheme than (for instance) a proxy gateway.

2.3.1 Proxy into HTTP/HTML

One way to provide a demonstration of utility is via a gateway which provides objects in the new scheme for clients using an existing protocol. It is much easier to deploy gateways to a new service than it is to deploy browsers that understand the new URL object.

Things to look for when thinking about a proxy are:

- o Is there a single global resolution mechanism whereby any proxy can find the referenced object?
- o If not, is there a way in which the user can find any object of this type, and "run his own proxy"?
- o Are the operations mappable one-to-one (or possibly using modifiers) to HTTP operations?
- o Is the type of returned objects well defined?
  - as MIME content-types?
  - as something that can be translated to HTML?
- o Is there running code for a proxy?

2.4 Are there security considerations?

Above and beyond the security considerations of the base mechanism a scheme builds upon, one must think of things that can happen in the normal course of URL usage.

In particular:

- o Does the user need to be warned that such a thing is happening without an explicit request (GET for the source of an IMG tag, for instance)? This has implications for the design of a proxy gateway, of course.

- o Is it possible to fake URLs of this type that point to different things in a dangerous way?

- o Are there mechanisms for identifying the requester that can be used or need to be used with this mechanism (the From: field in a mailto: URL, or the Kerberos login required for AFS access in the AFS: URL, for instance)?

- o Does the mechanism contain passwords or other security information that are passed inside the referring document in the clear (as in the "ftp" URL, for instance)?

2.5 Does it start with UR?

Any scheme starting with the letters "U" and "R", in particular if it attaches any of the meanings "uniform", "universal" or "unifying" to the first letter, is going to cause intense debate, and generate much heat (but maybe little light).

Any such proposal should either make sure that there is a large consensus behind it that it will be the only scheme of its type, or pick another name.

2.6 Non-considerations

Some issues that are often raised but are not relevant to new URL schemes include the following.

2.6.1 Are all objects accessible?

Can all objects in the world that are validly identified by a scheme be accessed by any UA implementing it?

Sometimes the answer will be yes and sometimes no; often it will depend on factors (like firewalls or client configuration) not directly related to the scheme itself.

3. Security considerations

New URL schemes are required to address all security considerations in their definitions.

4. References

[1] Berners-Lee, T., Fielding, R., Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998

[2] Petke, R., "Registration Procedures for URL Scheme Names", RFC [URL-PROCESS], November 1998

[3] Yergeau, F., "UTF-8, A Transformation Format of Unicode and ISO 10646", RFC 2279, January 1998.

5. Authors' Addresses

Larry Masinter
Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
Fax: +1-415-812-4333
EMail: masinter@parc.xerox.com

Harald Tveit Alvestrand
Maxware, Pirsenteret
N-7005 Trondheim
NORWAY
Voice: +47 73 54 57 00
EMail: harald.alvestrand@maxware.no

Dan Zigmond
WebTV Networks, Inc.
305 Lytton Avenue
Palo Alto, CA 94301
USA
Voice: +1-650-614-6071
EMail: djz@corp.webtv.net

Rich Petke
UUNET Technologies
5000 Britton Road
P. O. Box 5000
Hilliard, OH 43026-5000
Voice: +1-614-723-4157
Fax: +1-614-723-1333
EMail: rpetke@wcom.net