Internet Draft   Paul Hoffman
draft-hoffman-rfc1738bis-02.txt   VPN Consortium
April 19, 2003
Expires in six months
Intended status: Standards Track

Definitions of Early URI Schemes

Abstract

This document specifies many Uniform Resource Identifier (URI) schemes that were originally specified in RFC 1738 [RFC1738]. Some of these schemes are specified more fully in this document. The purpose of this document is to allow RFC 1738 to be moved to historic while keeping the information about the schemes on standards track.

1. Introduction

URIs are currently defined RFC 2396, which is being updated by [RFC2396BIS]. Those documents also specify how to define schemes for URIs.

The first definition for many URI schemes appeared in RFC 1738. Because that document may be moved to Historic status, this document copies the still-needed material from it to allow that material to remain on standards track. Specifically, this document copies the URI schemes.

Some of the URI scheme definitions have been changed. The following lists all of the changes:

- http: was removed because it is specified in RFC 2616

- mailto: was removed because it is specified in RFC 2368

It should be noted that three of the schemes for protocols that are described in this document (Gopher+, WAIS, and Prospero) were never documented in RFCs, and the references to them are URLs that may not be long-lasting. In fact, at least two of those URLs are no longer working at the time of this writing.

2. Specific Schemes

The mapping for some existing standard and experimental protocols is outlined in the BNF syntax definition. Notes on particular protocols follow. The schemes covered are:

ftp	File Transfer protocol
gopher	The Gopher protocol
news and nntp	USENET news
telnet	Reference to interactive sessions
wais	Wide Area Information Servers
file	Host-specific file names
prospero	Prospero Directory Service

2.1. Common Internet Scheme Syntax

The common URL syntax is described in [RFC2396BIS] and is thus not repeated here.

2.2. FTP

The ftp URL scheme is used to designate files and directories on Internet hosts accessible using the FTP protocol (RFC959).

A FTP URL follow the syntax described in Section 2.1. If :<port> is omitted, the port defaults to 21.

2.2.1. FTP Name and Password

A user name and password may be supplied; they are used in the ftp "USER" and "PASS" commands after first making the connection to the FTP server. If no user name or password is supplied and one is requested by the FTP server, the conventions for "anonymous" FTP are to be used, as follows:

> The user name "anonymous" is supplied.
>
> The password is supplied as the Internet e-mail address of the end user accessing the resource.

If the URL supplies a user name but no password, and the remote server requests a password, the program interpreting the FTP URL should request one from the user.

2.2.2. FTP url-path

The url-path of a FTP URL has the following syntax:

```
        <cwd1>/<cwd2>/.../<cwdN>/<name>;type=<typecode>
```

Where <cwd1> through <cwdN> and <name> are (possibly encoded) strings and <typecode> is one of the characters "a", "i", or "d". The part ";type=<typecode>" may be omitted. The <cwdx> and <name> parts may be empty. The whole url-path may be omitted, including the "/" delimiting it from the prefix containing user, password, host, and port.

The url-path is interpreted as a series of FTP commands as follows:

  Each of the <cwd> elements is to be supplied, sequentially, as the argument to a CWD (change working directory) command.

  If the typecode is "d", perform a NLST (name list) command with <name> as the argument, and interpret the results as a file directory listing.

  Otherwise, perform a TYPE command with <typecode> as the argument, and then access the file whose name is <name> (for example, using the RETR command.)

Within a name or CWD component, the characters "/" and ";" are reserved and must be encoded. The components are decoded prior to their use in the FTP protocol. In particular, if the appropriate FTP sequence to access a particular file requires supplying a string containing a "/" as an argument to a CWD or RETR command, it is

For example, the URL <URL:ftp://myname@host.dom/%2Fetc/motd> is interpreted by FTP-ing to "host.dom", logging in as "myname" (prompting for a password if it is asked for), and then executing "CWD /etc" and then "RETR motd". This has a different meaning from <URL:ftp://myname@host.dom/etc/motd> which would "CWD etc" and then "RETR motd"; the initial "CWD" might be executed relative to the default directory for "myname". On the other hand, <URL:ftp://myname@host.dom//etc/motd>, would "CWD " with a null argument, then "CWD etc", and then "RETR motd".

FTP URLs may also be used for other operations; for example, it is possible to update a file on a remote file server, or infer information about it from the directory listings. The mechanism for doing so is not spelled out here.

2.2.3. FTP Typecode is Optional

The entire ;type=<typecode> part of a FTP URL is optional. If it is omitted, the client program interpreting the URL must guess the appropriate mode to use. In general, the data content type of a file can only be guessed from the name, e.g., from the suffix of the name; the appropriate type code to be used for transfer of the file can then be deduced from the data content of the file.

2.2.4. Hierarchy

For some file systems, the "/" used to denote the hierarchical structure of the URL corresponds to the delimiter used to construct a file name hierarchy, and thus, the filename will look similar to the URL path. This does NOT mean that the URL is a Unix filename.

2.2.5. Optimization

Clients accessing resources via FTP may employ additional heuristics to optimize the interaction. For some FTP servers, for example, it may be reasonable to keep the control connection open while accessing multiple URLs from the same server. However, there is no common hierarchical model to the FTP protocol, so if a directory change command has been given, it is impossible in general to deduce what sequence should be given to navigate to another directory for a second retrieval, if the paths are different. The only reliable algorithm is to disconnect and reestablish the control connection.

2.3. Gopher

The gopher URL scheme is used to designate Internet resources accessible using the Gopher protocol.

The base Gopher protocol is described in [RFC1436] and supports items and collections of items (directories). The Gopher+ protocol is a set of upward compatible extensions to the base Gopher protocol and is described in [Gopher+]. Gopher+ supports associating arbitrary sets of attributes and alternate data representations with Gopher items. Gopher URLs accommodate both Gopher and Gopher+ items and item attributes.

2.3.1. Gopher URL syntax

A Gopher URL takes the form:

```
  gopher://<host>:<port>/<gopher-path>
```

where <gopher-path> is one of

```
   <gophertype><selector>
   <gophertype><selector>%09<search>
   <gophertype><selector>%09<search>%09<gopher+_string>
```

If :<port> is omitted, the port defaults to 70. <gophertype> is a single-character field to denote the Gopher type of the resource to which the URL refers. The entire <gopher-path> may also be empty, in which case the delimiting "/" is also optional and the <gophertype>

defaults to "1".

<selector> is the Gopher selector string. In the Gopher protocol, Gopher selector strings are a sequence of octets which may contain any octets except 09 hexadecimal (US-ASCII HT or tab) 0A hexadecimal (US-ASCII character LF), and 0D (US-ASCII character CR).

Gopher clients specify which item to retrieve by sending the Gopher selector string to a Gopher server.

Within the <gopher-path>, no characters are reserved.

Note that some Gopher <selector> strings begin with a copy of the <gophertype> character, in which case that character will occur twice consecutively. The Gopher selector string may be an empty string; this is how Gopher clients refer to the top-level directory on a Gopher server.

2.3.2 Specifying URLs for Gopher Search Engines

If the URL refers to a search to be submitted to a Gopher search engine, the selector is followed by an encoded tab (%09) and the search string. To submit a search to a Gopher search engine, the Gopher client sends the <selector> string (after decoding), a tab, and the search string to the Gopher server.

2.3.3 URL syntax for Gopher+ items

URLs for Gopher+ items have a second encoded tab (%09) and a Gopher+ string. Note that in this case, the %09<search> string must be supplied, although the <search> element may be the empty string.

The <gopher+_string> is used to represent information required for retrieval of the Gopher+ item. Gopher+ items may have alternate views, arbitrary sets of attributes, and may have electronic forms associated with them.

To retrieve the data associated with a Gopher+ URL, a client will connect to the server and send the Gopher selector, followed by a tab and the search string (which may be empty), followed by a tab and the Gopher+ commands.

2.3.4 Default Gopher+ data representation

When a Gopher server returns a directory listing to a client, the Gopher+ items are tagged with either a "+" (denoting Gopher+ items) or a "?" (denoting Gopher+ items which have a +ASK form associated with them). A Gopher URL with a Gopher+ string consisting of only a "+" refers to the default view (data representation) of the item while a Gopher+ string containing only a "?" refer to an item with a Gopher electronic form associated with it.

2.3.5 Gopher+ items with electronic forms

Gopher+ items which have a +ASK associated with them (i.e. Gopher+ items tagged with a "?") require the client to fetch the item's +ASK attribute to get the form definition, and then ask the user to fill out the form and return the user's responses along with the selector string to retrieve the item. Gopher+ clients know how to do this but depend on the "?" tag in the Gopher+ item description to know when to handle this case. The "?" is used in the Gopher+ string to be consistent with Gopher+ protocol's use of this symbol.

2.3.6 Gopher+ item attribute collections

To refer to the Gopher+ attributes of an item, the Gopher URL's Gopher+ string consists of "!" or "$". "!" refers to the all of a Gopher+ item's attributes. "$" refers to all the item attributes for all items in a Gopher directory.

2.3.7 Referring to specific Gopher+ attributes

To refer to specific attributes, the URL's gopher+_string is "!<attribute_name>" or "$<attribute_name>". For example, to refer to the attribute containing the abstract of an item, the gopher+_string would be "!+ABSTRACT".

To refer to several attributes, the gopher+_string consists of the attribute names separated by coded spaces. For example, "!+ABSTRACT%20+SMELL" refers to the +ABSTRACT and +SMELL attributes of an item.

2.3.8 URL syntax for Gopher+ alternate views

Gopher+ allows for optional alternate data representations (alternate views) of items. To retrieve a Gopher+ alternate view, a Gopher+ client sends the appropriate view and language identifier (found in the item's +VIEW attribute). To refer to a specific Gopher+ alternate view, the URL's Gopher+ string would be in the form:

For example, a Gopher+ string of "+application/postscript%20Es_ES" refers to the Spanish language postscript alternate view of a Gopher+ item.

2.3.9 URL syntax for Gopher+ electronic forms

The gopher+_string for a URL that refers to an item referenced by a Gopher+ electronic form (an ASK block) filled out with specific values is a coded version of what the client sends to the server. The gopher+_string is of the form:

```
+%091%0D%0A+-1%0D%0A<ask_item1_value>%0D%0A<ask_item2_value>%0D%0A.%0D%0A
```

To retrieve this item, the Gopher client sends:

```
   <a_gopher_selector><tab>+<tab>1<cr><lf>
   +-1<cr><lf>
   <ask_item1_value><cr><lf>
   <ask_item2_value><cr><lf>
   .<cr><lf>
```

to the Gopher server.

2.4. news and nntp

The news and nntp URL schemes are used to refer to either news groups or individual articles of USENET news, as specified in RFC 1036.

A news URL takes one of two forms:

```
   newsURL      =  scheme ":" [ news-server ] [ refbygroup | message ]
   scheme       =  "news" | "nntp"
   news-server  =  "//" server "/"
   refbygroup   = group [ "/" messageno [ "-" messageno ] ]
   messageno    = local-part "@" domain
```

A <group> is a period-delimited hierarchical name, such as "comp.infosystems.www.misc". A <messageno> corresponds to the Message-ID of section 2.1.5 of RFC 1036, without the enclosing "<" and ">"; it takes the form <unique>@<full_domain_name>. A message identifier may be distinguished from a news group name by the presence of the commercial at "@" character. No additional characters are reserved within the components of a news URL.

If <newsgroup-name> is "*" (as in <URL:news:*>), it is used to refer to "all available news groups".

2.5. TELNET

The Telnet URL scheme is used to designate interactive services that may be accessed by the Telnet protocol.

A telnet URL takes the form:

```
   telnet://<user>:<password>@<host>:<port>/
```

as specified in Section 2.1. The final "/" character may be omitted. If :<port> is omitted, the port defaults to 23. The :<password> can be omitted, as well as the whole <user>:<password> part.

This URL does not designate a data object, but rather an interactive service. Remote interactive services vary widely in the means by which they allow remote logins; in practice, the <user> and <password> supplied are advisory only: clients accessing a telnet URL merely advise the user of the suggested username and password.

2.6. WAIS

The WAIS URL scheme is used to designate WAIS databases, searches, or individual documents available from a WAIS database. WAIS is described in [WAIS]. The WAIS protocol is described in RFC 1625 [RFC1625]; Although the WAIS protocol is based on Z39.50-1988, the WAIS URL scheme is not intended for use with arbitrary Z39.50 services.

A WAIS URL takes one of the following forms:

```
 wais://<host>:<port>/<database>
 wais://<host>:<port>/<database>?<search>
 wais://<host>:<port>/<database>/<wtype>/<wpath>
```

where <host> and <port> are as described in Section 2.1. If :<port> is omitted, the port defaults to 210. The first form designates a WAIS database that is available for searching. The second form designates a particular search. <database> is the name of the WAIS database being queried.

The third form designates a particular document within a WAIS database to be retrieved. In this form <wtype> is the WAIS designation of the type of the object. Many WAIS implementations require that a client know the "type" of an object prior to retrieval, the type being returned along with the internal object identifier in the search response. The <wtype> is included in the URL in order to allow the client interpreting the URL adequate information to actually retrieve the document.

The <wpath> of a WAIS URL consists of the WAIS document-id. The WAIS document-id should be treated opaquely; it may only be decomposed by the server that issued it.

2.7 FILES

The file URL scheme is used to designate files accessible on a particular host computer. This scheme, unlike most other URL schemes, does not designate a resource that is universally accessible over the Internet.

A file URL takes the form:

```
   file://<host>/<path>
```

where <host> is the fully qualified domain name of the system on which the <path> is accessible, and <path> is a hierarchical directory path of the form <directory>/<directory>/.../<name>.

As a special case, <host> can be the string "localhost" or the empty string; this is interpreted as "the machine from which the URL is being interpreted". However, this part of the syntax has been ignored on many systems. That is, for some systems, the following are considered equal, while on others they are not:

```
   file://localhost/path/to/file.txt
   file:///path/to/file.txt
```

Some systems allow URLs to point to directories. In this case, there is usually (but not always) a terminating "/" character, such as in:

```
   file://usr/local/bin/
```

On systems running some versions of Microsoft Windows, the local drive specification is preceded by a "/" character. Thus, for a file called "example.ini" in the "windows" directory on the "c:" drive, the URL would be:

```
   file:///c:/windows/example.ini
```

For Windows shares, there is an additional "/" prepended to the name. Thus, the file "example.doc" on the shared directory "department" would have the URL:

```
   file:////department/example.doc
```

The file URL scheme is unusual in that it does not specify an Internet protocol or access method for such files; as such, its utility in network protocols between hosts is limited.

2.8 Prospero

The prospero URL scheme is used to designate resources that are accessed via the Prospero Directory Service. The Prospero protocol is described elsewhere [PROSPERO].

A prospero URLs takes the form:

```
  prospero://<host>:<port>/<hsoname>;<field>=<value>
```

where <host> and <port> are as described in Section 2.1. If :<port> is omitted, the port defaults to 1525. No username or password is allowed.

The <hsoname> is the host-specific object name in the Prospero protocol, suitably encoded. This name is opaque and interpreted by the Prospero server. The semicolon ";" is reserved and may not appear without quoting in the <hsoname>.

Prospero URLs are interpreted by contacting a Prospero directory server on the specified host and port to determine appropriate access methods for a resource, which might themselves be represented as different URLs. External Prospero links are represented as URLs of the underlying access method and are not represented as Prospero URLs.

Note that a slash "/" may appear in the <hsoname> without quoting and no significance may be assumed by the application. Though slashes may indicate hierarchical structure on the server, such structure is not guaranteed. Note that many <hsoname>s begin with a slash, in which case the host or port will be followed by a double slash: the slash from the URL syntax, followed by the initial slash from the <hsoname>. (E.g., <URL:prospero://host.dom//pros/name> designates a <hsoname> of "/pros/name".)

In addition, after the <hsoname>, optional fields and values associated with a Prospero link may be specified as part of the URL. When present, each field/value pair is separated from each other and from the rest of the URL by a ";" (semicolon). The name of the field and its value are separated by a "=" (equal sign). If present, these fields serve to identify the target of the URL. For example, the OBJECT-VERSION field can be specified to identify a specific version of an object.

3. Security Considerations

There are many security considerations for URIs, as described in [RFC2396BIS].

4. References

[Gopher+] Anklesaria, F., Lindner, P., McCahill, M., Torrey, D., Johnson, D., and B. Alberti, "Gopher+: Upward compatible enhancements to the Internet Gopher protocol", University of Minnesota, July 1993.

[PROSPERO] Neuman, B., and S. Augart, "The Prospero Protocol", USC/Information Sciences Institute, June 1993.

[RFC1436] Anklesaria, et. al., "Internet Gopher Protocol", RFC 1436, March 1993.

[RFC1625] St. Pierre, et. al., "WAIS over Z39.50-1988", RFC 1625, June 1994.

[RFC1738] Berners-Lee, et. al., "Uniform Resource Locators (URL)", RFC 1738, December 1994.

[RFC2396BIS] Berners-Lee, et. al., "Uniform Resource Identifier (URI): Generic Syntax", draft-fielding-uri-rfc2396bis.

[STD3] Braden, R., Editor, "Requirements for Internet Hosts -- Application and Support", STD 3, RFC 1123, October 1989.

[STD13] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[WAIS] Davis, et. al, "WAIS Interface Protocol Prototype Functional Specification", (v1.5), Thinking Machines Corporation, April 1990.

5. Authors' Contact Information

Paul Hoffman
VPN Consortium
127 Segre Place
Santa Cruz, CA 95060 USA
Phone: +1-831-426-9827
EMail: paul.hoffman@vpnc.org