draft-ietf-uri-url-03.txt,ps
Expires 1 July 1994

Tim Berners-Lee
CERN
1 Jan 1994

Uniform Resource Locators

A unifying syntax for the expression of names and addresses of objects on the network

Uniform Resource Locators

Generated from the Hypertext

March 31, 1994

Table of Contents

1. About this document

This document specifies a Uniform Resource Locator (URL), the syntax and semantics of formalized information for location and access of resources on the Internet.

This document was written by the URI working group of the Internet Engineering Task Force. Comments may be addressed to the editor, Tim Berners-Lee <timbl@info.cern.ch>, or to the URI-WG <uri@bunyip.com>. Discussions of the group are archived at

```
<http://www.acl.lanl.gov/URI/archive/uri-archive.index.html>
```

This document is bound by the Requirements Specification in preparation.

The work is derived from concepts introduced by the World-Wide Web global information initiative, whose use of such objects dates from 1990 and is described in "Universal Resource identifeirs for the World-Wide Web", RFCXXX .

This document is available in hypertext form, with links to background information, as:

```
<http://info.cern.ch/hypertext/WWW/Addressing/URL/Overview.html>
```

.

1.0.1 Status of this memo

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are working documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress".

Distribution of this document is unlimited.

1.1 Recommendations

This section describes the syntax for "Uniform Resource Locators" (URLs): that is, basically physical addresses of objects which are retrievable using protocols already deployed on the net. The generic syntax provides a framework for new schemes for names to be resolved using as yet undefined protocols.

The syntax is described in two parts. Firstly, we give the syntax rules of a completely specified name; secondly, we give the rules under which parts of the name may be omitted in a well-defined context.

1.1.1 URL syntax

A complete URL consists of a naming scheme specifier followed by a string whose format is a function of the naming scheme. For locators of information on the internet, a common syntax is used for the IP address part. A BNF description of the URL syntax is given in an a later section. The components are as follows. Fragment identifiers and partial URLs are not involved in the basic URL definition.

PrePrefix

To be a Uniform Resource Locator as currently defined by the URI working group, the whole string must start with a constant prefix "URL:". Note that to save space in this document, some URLs may have been quoted throughout without this preprefix.

Scheme

Within the URL of a object, the first element is the name of the scheme, separated from the rest of the object by a colon. The rest of the URL follows the colon in a format depending on the scheme.

Internet protocol parts

Those schemes which refer to internet protocols mostly have a common syntax for the rest of the object name. This starts with a double slash "//" to indicate its presence, and continues until the following slash "/". Within that section are

An optional user name, if required (as it is with a few FTP servers). The password, is present, follows the user name, separated from it by a colon; the user name and optional password are followed by a commercial at sign "@". The user of user name and passwords which are public is discouraged.

The internet domain name of the host in RFC1037 format (or, optionally and less advisably, the IP address as a set of four decimal digits)

The port number, if it is not the default number for the protocol, is given in decimal notation after a colon.

Path The rest of the locator is known as the "path". It may define details of how the client should communicate with the server, including information to be passed transparently to the server without any processing by the client.

The path is interpreted in a manner dependent on the scheme being used. Generally, the reserved slash "/" character (ASCII 2F hex) denotes a level in a hierarchical structure, the higher level part to the left of the slash.

1.1.2 Encoding prohibited characters

When a system uses a local addressing scheme, it is useful to provide a mapping from local addresses into URLs so that references to objects within the addressing scheme may be referred to globally, and possibly accessed through gateway servers.

Any mapping scheme may be defined provided it is unambiguous, reversible, and provides valid URLs. It is recommended that where hierarchical aspects to the local naming scheme exist, they be mapped onto the hierarchical URL path syntax in order to allow the partial form to be used.

The following encoding method shall be used for mapping WAIS, FTP, Prospero and Gopher addresses onto URLs. Where the local naming scheme uses octet values which are not allowed in the URL, these shall be represented in the URL by a percent sign "%" followed by two hexadecimal digits (0-9, A-F) giving the value for that octet. This specification makes no assumptions or requirements about the character sets, if any, referred to be the (decoded) octets a URL. Character codes other than those allowed by the syntax shall not be used unencoded in a URL.

The same encoding method may be used for encoding characters whose use, although technically allowed in a URL, would be unwise due to problems of corruption by imperfect gateways or misrepresentation due to the use of variant character sets, or which would simply be awkward in a given environment. Because a % sign always indicates an encoded character, a URL may be made safer simply by encoding any characters considered unsafe, while leaving already encoded characters still encoded. Similarly, in cases where a larger set of characters is acceptable, % signs can be selectively and reversibly expanded.

The reserved characters shall however never be arbitrarily encoded and decoded.

1.2 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

http Hypertext Transfer Protocol

ftp File Transfer protocol

gopher	The Gopher protocol
mailto	Electronic mail address
mid	Message identifiers for electronic mail
cid	Content identifiers for MIME body part
news	Usenet news
nntp	Usenet news for local NNTP access only
prospero	Access using the prospero protocols
telnet , rlogin and tn3270	Reference to interactive sessions
wais	Wide Area Information Servers

Other schemes may be specified by future specifications

New schemes may be registered at a later time.

1.2.1 FTP

The ftp: prefix indicates that the FTP protocol is used, as defined in RFC957 or any successor. The port number, if present, gives the port of the FTP server if not the FTP default.

User name and password The syntax allows for the inclusion of a user name and even a password for those systems which do not use the anonymous FTP convention. The default, however, if no user or password is supplied, will be to use that convention, viz. that the user name is "anonymous" and the password the user's Internet-style mail address .

Where possible, this mail address should correspond to a usable mail address for the user, and preferably give a DNS host name which resolves to the IP address of the client. Note that servers currently vary in their treatment of the anonymous password.

Path The FTP protocol allows for a sequence of CWD commands (change working directory) and a TYPE command prior to service commands such as RETR (retrieve) or NLIST (etc) which actually access a file.

The arguments of any CWD commands are successive segment parts of the URL delimited by slash, and the final segment is suitable as the filename argument to the RETR command for retrieval or the directory argument to NLIST.

For some file systems (Unix in particular), 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 the same as the URL path. This does NOT mean that the URL is a Unix filename.

Note: Retrieving subsequent URLs from the same host 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.

Data type The data content type of a file can only, in the general FTP case, be deduced from the name, normally the suffix of the name. This is not standardized. An alternative is for it to be transferred in information outside the URL. A suitable FTP transfer type (for example binary "I" or text "A") must in turn be deduced from the data content type. It is recommended that conventions for suffixes of public archives be established, but it is outside the scope of this standard.

An FTP URL may optionally specify the FTP data transfer type by which an object is to be retrieved. Most of the methods correspond to the FTP "Data Types" ASCII and IMAGE for the retrieval of a document, as specified in FTP by the TYPE command . One method indicates directory access.

The data type is specified by a suffix to the URL. Possible suffixes are:

;type = <type-code>	Use FTP type as given to perform data transfer.
/	Use FTP directory list commands to read directory]

The type code is in the format defined in RFC959 except that THE SPACE IS OMITTED FROM THE URL.

Transfer Mode Stream Mode is always used.

1.2.2 HTTP

The HTTP protocol specifies that the path is handled transparently by those who handle URLs, except for the servers which de-reference them. The path is passed by the client to the server with any request, but is not otherwise understood by the client.

The host details are not passed on to the client when the URL is an http URL which refers to the server in question. In this case the string sent starts with the slash which follows the host details. However, when an http server is being used as a gateway (or "proxy") then the entire URI, whether HTTP or some other scheme, is passed on the HTTP command line.The search part, if present, is sent as part of the HTTP command, and may in this respect be treated as part of the path.No fragmentid part of a WWW URI (the hash sign and following) is sent with the request. Spaces and control characters in URLs must be escaped for transmission in HTTP, as must other disallowed characters.

1.2.3 Gopher

Gopher selector strings are, in general, interpreted as a sequence of 8-bit bytes which may contain any characters other than tab, return, or linefeed. It is necessary to encode any characters disallowed in a URL, including spaces and other binary data not in the allowed character set, using the standard convention of the "%" character followed by two hexadecimal digits.

Note that slash "/" in gopher selector strings may not correspond to a level in a hierarchical structure.

The format of a gopher URL is:

- 1. A single-character field to denote the Gopher type of the resource to which the URL refers.
- 2. The gopher selector string. Note that some gopher selector strings begin with a copy of the gopher type character, in which case that character will occur twice consecutively. Also note that the gopher selector string may be an empty string since this is how gopher clients refer to the top-level directory on a gopher server.

If the URL does not refer to a Gopher+ item and if there is no gopher search string then parts 3, 4, 5, and 6 of the URL are optional

- 3. An encoded tab character (%09) to seperate the gopher selector string from the optional search string (see 4 below).
- 4. The gopher search string. If the URL refers to a search to be submitted to a gopher search engine, the search string is required. Otherwise this is an empty string.
- 5. An encoded tab character (%09) to seperate the gopher search string from the optional gopher+ string (see 6 below). Note that if the URL refers to a gopher+ item and does not have a gopher search string, there will be two encoded tab characters in a row.
- 6. The Gopher+ string. Gopher+ strings consist of a one or more characters and are 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 accomodate the various Gopher+ objects, the Gopher+ string in the URL must accomodate a mapping of the information a Gopher+ client sends to the server. This makes this section a bit long since we basically cover the entire Gopher+ protocol here.

When a Gopher server returns a directory listing to a client, Gopher+ items are tagged with either a "+" (denoting gopher+ items) or a "?" (denoting items which have a +ASK form associated with them). A Gopher+ string which is only a "+" refers to the default view (data representation) of the item. To retrieve this item a gopher+ client should send

```
a_gopher_selector<tab>+<cr><lf>
```

to the gopher+ server.

Note that items which have a +ASK asssociated with them (ie. 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 responces 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.

To refer to the Gopher+ attributes of an item, the Gopher+ string shall consist 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. To retrieve an item or directory's attributes, a gopher client will send:

```
a_gopher_selector<tab>!<cr><lf>
```

for items or

```
a_gopher_selector<tab>$<cr><lf>
```

for directories to the gopher+ server.

To refer to specific attributes, the 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, clients send the server the attribute names seperated by spaces so it is neccesary to seperate the attribute names with coded spaces. To retrieve a collection of item attributes specified with a gopher+ string of "!+ABSTRACT%20+SMELL" a gopher client would send

```
a_gopher_selector<tab>!+ABSTRACT +SMELL<cr><lf>
```

to the gopher server.

Gopher+ allows for optional alternate data representations (alternate views) of items. To retrieve a Gopher+ alternate view, the 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 "+view_name%20language_name". For example, a gopher+ string of "+application/postscript%20Es_ES" refers to the spanish language postscript alternate view of a gopher+ item. To retrieve this alternate view the client would send

```
a_gopher_selector<tab>+application/postscript Es_ES<cr><lf>
```

to the gopher server.

The gopher+ string for a URL that refers to an item referenced by an ASK form filled out with specific values is essentially a coded version of what the client sends to the server. The gopher+ string will be of the form

```
+%091%0D%0A+-1%0D%0Aask_item1_value%0D%0Aask_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.

For a really complex example, consider a URL that refers to an alternate view of an item that is referenced with a filled-out Gopher +ASK form. The gopher+ string will be of the form:

```
+view_name%20language_name%091%0D%0A+-1%0D%0Aask_item1_value%0D%0A
ask_item2_value%0D%0A.%0D%0A
```

To retrieve this item, the gopher client sends:

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

to the gopher server.

Summary: gopher+ string part of Gopher URL

To refer to an item which has an ASK form associated with it where the intent is to allow the user to enter values into the form as part of the retrieval process:

```
%3F
```

To refer to all or specific attributes of a gopher item:

```
![attribute_name][%20attribute_name][%20attribute_name]...
```

To refer to all or specific attributes of a gopher directory:

```
$[attribute_name][%20attribute_name][%20attribute_name]...
```

To refer to the content of a gopher+ item (including an item referred to by specific values in a filled-out ASK form):

```
+[view_name[%20language_name]]
 [%091%0D%0A+-1%0D%0Aask_item1_value%0D%0Aask_item2_value...%0D%0A.%0D%0A]
```

Overall summary and examples

The general format of a Gopher URL path refering to a gopher type "T" item is:

```
gopher://host [port]/T[gopher_selector]%09[search_string]%09[gopher+_string]
```

Examples: An example of a URL pointing to a gopher type 0 item (a document) is:

```
gopher://host [port]/0a_gopher_selector
```

An example of a URL pointing to a gopher type 7 item (a search engine) where the string foobar is to be submitted to the search engine is:

```
gopher://host [port]/7a_gopher_selector%09foobar
```

An example of a URL pointing to a Gopher+ type 0 item (a document) is:

```
gopher://host [port]/0a_gopher_selector%09%09some_gplus_stuff
```

An example of a URL pointing to a Gopher+ type 0 (document) item's attribute information is:

```
gopher://host [port]/0a_gopher_selector%09%09!
```

An example of a URL pointing to a Gopher+ document's spanish postscript representation is:

```
gopher://host [port]/0a_gopher_selector%09%09+application/postscript%20Es_ES
```

.

1.2.4 Mailto

This allows a URL to specify an RFC822 addr-spec mail address. Note that use of % , for example as used in forming a gatewayed mail address, requires conversion to %25 in a URL.

1.2.5 News

The news locators refer to either news group names or article message identifiers which must conform to the rules for a Message-Idof RFC 1036 (Horton 1987). A message identifier may be distinguished from a news group name by the presence of the commercial at "@" character. These rules imply that within an article, a reference to a news group or to another article will be a valid URL (in the partial form).

A news URL may be dereferenced using NNTP (RFC977, Kantor 86) (The ARTICLE by message-id command) or using any other protocol for the conveyance of usenet news articles, or by reference to a body of news articles already received.

Note1: Among URLs the "news" URLs are anomalous in that they are location-independent. They are unsuitable as URN candidates because the NNTP architecture relies on the expiry of articles and therefore a small number of articles being available at any time. When a news: URL is quoted, the assumption is that the reader will fetch the article or group from his or her local news host. News host names are NOT part of news URLs.

Note 2: An outstanding problem is that the message identifier is insufficient to allow the retrieval of an expired article, as no algorithm exists for deriving an archive site and file name. The addition of the date and news group set to the article's URL would allow this if a directory existed of archive sites by news group. Suggested subject of study in conjunction with NNTP working group. Further extension possible may be to allow the naming of subject threads as addressable objects.

NNTP

This is an alternative form of reference for news articles, specifically to be used with NNTP servers, and particularly those incomplete server implementations which do not allow retrieval by message identifier. In all other cases the "news" scheme should be used.

The news server name, newsgroup name, and index number of an article within the newsgroup on that particular server are given. The NNTP protocol must be used.

Note1. This form of URL is not of global accessability, as typically NNTP servers only allow access from local clients. Note that the article numbers within groups vary from server to server.

This form or URL should not be quoted outside this local area. It should not be used within news articles for wider circulation than the one server. This is a local identifier for a resource which is often available globally, and so is not recommended except in the case in which incomplete NNTP implementations on the local server force its adoption.

1.2.6 Prospero

The Prospero (Neuman, 1991) directory service is used to resolve the URL yielding an access method for the object (which can then itself be represented as a URL if translated). The host part contains a host name or internet address. The port part is optional.

The path part contains a host specific object name and an optional version number. If present, the version number is separated from the host specific object name by the characters "%00" (percent zero zero), this being an escaped string terminator (null). External Prospero links are represented as URLs of the underlying access method and are not represented as Prospero URLs.

1.2.7 Telnet, rlogin, tn3270

The use of URLs to represent interactive sessions is a convenient extension to their uses for objects. This allows access to information systems which only provide an interactive service, and no information server. As information within the service cannot be addressed individually or, in general, automatically retrieved, this is a less desirable, though currently common, solution.

1.2.8 WAIS

The current WAIS implementation public domain requires that a client know the "type" of a object prior to retrieval. This value is returned along with the internal object identifier in the search response. It has been encoded into the path part of the URL in order to make the URL sufficient for the retrieval of the object. Within the WAIS world, names do not of course need to be prefixed by "wais:" (by the partial form rules).

The wpath of a WAIS URL consists of encoded fields of the WAIS identifier, in the same order as inthe WAIS identifier. For each field, the identifier field number is the digits before the equals sign, and the field contents follow, encoded in the conventional encoding, terminated by ";".

1.2.9 Registration of naming schemes

A new naming scheme may be introduced by defining a mapping onto a conforming URL syntax, using a new prefix. Experimental prefixes may be used by mutual agreement between parties, and must start with the characters "x-". The scheme name "urn:" is reserved for the work in progress on a scheme for more persistent names.

It is proposed that the Internet Assigned Numbers Authority (IANA) perform the function of registration of new schemes. Any submission of a new URI scheme must include a definition of an algorithm for the retrieval of any object within that scheme. The algorithm must take the URI and produce either a set of URL(s) which will lead to the desired object, or the object itself, in a well-defined or determinable format.

It is recommended that those proposing a new scheme demonstrate its utility and operability by the provision of a gateway which will provide images of objects in the new scheme for clients using an existing protocol. If the new scheme is not a locator scheme, then the properties of names in the new space should be clearly defined. It is likewise recommended that, where a protocol allows for retrieval by URL, that the client software have provision for being configured to use specific gateway locators for indirect access through new naming schemes.

1.3 BNF for specific URL schemes

This is a BNF-like description of the Uniform Resource Locator syntax. A vertical line "|" indicates alternatives, and [brackets]indicate optional parts. Spaces are represented by the word "space", and the vertical line character by "vline". Single letters stand for single letters. All words of more than one letter below are entities described somewhere in this description.

The current IETF URI working group preference is for the prefixedurl production. (Nov 1993. July 93: url).

The "national" and "punctuation" characters do not appear in any productions and therefore may not appear in URLs.

The "afsaddress" is left in as historical note, but is not a url production

prefixedurl	u r l : url
ur l	httpaddress | ftpaddress | newsaddress | nntpaddress | prosperoaddress | telnetaddress | gopheraddress | waisaddress | mailtoaddress | midaddress | cidaddress
scheme	ialpha
httpaddress	h t t p : / / hostport [/ path][? search]
ftpaddress	f t p : / / login / path [ftptype]
afsaddress	a f s : / / cellname / path
newsaddress	n e w s : groupart
nntpaddress	n n t p : group / digits
midaddress	m i d : addr-spec
cidaddress	c i d : content-identifier
mailtoaddress	m a i l t o : : xalphas @ hostname
waisaddress	waisindex | waisdoc
waisindex	w a i s : / / hostport / database [? search]
waisdoc	w a i s : / / hostport / database / wtype / wpath
wpath	digits = path ; [wpath]
groupart	* | group | article
group	ialpha [. group]
article	xalphas @ host
database	xalphas
wtype	xalphas
prosperoaddress	prosperolink
prosperolink	p r o s p e r o : / / hostport / hsoname [% 0 0 version [attributes]]
hsoname	path
version	digits
attributes	attribute [attributes]
attribute	alphanums
telnetaddress	t e l n e t : / / login
gopheraddress	g o p h e r : / / hostport [/ gtype [gcommand]]
login	[user [: password]@]hostport
hostport	host [: port]
host	hostname | hostnumber

ftptype A formcode | E formcode | I | L digits
formcode N | T | C
cellname hostname
hostname ialpha [. hostname]
hostnumber digits . digits . digits . digits
port digits
gcommand path
path void | segment [/ path]
segment xpalphas
search xalphas [+ search]
user alphanum2 [user]
password alphanum2 [password]
fragmentid xalphas
gtype xalpha
alphanum2 alpha | digit | - | _ | . | +
xalpha alpha | digit | safe | extra | escape
xalphas xalpha [xalphas]
xpalpha xalpha | +
xpalphas xpalpha [xpalphas]
ialpha alpha [xalphas]
alpha a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z
digit 0 |1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
safe $ | - | _ | @ | . | & | + | -
extra ! | * | " | ' | (|) | ,
reserved = | ; | / | # | ? | : | space
escape % hex hex
hex digit | a | b | c | d | e | f | A | B | C | D | E | F
national { | } | vline | [|]| \| ^ | ~ [punctuation] < | >
digits digit [digits]
alphanum alpha | digit
alphanums alphanum [alphanums]
void

(end of URL BNF)

1.4 Security considerations

The URL scheme does not in itself pose a security threat. Users should beware that there is no general guarantee that a URL which at one time points to a given object continues to do so, and does not even at some later time point to a different object due to the movement of objects on servers.

A URL-related security threat is that it is sometimes possible to construct a URL such that an attempt to perform a harmless idempotent operation such as the retrieval of the object will in fact cause a possibly damaging remote operation to occur. The unsafe URL is typically constructed by specifying a port number other than that reserved for the network protocol in question. The client unwittingly contacts a server which is in fact running a different protocol. The content of the URL contains instructions which when interpreted according to this other protocol cause an unexpected operation. (An example has been the use of gopher URLs to cause a rude message to be sent via a SMTP server). It is potentially harmful for client software use any URL which specifies a port number other than the default for the protocol, especially when it is a number within the reserved space.

Care should be taken when URLs contain embedded encoded delimiters for a given protocol (for example, CR and LF characters for telnet protocols) that these are not unencoded before transmission. This misimplementation of the specification would could violate the protocol but could without violating the protocol be used to simulate an extra operation or parameter, again causing an unexpected and possible harmful remote operation to be performed.

The use of URLs containing passwords is clearly unwise.

1.5 Acknowledgements

This paper builds on the basic W3 design and much discussion of these issues by many people on the network. The discussion was particularly stimulated by articles by Clifford Lynch (1991), Brewster Kahle (1991) and Wengyik Yeong (1991b). Contributions from John Curran (NEARnet), Clifford Neuman (ISI) Ed Vielmetti (MSEN) and later the IETF URL BOF and URI working group have been incorporated into this issue of this paper.

The draft url4 (Internet Draft 00) was generated from url3 following discussion and overall approval of the URL working group on 29 March 1993. The paper url3 had been generated from udi2 in the light of discussion at the UDI BOF meeting at the Boston IETF in July 1992. Draft url4 was Internet Draft 00. Draft url5 incorporated changes suggested by Clifford Neuman, and draft url6 (ID 01) incorporated character group changes and a few other fixes defined by the IETF URI WG in submitting it as a proposed standard. URL7 (Internet Draft 02) incorporated changes introduced at the Amsterdam IETF and refined in net discussion.

The draft 03 includes changes made at Houston in Nov 93, and on the net before Seattle March 1994.

2. Appendices

The following are not formally part of this document.

2.1 Wrappers for URIs in plain text

This section does not formally form part of the URL specification .

URIs, including URLs, will ideally be transmitted though protocols which accept them and data formats which define a context for them. However, in practice nowadays there are many occasions when URLs are included in plain ASCII non-marked-up text such as electronic mail and usenet news messages.

In this case, it is convenient to have a separate wrapper syntax to define delimiters which will enable the human or automated reader to recognize that the URI is a URI.

The recommendation is that the angle brackets (less than and greater than signs) of the ASCII set be used for this purpose.

These wrappers do not form part of the URL, are not mandatory, and should not be used in contexts (such as SGML parameters, HTTP requests, etc) in which delimiters are already specified.

Example

```
Yes, Jim, I found it under <ftp://info.cern.ch/pub/www/doc> but
    you can probably pick it up from <ftp://ds.internic.net/rfc>.
```

3. References

Alberti, R., et.al. (1991) "Notes on the Internet Gopher Protocol" University of Minnesota, December 1991, <ftp://boombox.micro.umn.edu/pub/gopher/gopher_protocol> . See also <gopher://gopher.micro.umn.edu/00/Information About Gopher/About Gopher>

Berners-Lee, T ., (1991) "Hypertext Transfer Protocol (HTTP)" , CERN, December 1991, as updated from time to time, <ftp://info.cern.ch/pub/www/doc/http-spec.txt>

Crocker "Standard for ARPA Internet Text Messages" . David H. Crocker, RFC822,

Davis, F, et al., (1990) "WAIS Interface Protocol: Prototype Functional Specification", Thinking Machines Corporation, April 23, 1990 <ftp://quake.think.com/pub/wais/doc/protspec.txt>

International Standards Organization, (1991) Information and Documentation - Search and Retrieve Application Protocol Specification for open Systems Interconnection, ISO-10163

Horton (1987) M. Horton, R. Adams, "Standard for interchange of USENET messages", Internet RFC 1036 , 12/01/1987.

Huitema, C., (1991) "Naming: strategies and techniques", Computer Networks and ISDN Systems 23 (1991) 107-110.

Kahle, Brewster, (1991) "Document Identifiers, or International Standard Book Numbers for the Electronic Age", <ftp://quake.think.com/pub/wais/doc/doc-ids.txt>

Kantor, B., and Lapsley, P., (1986) "A proposed standard for the stream-based transmission of news" , Internet RFC-977, February 1986. <ftp://ds.internic.net/rfc/rfc977.txt>

Kunze, 1994 J. Kunze, Requirements for URLs, to be published.

Lynch, C., Coallition for Networked Information: (1991) "Workshop on ID and Reference Structures for Networked Information", November 1991. See <wais://quake.think.com/wais-discussion-archives?lynch>

Mockapetris, P., (1987) "Domain names + concepts and facilities", RFC-1034, USC-ISI, November 1987, <ftp://ds.internic.net/rfc/rfc1034.txt>

Neuman, B. Clifford, (1992) "Prospero: A Tool for Organizing Internet Resources", Electronic Networking: Research, Applications and Policy, Vol 1 No 2, Meckler Westport CT USA. See also <ftp://prospero.isi.edu/pub/prospero/oir.ps>

Postel, J. and Reynolds, J. (1985) "File Transfer Protocol (FTP)", Internet RFC-959, October 1985. <ftp://ds.internic.net/rfc/rfc959.txt>

Sollins 1994 K. Sollins and L. Masinter, Requiremnets for URNs, to be published.

Yeong, W., (1991a) "Towards Networked Information Retrieval", Technical report 91-06-25-01, June 1991, Performance Systems International, Inc. <ftp://uu.psi.com/wp/nir.txt>

Yeong, W., (1991b), "Representing Public Archives in the Directory", Internet Draft, November 1991, now expired.

.

4. Editor's address


```
   Tim Berners-Lee
Address:   World-Wide Web project
   CERN,
   1211 Geneva 23,
           Switzerland

     Telephone: +41 (22)767 3755
Fax:       +41 (22)767 7155
Email:     timbl@info.cern.ch
```