Tim Berners-Lee
CERN
12 March 1994

Universal Resource Identifiers in WWW

A unifying syntax for the expression of names and addresses of objects on the network as used in the World-Wide Web

Universal Resource identifiers in WWW

Generated from the Hypertext

May 10, 1994

Table of Contents

1 About this document **1**
1.0.1 Status of this memo 1

2 The need for a universal syntax **2**
2.1 URIs 2
2.2 URLs 2
2.3 URNs 2

3 Design criteria and choices **3**
3.1 Design criteria 3
3.2 Choices for a universal syntax 3

4 Recommendations **4**
4.1 URI syntax 4
4.1.1 Scheme 4
4.1.2 Path 4
4.2 Reserved characters 4
4.2.1 The percent sign 4
4.2.2 Hierarchical forms 4
4.2.3 Hash for Fragment Identifiers 4
4.2.4 Query strings 5
4.2.5 Other reserved characters 5
4.3 Unsafe characters 5
4.4 Encoding reserved characters 5
4.4.1 Conventional URI encoding scheme 5
4.4.2 Reduced or increased safe character sets 5
4.5 Partial (relative) form 7
4.6 Fragment-id 8

5 Specific Schemes **9**
5.1 HTTP 9
5.1.1 Examples 9
5.2 FTP 10
5.3 Gopher 11
5.4 Mailto 11
5.5 News 12
5.6 Telnet, rlogin, tn3270 12
5.7 URN 12
5.8 WAIS 13
5.9 file 13
5.10 Message-Id 13
5.11 Content-Id 13

5.12 Schemes for Further Study 14
5.12.1 x500 14
5.12.2 WHOIS 14
5.12.3 Network Management Database 14
5.12.4 NNTP 14
5.13 Prospero 14
5.14 Registration of naming schemes 15

6 BNF of generic URI syntax **16**
6.1 BNF for specific URL schemes 16

7 References **19**

8 Author's address **20**

1. About this document

This document defines the syntax used by the World-Wide Web initiative to encode the names and addresses of objects on the Internet. The web is considered to include objects accessed using an extendable number of protocols, existing, invented for the web itself, or to be invented in the future. Access instructions for an individual object under a given protocol are encoded into forms of address string. Other protocols allow the use of object names of various forms. In order to abstract the idea of a generic object, the web needs the concepts of the universal set of objects, and of the universal set of names or addresses of objects.

A Universal Resource Identifier (URI) is a member of this universal set of names in registered name spaces and addresses referring to registered protocols or name spaces. A Uniform Resource Locator (URL), defined elsewhere, is a form of URI which expresses an address which maps onto an access algorithm using network protocols. Existing URI schemes which correspond to the (still mutating) concept of IETF URLs are listed here. The Uniform Resource Name (URN) debate attempts to define a name space (and presumably resolution protocols) for persistent object names. This area is not addressed by this document, which is written in order to document existing practice and provide a reference point for URL and URN discussions.

This document is therefore to be issued under the "informational RFC" disclaimer .

The world-wide web protocols are discussed on the mailing list www-talk-request@info.cern.ch and the newsgroup comp.infosystems.www is preferable for beginner's questions. The mailing list uri-request@bunyip.com has discussion related particularly to the URI issue. The author may be contacted as timbl@info.cern.ch.

This document is available in hypertext form at http://info.cern.ch/hypertext/WWW/Addressing/URL/URI_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.

2. The need for a universal syntax

This section describes the concept of the URI and does not form part of the specification.

Many protocols and systems for document search and retrieval are currently in use, and many more protocols or refinements of existing protocols are to be expected in a field whose expansion is explosive.

These systems are aiming to achieve global search and readership of documents across differing computing platforms, and despite a plethora of protocols and data formats. As protocols evolve, gateways can allow global access to remain possible. As data formats evolve, format conversion programs can preserve global access. There is one area, however, in which it is impractical to make conversions, and that is in the names and addresses used to identify objects. This is because names and addresses of objects are passed on in so many ways, from the backs of envelopes to hypertext objects, and may have a long life.

A common feature of almost all the data models of past and proposed systems is something which can be mapped onto a concept of "object" and some kind of name, address, or identifier for that object. One can therefore define a set of name spaces in which these objects can be said to exist.

Practical systems need to access and mix objects which are part of different existing and proposed systems. Therefore, the concept of the universal set of all objects, and hence the universal set of names and addresses, in all name spaces, becomes important. This allows names in different spaces to be treated in a common way, even though names in different spaces have differing characteristics, as do the objects to which they refer.

2.1 URIs

This document defines a way to encapsulate a name in any registered name space, and label it with the the name space, producing a member of the universal set. Such an encoded and labelled member of this set is known as a Universal Resource Identifier, or URI.

The universal syntax allows access of objects available using existing protocols, and may be extended with technology.

The specification of the URI syntax does not imply anything about the properties of names and addresses in the various name spaces which are mapped onto the set of URI strings. The properties follow from the specifications of the protocols and the associated usage conventions for each scheme.

2.2 URLs

For existing Internet access protocols, it is necessary in most cases to define the encoding of the access algorithm into something concise enough to be termed address. URIs which refer to objects accessed with existing protocols are known as "Uniform Resource Locators" (URLs) and are listed here as used in WWW, but to be formally defined in a separate document .

2.3 URNs

There is currently a drive to define a space of more persistent names than any URLs. These "Uniform Resource Names" are the subject of an IETF working group's discussions. (See Sollins and Masinter, Functional Specifications for URNs, circulated informally.)

The URI syntax and URL forms have been in widespread use by World-Wide Web software since 1990.

3. Design criteria and choices

This section is not part of the specification: it is simply an explanation of the way in which the specification was derived.

3.1 Design criteria

The syntax was designed to be

Extensible New naming schemes may be added later.

Complete It is possible to encode any naming scheme.

Printable It is possible to express any URI using 7-bit ASCII characters so that URIs may if necessary be passed using pen and ink.

3.2 Choices for a universal syntax

For the syntax itself there is little choice except for the order and punctuation of the elements, and the acceptable characters and escaping rules.

The extensibility requirement is met by allowing an arbitrary (but registered) string to be used as a prefix. A prefix is chosen as left to right parsing is more common than right to left. The choice of a colon as separator of the prefix from the rest of the URI was arbitrary.

The decoding of the rest of the string is defined as a function of the prefix. New prefixed are introduced for new schemes as necessary, in agreement with the registration authority. The registration of a new scheme clearly requires the definition of the decoding of the URI into a given name space, and a definition of the properties and, where applicable, resolution protocols, for the name space.

The completeness requirement is easily met by allowing particularly strange or plain binary names to be encoded in base 16 or 64 using the acceptable characters.

The printability requirement could have been met by requiring all schemes to encode characters not part of a basic set. This led to many discussions of what the basic set should be. A difficult case, for example, is when an ISO latin 1 string appears in a URL, and within an application with ISO Latin-1 capability, it can be handled intact. However, for transport in general, the non-ASCII characters need to be escaped.

The solution to this was to specify a safe set of characters, and a general escaping scheme which may be used for encoding "unsafe" characters. This "safe" set is suitable, for example, for use in electronic mail. This is the canonical form of a URI.

The choice of escape character for introducing representations of non-allowed characters also tends to be a matter of taste. An ANSI standard exists in the C language, using the back-slash character "\". The use of this character on unix command lines, however, can be a problem as it is interpreted by many shell programs, and would have itself to be escaped. It is also a character which is not available on certain keyboards. The equals sign is commonly used in the encoding of names having attribute=value pairs. The percent sign was eventually chosen as a suitable escape character.

There is a conflict between the need to be able to represent many characters including spaces within a URI directly, and the need to be able to use a URI in environments which have limited character sets or in which certain characters are prone to corruption. This conflict has been resolved by use of an hexadecimal escaping method which may be applied to any characters forbidden in a given context. When URLs are moved between contexts, the set of characters escaped may be enlarged or reduced unambiguously.

The use of white space characters is risky in URIs to be printed or sent by electronic mail, and the use of multiple white space characters is very risky. This is because of the frequent introduction of extraneous white space when lines are wrapped by systems such as mail, or sheer necessity of narrow column width, and because of the inter-conversion of various forms of white space which occurs during character code conversion and the transfer of text between applications. This is why the canonical form for URIs has all white spaces encoded.

4. Recommendations

This section describes the syntax for URIs as used in the WorldWide Web initiative. The generic syntax provides a framework for new schemes for names to be resolved using as yet undefined protocols.

4.1 URI syntax

A complete URI 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 relative URIs are not involved in the basic URL definition.

4.1.1 Scheme

Within the URI of a object, the first element is the name of the scheme, separated from the rest of the object by a colon.

4.1.2 Path

The rest of the URI follows the colon in a format depending on the scheme. The path is interpreted in a manner dependent on the protocol being used. However, when it contains slashes, these must imply a hierarchical structure.

4.2 Reserved characters

The path in the URI has a significance defined by the particular scheme. Typically it is used to encode a name in a given name space, or an algorithm for accessing an object. In either case, the encoding may use those characters allowed by the BNF syntax, or hexadecimal encoding of other characters.

Some of the reserved characters have special uses as defined here.

4.2.1 The percent sign

The percent sign ("%", ASCII 25 hex) is used as the escape character in the encoding scheme and is never allowed for anything else.

4.2.2 Hierarchical forms

The slash ("/", ASCII 2F hex) character is reserved for the delimiting of substrings whose relationship is hierarchical. This enables partial forms of the URI. Substrings consisting of single or double dots ("." or "..") are similarly reserved.

The significance of the slash between two segments is that the segment of the path to the left is more significant than the segment of the path to the right. ("Significance" in this case refers solely to closeness to the root of the hierarchical structure and makes no value judgement!)

Note

The similarity to unix and other disk operating system filename conventions should be taken as purely coincidental, and should not be taken to indicate that URIs should be interpreted as file names.

4.2.3 Hash for Fragment Identifiers

The hash ("#", ASCII 23 hex) character is reserved as a delimiter to separate the URI of an object from a fragment identifier .

4.2.4 Query strings

The question mark ("?", ASCII 3F hex) is used to delimit the boundary between the URI of a queryable object, and a set of words used to express a query on that object. When this form is used, the combined URI stands for the object which results from the query being applied to the original object.

Within the query string, the plus sign is reserved as shorthand notation for a space. Therefore, real plus signs must be encoded. This method was used to make query URIs easier to pass in systems which did not allow spaces.

The query string represents some operation applied to the object, but this specification gives no common syntax or semantics for it. In practice the syntax and sematics may depend on the scheme and may even on the base URI.

4.2.5 Other reserved characters

The astersik ("*", ASCII 2A hex) and exclamation mark ("!" , ASCII 21 hex) are reserved for use as having special signifiance within specific schemes.

4.3 Unsafe characters

In canonical form, certain characters such as spaces, control characters, some characters whose ASCII code is used differently in different national character variant 7 bit sets, and all 8bit characters beyond DEL (7F hex) of the ISO Latin-1 set, shall not be used unencoded. This is a recommendation for trouble-free interchange, and as indicated below, the encoded set may be extended or reduced.

4.4 Encoding reserved characters

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

For a new naming scheme, any mapping scheme may be defined provided it is unambiguous, reversible, and provides valid URIs. 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.

It is also recommended that the conventional scheme below be used in all cases except for any scheme which encodes binary data as opposed to text, in which case a more compact encoding such as pure hexadecimal or base 64 might be more appropriate. For example, the conventional URI encoding method is used for mapping WAIS, FTP, Prospero and Gopher addresses in the URI specification.

4.4.1 Conventional URI encoding scheme

Where the local naming scheme uses ASCII characters which are not allowed in the URI, these may be represented in the URL by a percent sign "%" immediately followed by two hexadecimal digits (0-9, A-F) giving the ISO Latin 1 code for that character. Character codes other than those allowed by the syntax shall not be used unencoded in a URI.

4.4.2 Reduced or increased safe character sets

The same encoding method may be used for encoding characters whose use, although technically allowed in a URI, 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 URI 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.

Before two URIs can be compared, it is therefore necessary to bring them to the same encoding level.

However, the reserved characters mentioned above have a quite different significance when encoded, and so may NEVER be encoded and unencoded in this way.

The percent sign intended as such must always be encoded, as its presence otherwise always indicates an encoding. Sequences which start with a percent sign but are not followed by two hexadecimal characters are reserved for future extension. (see example 3)

Example 1

The URIs

```
http://info.cern.ch/albert/bertram/marie-claude
```

and

```
http://info.cern.ch/albert/bertram/marie%2Dclaude
```

are identical, as the %2D encodes a hyphen character.

Example 2

The URIs

```
  http://info.cern.ch/albert/bertram/marie-claude
```

and

```
  http://info.cern.ch/albert/bertram%2Fmarie-claude
```

are NOT identical, as in the second case the encoded slash does not have hierarchical significance.

Example 3

The URIs

```
fxqn:/us/va/reston/cnri/ietf/24/asdf%*.fred
```

and

```
news:12345667123%asdghfh@info.cern.ch
```

are illegal, as all % characters imply encodings, and there is no decoding defined for "%*" or "%as" in this recommendation.

4.5 Partial (relative) form

Within a object whose URI is well defined, the URI of another object may be given in abbreviated form, where parts of the two URIs are the same. This allows objects within a group to refer to each other without requiring the space for a complete reference, and it incidentally allows the group of objects to be moved without changing any references. It must be emphasized that when a reference is passed in anything other than a well controlled context, the full form must always be used.

In the World-Wide Web applications, the context URI is that of the document or object containing a reference. In this case partial URIs can be generated in virtual objects or stored in real objects, without the need for dramatic change if the higher-order parts of a hierarchical naming system are modified. Apart from terseness, this gives greater robustness to practical systems, by enabling information hiding between system components.

The partial form relies on a property of the URI syntax that certain characters ("/") and certain path elements ("..", ".") have a significance reserved for representing a hierarchical space, and must be recognized as such by both clients and servers.

A partial form can be distinguished from an absolute form in that the latter must have a colon and that colon must occur before any slash characters. Systems not requiring partial forms should not use any unencoded slashes in their naming schemes. If they do, absolute URIs will still work, but confusion may result. (See note on Gopher below).

The rules for the use of a partial name relative to the URI of the context are:

- If the scheme parts are different, the whole absolute URI must be given. Otherwise, the scheme is omitted, and:
- If the partial URI starts with a non-zero number of consecutive slashes, then everything from the context URI up to (but not including) the first occurrence of exactly the same number of consecutive slashes which has no greater number of consecutive slashes anywhere to the right of it is taken to be the same and so prepended to the partial URL to form the full URL. Otherwise:
- The last part of the path of the context URI (anything following the rightmost slash) is removed, and the given partial URI appended in its place, and then:
- Within the result, all occurrences of "xxx/../" or "/." are recursively removed, where xxx, ".." and "." are complete path elements.

Note: Trailing slashes If a path of the context locator ends in slash, partial URIs are treated differently to the URI with the same path but without a trailing slash. The trailing slash indicates a void segment of the path.

Note: Gopher The gopher system does not have the concept of relative URIs, and the gopher community currently allows / as data characters in gopher URIs without escaping them to %2F. Relative forms may not in general be used for documents served by gopher servers. If they are used, then WWW software assumes, normally correctly, that in fact they do have hierarchical significance despite the specifications. The use of HTTP rather than gopher protocol is however recommended.

Examples

In the context of URI

```
magic://a/b/c//d/e/f
```

the partial URIs would expand as follows:

g	magic://a/b/c//d/e/g
/g	magic://a/g

//g	magic://g
../g	magic://a/b/c//d/g
g:h	g:h

and in the context of the URI

```
magic://a/b/c//d/e/
```

the results would be exactly the same.

4.6 Fragment-id

This represents a part of, fragment of, or a sub-function within, an object . Its syntax and semantics are defined by the application responsible for the object, or the specification of the content type of the object. The only definition here is of the allowed characters by which it may be represented in a URL.

Specific syntaxes for representing fragments in text documents by line and character range, or in graphics by coordinates, or in structured documents using ladders, are suitable for standardization but not defined here.

The fragment-id follows the URL of the whole object from which it is separated by a hash sign (#). If the fragment-id is void, the hash sign may be omitted: A void fragment-id with or without the hash sign means that the URL refers to the whole object.

While this hook is allowed for identification of fragments, the question of addressing of parts of objects, or of the grouping of objects and relationship between continued and containing objects, is not addressed by this document.

Fragment identifiers do NOT address the question of objects which are different versions of a "living" object, nor of expressing the relationships between different versions and the living object.

There is no implication that a fragment identifier refers to anything which can be extracted as an object in its own right. It may, for example, refer to an indivisible point within an object.

5. Specific Schemes

The mapping for URIs onto some existing standard and experimental protocols is outlined in the BNF syntax definition . Notes on particular protocols follow. These URIs are frequently referred to as URLs, though the exact definition of the term URL is still under discussion (March 1993). The schemes covered are:

http	Hypertext Transfer Protocol (examples)
ftp	File Transfer protocol
gopher	Gopher protocol
mailto	Electronic mail address
news	Usenet news
telnet , rlogin and tn3270	Reference to interactive sessions
wais	Wide Area Information Servers
file	Local file access

The following schemes are proposed as essential to the unification of the web with electronic mail, but not currently (to the author's knowledge) implemented:

mid	Message identifiers for electronic mail
cid	Content identifiers for MIME body part

The schemes for x.500, network management database, and whois++ have not been specified and may be the subject of further study. Schemes for Prospero , and restricted NNTP use are not currently implemented as far as the author is aware.

The "urn" prefix is reserved for use in encoding a Uniform Resource Name when that has been developed by the IETF working group.

New schemes may be registered at a later time.

5.1 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.

5.1.1 Examples

These examples are not part of the specification: they are provided as illustations only. The URI of the "welcome" page to a server is conventionally

```
http://www.my.work.com/
```

As the rest of the URL (after the hostname an port) is opaque to the client, it shows great variety but the following are all fairly typical.

```
http://www.my.uni.edu/info/matriculation/enroling.html
```

```
http://info.my.org/AboutUs/Phonebook
```

```
http://www.library.my.town.va.us/Catalogue/76523471236%2Fwen44--4.98

http://www.my.org/462F4F2D4241522A314159265358979323846
```

A URL for a server on a different port to 80 looks like

```
http://info.cern.ch:8000/imaginary/test
```

A reference to a particular part of a document may, including the fragment identifier, look like

```
http://www.myu.edu/org/admin/people#andy
```

in which case the string "#andy" is not sent to the server, but is retained by the client and used when the whole object had been retrieved.
A search on a text database might look like

```
http://info.my.org/AboutUs/Index/Phonebook?dobbins
```

and on another database

```
http://info.cern.ch/RDB/EMP?*%20where%20name%%3Ddobbins
```

In all cases the client passes the path string to the server uninterpreted, and for the client to deduce anything from

5.2 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.

5.3 Gopher

The gopher URL specifies the host and optionally the port to which the client should connect. This is followed by a slash and a single gopher type code. This type code is used by the client to determine how to interpret the server's reply and is is not for sending to server. The command string to be sent to the server immediately follows the gopher type character. It consists of the gopher selector string followed by any "Gopher plus" syntax, but always omitting the trainling CR LF pair.

When the gopher command string contains characters (such a embedded CR LF and HT characters) not allowed in a URL, these are encoded using the conventional encoding.

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 encoded command string (with trailing CR LF stripped) would be void then the gopher type character may be omiited and "1" (ASCII 31 hex) is assumed.

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

5.4 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.

5.5 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.

5.6 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.

5.7 URN

The "Universal Resource Name" is currently (March 1993) under development in the IETF. A requirements specification is in preparation. It currently looks as though it will be a short string suitable for encoding in URI syntax, for which case the "urn:" prefix is reserved. The URN shall be encoded precisely as defined in the (future) URN standard, except in that:

- If the official description of the URN syntax includes any constant wrapper characters, then they shall not be omitted from the URI encoding of the URN;
- If the URN has a hierarchical nature, then the slash delimiter shall be used in the URI encoding;
- If the URN has a hierarchical nature, the most significant part shall be encoded on the left in the URI encoding;
- Any characters with reserved meanings in the URI syntax shall be escape encoded

These rules of course apply to any URI scheme. It is of course possible that the URN syntax will be chosen such that the URI encoding will be a 1-1 transcription.
An example might be a name such as

```
urn:/iana/dns/ch/cern/cn/techdoc/94/1642-3
```

but the reader should refer to the latest URN drafts or specifications.

5.8 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 ";".

5.9 file

The other URI schemes (except nntp) share the property that they are equally valid at any geographical place.

There is however a real practical requirement to be able to generate a URL for an object in a machine's local file system.

The syntax is similar to the ftp syntax, but in this case the slash is used to donate boundaries between directory levels of a hierarchical file system is used. The "client" software converts the file URL into a file name in the local file name conventions. This allows local files to be treated just as network objects without any necessity to use a network server for access. This may be used for example for defining a user's "home" document in WWW.

There is clearly a danger of confusion that a link made to a local file should be followed by someone on a different system, with unexpected and possibly harmful results. Therefore, the convention is that even a "file" URL is provided with a host part. This allows a client on another system to know that it cannot access the file system, or perhaps to use some other local mecahnism to access the file.

The special value "localhost" is used in the host field to indicate that the filename should really be used on whatever host one is. This for example allows links to be made to files which are distribted on many machines, or to "your unix local password file" subject of course to consistency across the users of the data.

A void host field is equivalent to "localhost".

</ADDRESS>

5.10 Message-Id

For systems which include information transferred using mail protocols, there is a need to be able to make cross-references between different items of information, even though, by the nature of mail, those items are only available to a restricted set of people.

Two schemes are defined. The first, "mid:", refers to the RFC822 Message-Id of a mail message. This Identifier is already used in RFC822 in for example the References and In-Reply-to field . The rest of the URL after the "mid:" is the RFC822 msg-id with the constant <> wrapper removed, leaving an identifier whose format in fact happens to be the same as addr-spec format for mailboxes (though the semantics are different).

The use of a "mid" URL implies access to a body of mail already received. If a message has been distributed using NNTP or other usenet protocols over the news system, then the "news:" form should be used.

5.11 Content-Id

The second scheme, "cid:", is similar to "mid:" , but makes reference to a body part of a MIME message by the value of its content-id field. This allows, for example, a master document being the first part of a multipart/related MIME message to refer to component parts which are transferred in the same message.

Note

Beware however, that content identifiers are only required to be unique within the context of a given MIME message, and so the cid: URL is only meaningful with the context the same MIME message. For a reference outside the message, it would need to be appended to the message-id of the whole message. A syntax for this has not been defined.

5.12 Schemes for Further Study

5.12.1 x500

The mapping of x500 names onto URLs is not defined here. A decision is required as to whether "distinguished names" or "user friendly names" (ufn), or both, should be allowed. If any punctuation conversions are needed from the adopted x500 representation (such as the use of slashes between parts of a ufn) they must be defined. This is a subject for study.

5.12.2 WHOIS

This prefix describes the access using the "whois++" scheme in the process of definition. The host name part is the same as for other IP based schemes. The path part can be either a whois handle for a whois object, or it can be a valid whois query string. This is a subject for further study.

5.12.3 Network Management Database

This is a subject for study.

5.12.4 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.

5.13 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.

5.14 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.

6. BNF of generic URI syntax

This is a BNF-like description of the URI syntax. at the level at which specific schemes are not considered.

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 "generic" production gives a higher level parsing of the same URIs as the other productions. The "national" and "punctuation" characters do not appear in any productions and therefore may not appear in URIs.

fragmentaddress uri [# fragmentid]
uri scheme : path [? search]
scheme ialpha
path void | xpalphas [/ path]
search xalphas [+ search]
fragmentid xalphas
xalpha alpha | digit | safe | extra | escape
xalphas xalpha [xalphas]
xpalpha xalpha | +
xpalphas xpalpha [xpalpha]
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] < | >
void

(end of URI BNF)

6.1 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)

7. 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.

.

8. Author'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