



The page cannot be found

The Online Requests For Comments - RFCs

[Home](#) | [Books](#) | [Bookmark!](#) | [Link to Us](#) | [Help](#)

RFC 2295



Network Working Group
Request for Comments: 2295
Category: Experimental

[Selected Books:](#)

Transparent Content Negotiation



Status of this Memo

This memo defines an Experimental Protocol fo community. It does not specify an Internet s Discussion and suggestions for improvement ar Distribution of this memo is unlimited.

[Buy this book!](#)

Copyright Notice

Copyright (C) The Internet Society (1998). A



ABSTRACT

HTTP allows web site authors to put multiple information under a single URL. Transparent an extensible negotiation mechanism, layered automatically selecting the best version when This enables the smooth deployment of new web tags.

[Buy this book!](#)

TABLE OF CONTENTS

- 1 Introduction.....
- 1.1 Background.....
- 2 Terminology.....
- 2.1 Terms from HTTP/1.1.....

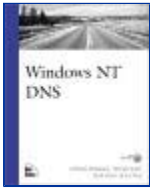


[Buy this book!](#)





[Buy this book!](#)



[Buy this book!](#)

- 2.2 New terms.....
- 3 Notation.....
- 4 Overview.....
 - 4.1 Content negotiation.....
 - 4.2 HTTP/1.0 style negotiation scheme.....
 - 4.3 Transparent content negotiation scheme..
 - 4.4 Optimizing the negotiation process.....
 - 4.5 Downwards compatibility with non-negotia
 - 4.6 Retrieving a variant by hand.....
 - 4.7 Dimensions of negotiation.....

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

- 4.8 Feature negotiation.....
- 4.9 Length of variant lists.....
- 4.10 Relation with other negotiation schemes
- 5 Variant descriptions.....
 - 5.1 Syntax.....
 - 5.2 URI.....
 - 5.3 Source-quality.....
 - 5.4 Type, charset, language, and length.....
 - 5.5 Features.....
 - 5.6 Description.....
 - 5.7 Extension-attribute.....
- 6 Feature negotiation.....
 - 6.1 Feature tags.....
 - 6.1.1 Feature tag values.....
 - 6.2 Feature sets.....
 - 6.3 Feature predicates.....
 - 6.4 Features attribute.....
- 7 Remote variant selection algorithms.....
 - 7.1 Version numbers.....
- 8 Content negotiation status codes and heade
 - 8.1 506 Variant Also Negotiates.....
 - 8.2 Accept-Features.....
 - 8.3 Alternates.....
 - 8.4 Negotiate.....
 - 8.5 TCN.....

8.6	Variant-Vary.....
9	Cache validators.....
9.1	Variant list validators.....
9.2	Structured entity tags.....
9.3	Assigning entity tags to variants.....
10	Content negotiation responses.....
10.1	List response.....
10.2	Choice response.....
10.3	Adhoc response.....
10.4	Reusing the Alternates header.....
10.5	Extracting a normal response from a cho
10.6	Elaborate Vary headers.....
10.6.1	Construction of an elaborate Vary hea
10.6.2	Caching of an elaborate Vary header..
10.7	Adding an Expires header for HTTP/1.0 c
10.8	Negotiation on content encoding.....

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

11	User agent support for transparent negotia
11.1	Handling of responses.....
11.2	Presentation of a transparently negotia
12	Origin server support for transparent nego
12.1	Requirements.....
12.2	Negotiation on transactions other than
13	Proxy support for transparent negotiation.
14	Security and privacy considerations.....
14.1	Accept- headers revealing personal info
14.2	Spoofing of responses from variant reso
14.3	Security holes revealed by negotiation.
15	Internationalization considerations.....
16	Acknowledgments.....
17	References.....
18	Authors' Addresses.....

19	Appendix: Example of a local variant selection
19.1	Computing overall quality values.....
19.2	Determining the result.....
19.3	Ranking dimensions.....
20	Appendix: feature negotiation examples....
20.1	Use of feature tags.....
20.2	Use of numeric feature tags.....
20.3	Feature tag design.....
21	Appendix: origin server implementation conventions
21.1	Implementation with a CGI script.....
21.2	Direct support by HTTP servers.....
21.3	Web publishing tools.....
22	Appendix: Example of choice response construction
23	Full Copyright Statement.....

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation

1 Introduction

HTTP allows web site authors to put multiple information under a single URI. Each of these is a 'variant'. Transparent content negotiation is a negotiation mechanism for automatically and efficiently selecting the best variant when a GET or HEAD request is received. This mechanism allows for the smooth deployment of new web data formats.

This specification defines transparent content negotiation as an extension on top of the HTTP/1.1 protocol [1]. This extension does not require use of HTTP/1.1: transparent content negotiation can also be done if some or all of the systems are HTTP/1.0 [2] systems.

Transparent content negotiation is called `tr` makes all variants which exist inside the ori outside parties.

Note: Some members of the IETF are currentl of activities which are loosely related to protocol. First, there is an effort to def independent registry for feature tags. The experimental protocol will be one of the cl Second, some research is being done on cont for other transport protocols (like interne and on generalized negotiation systems for protocols. At the time of writing, it is u research will lead to results in the form o system specifications. It is also unclear future specifications can or will re-use el experimental protocol.

1.1 Background

The addition of content negotiation to the we been considered important since the early day expected benefits of a sufficiently powerful negotiation are

- * smooth deployment of new data formats and allow graceful evolution of the web
- * eliminating the need to choose between a multimedia homepage' and one which can be
- * enabling good service to a wider range of platforms (from low-end PDA's to high-end

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

- * eliminating error-prone and cache-unfrien User-Agent based negotiation
- * enabling construction of sites without `c version' links
- * internationalization, and the ability to

content without a bias towards one language

2 Terminology

The words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT" in this document are to be interpreted as described in [1].

This specification uses the term 'header' as defined in [1] as 'header field in a request or response message'.

2.1 Terms from HTTP/1.1

This specification mostly uses the terminology defined in [1]. For the convenience of the reader, this section reproduces some key terminology definitions from [1].

request

An HTTP request message.

response

An HTTP response message.

resource

A network data object or service that can be identified by a URI. Resources may be available in multiple representations in multiple languages, data formats, sizes, and media types, or in other ways.

content negotiation

The mechanism for selecting the appropriate representation servicing a request.

client

A program that establishes connections for requests.

user agent

The client which initiates a request. These include web browsers, editors, spiders (web-traversing robots), and other user agents.

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation

server

An application program that accepts connect requests by sending back responses. Any gi capable of being both a client and a server refers only to the role being performed by particular connection, rather than to the p general. Likewise, any server may act as a gateway, or tunnel, switching behavior base request.

origin server

The server on which a given resource reside

proxy

An intermediary program which acts as both for the purpose of making requests on behal Requests are serviced internally or by pass possible translation, to other servers. A both the client and server requirements of

age

The age of a response is the time since it successfully validated with, the origin ser

fresh

A response is fresh if its age has not yet lifetime.

2.2 New terms

transparently negotiable resource

A resource, identified by a single URI, whi representations (variants) associated with request on its URI, it allows selection of using the transparent content negotiation m transparently negotiable resource always ha to it, which can be represented as an Alter section 8.3).

variant list

A list containing variant descriptions, whi transparently negotiable resource.

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

variant description

A machine-readable description of a variant in a variant list. A variant description contains a resource URI and various attributes which define the variant. Variant descriptions are defined in section 10.1.

variant resource

A resource from which a variant of a negotiated resource can be retrieved with a normal HTTP/1.x GET request which does not use transparent content negotiation.

neighboring variant

A variant resource is called a neighboring variant if it is a transparently negotiable HTTP resource if it is a HTTP URL, and if the absolute URL of the variant's last slash equals the absolute URL of the resource up to its last slash, where equality is determined by the comparison rules in section 3.2.3 of [1]. A neighboring variant is important because of its proximity to the resource (section 14.2). Not all variants of a negotiated resource are neighboring variants. However, access to a neighboring variant can be more highly optimized by the use of choice algorithms (section 7) and choice responses (section 10.1).

remote variant selection algorithm

A standardized algorithm by which a server selects the best variant on behalf of a negotiating user. The algorithm typically computes whether the Accept-Range header contains sufficient information to allow a choice response to select a variant is the best variant. The use of a standardized algorithm speeds up the negotiation process.

list response

A list response returns the variant list of a resource, but no variant data. It can be generated if the user does not want to, or is not allowed to, retrieve a specific variant for the request. List responses are defined in section 10.1.

choice response

A choice response returns a representation of the resource requested, and may also return the variant list.

the request, and may also return the variant resource. It can be generated when the server information to be able to choose the best variant user agent, but may only be generated if there is a neighboring variant. Choice responses are

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation

ad hoc response

An ad hoc response can be sent by an origin server, to achieve compatibility with a client if this compatibility cannot be achieved by a choice response. There are very little contents of an ad hoc response. Ad hoc response is defined in section 10.3.

Accept- headers

The request headers: Accept, Accept-Charset, Accept-Features.

server supports transparent content negotiation

From the viewpoint of an origin server or proxy server, a server supports transparent content negotiation if it sends a Negotiate header (section 8.4) which indicates that it

server-side override

If a request on a transparently negotiated resource from a client which supports transparent content negotiation and the server is said to perform a server-side override, the server ignores the directives in the Negotiate request and uses a custom algorithm to choose an appropriate response. A server-side override can sometimes be used to work around client bugs. It could also be used by protocols to support transparent content negotiation.

3 Notation

The version of BNF used in this document is that of the nonterminals used are defined in [1]. The underlying charset is US-ASCII.

One new BNF construct is added:

1%rule

stands for one or more instances of "rule", s

1%rule = rule *(1*LWS rule)

This specification also introduces

number = 1*DIGIT

short-float = 1*3DIGIT ["." 0*3DIGIT]

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

This specification uses the same conventions 1.2 of [1]) for defining the significance of requirement.

4 Overview

This section gives an overview of transparent It starts with a more general discussion of n by HTTP.

4.1 Content negotiation

HTTP/1.1 allows web site authors to put multi information under a single resource URI. Eac called a `variant'. For example, a resource h bind to three different variants of a paper:

1. HTML, English
2. HTML, French
3. Postscript, English

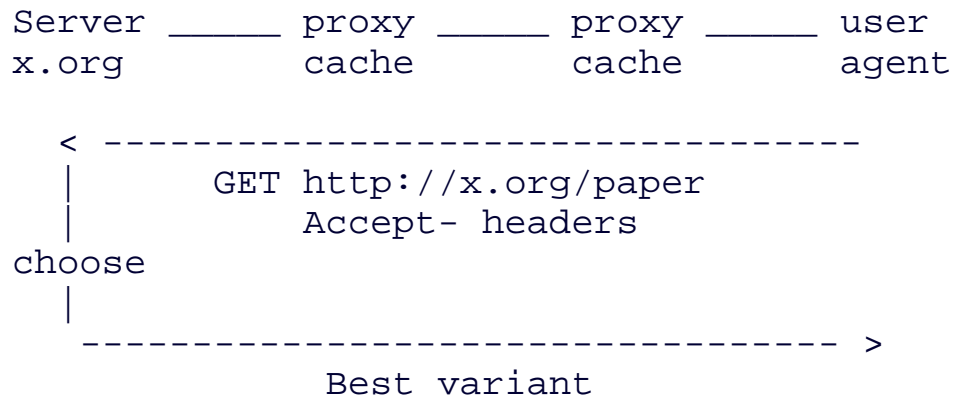
Content negotiation is the process by which t selected if the resource is accessed. The se matching the properties of the available vari of the user agent and the preferences of the

It has always been possible under HTTP to hav

representations available for one resource, a appropriate representation for each subsequent HTTP/1.1 is the first version of HTTP which handles this in a cache-friendly way. These provisions include the response header, entity tags, and the If-None-Match header.

4.2 HTTP/1.0 style negotiation scheme

The HTTP/1.0 protocol elements allow for a negotiation scheme as follows:



Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation

When the resource is accessed, the user agent sends a request with various Accept-headers which express capabilities and the user preferences. Then the server chooses the best variant in the response.

The biggest problem with this scheme is that for all but the most minimal user agents, sending all capabilities and preferences would be very inefficient because only a small fraction of the resource variants are used.

4.3 Transparent content negotiation scheme

The transparent content negotiation scheme eliminates the need to send huge Accept-headers, and nevertheless a process that always yields either the best variant or a message indicating that user agent is not capable of the available variants.

of the available variants.

Under the transparent content negotiation scheme, the server returns a list with the available variants and their priorities to the user agent. An example of a list with three variants is:

```
{ "paper.1" 0.9 {type text/html} {language
  "paper.2" 0.7 {type text/html} {language
  "paper.3" 1.0 {type application/postscript}
```

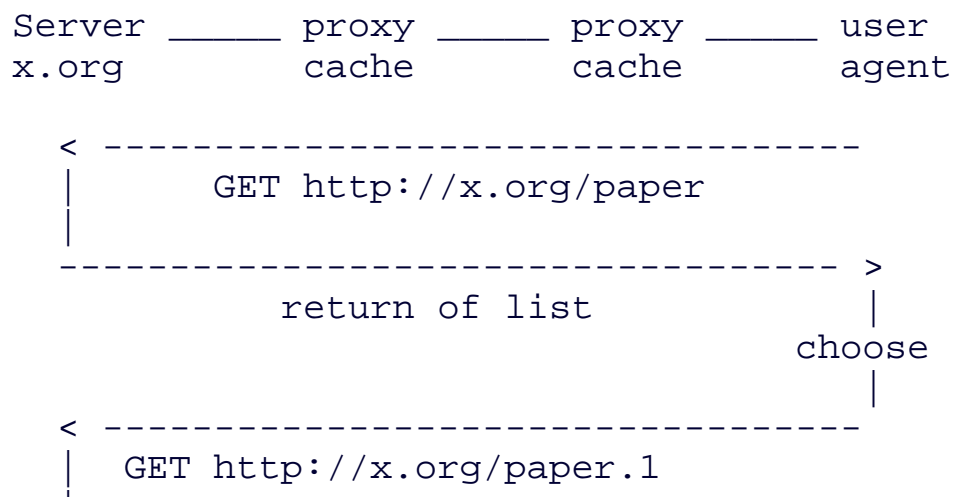
The syntax and semantics of the variant description are covered in section 5. When the user agent receives the list, it chooses the best variant and retrieves it. The transparent content negotiation can be represented as follows:

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation



```

|----->
return of paper.1

```

The first response returning the list of variants is a normal HTTP response. The second response is a normal HTTP response that does not contain special content negotiation related information. The user agent needs to know that the second response retrieves a variant. For the other parties in the second transaction is indistinguishable from the first transaction.

With this scheme, information about capabilities is only used by the user agent itself. Therefore, information in large Accept-headers is unnecessary. A limited use in transparent content negotiation. Sending of small Accept-headers can often speed up the process. This is covered in section 4.4.

List responses are covered in section 10.1. The response in the above picture could be:

```

HTTP/1.1 300 Multiple Choices
Date: Tue, 11 Jun 1996 20:02:21 GMT
TCN: list
Alternates: {"paper.1" 0.9 {type text/html}
             {"paper.2" 0.7 {type text/html}
             {"paper.3" 1.0 {type application/javascript}
             {language en}}
Vary: negotiate, accept, accept-language
ETag: "blah;1234"
Cache-control: max-age=86400
Content-Type: text/html
Content-Length: 227

```

Multiple Choices:

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiation

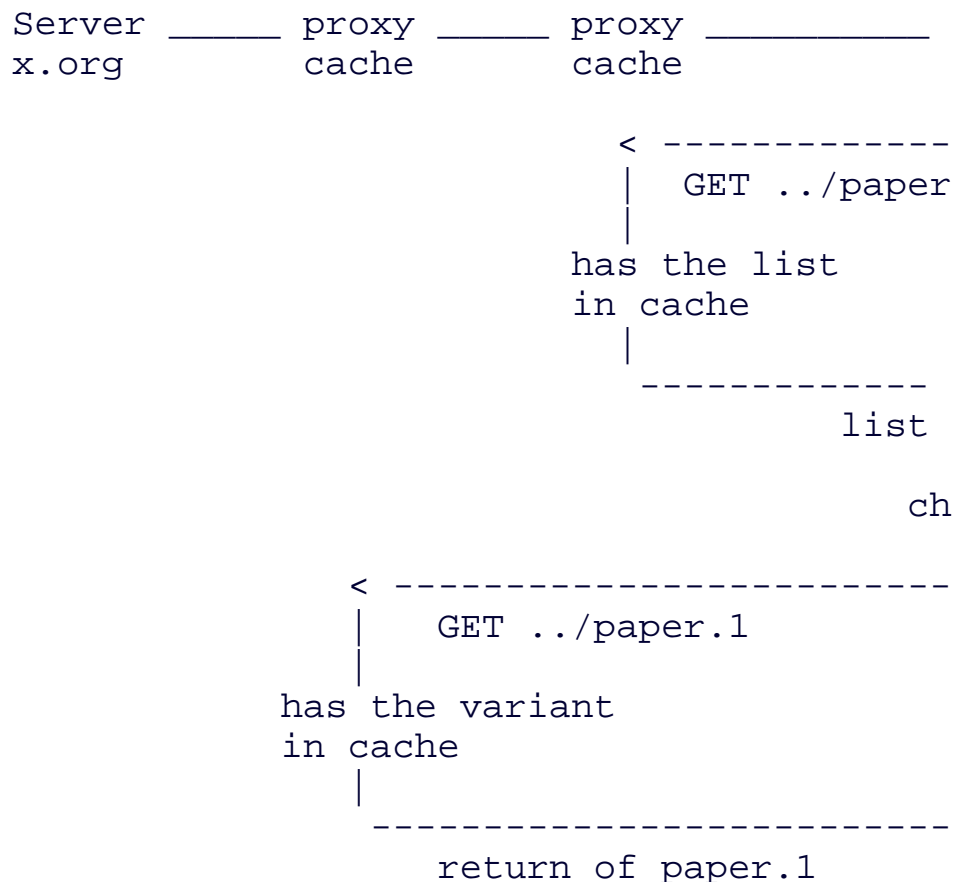
- [HTML, English version](#)
- [HTML, French version](#)
- [Postscript, English version](#)

The Alternates header in the response contain Vary header is included to ensure correct caches (see section 10.6). The ETag header a revalidated by caches, the Cache-Control head revalidation. The HTML entity included in th user to select the best variant by hand if de

4.4 Optimizing the negotiation process

The basic transparent negotiation scheme invo transactions: one to retrieve the list, and a the chosen variant. There are however severa in the data flow path of the basic scheme.

First, caching proxies can cache both variant Such caching can reduce the communication ove following example:



Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

Second, the user agent can send small Accept-headers to contain enough information to allow the server to choose a variant and return it directly.

```

Server _____ proxy _____ proxy _____ user
x.org           cache           cache           agent

```

```

< -----
|
|   GET http://x.org/paper
|   small Accept- headers
|
|   able to choose on
|   behalf of user agent
|
|----->
|
|   return of paper.1 and list

```

This choosing based on small Accept-headers variant selection algorithm'. Such an algorithm takes the variant list and the Accept-headers as input. If the Accept-headers contain sufficient information from the user agent, and if so, which variant is the best variant is a neighboring variant, it may return a choice response with the variant list, in a choice response.

A server may only choose on behalf of a user agent in transparent content negotiation if the user agent has the use of a particular remote variant selection algorithm. Negotiate request header. User agents with sophisticated variant selection algorithms may want to disable transparent content negotiation. They may want to allow it only when retrieving information from local caches. If the local algorithm of the user agent is superior to the server's algorithm in certain areas of negotiation, it is possible to enable transparent content negotiation for the easy areas only. More information about the variant selection algorithm can be found in [

Choice responses are covered in section 10.2. The choice response in the above picture could be

```

HTTP/1.1 200 OK
Date: Tue, 11 Jun 1996 20:05:31 GMT
TCN: choice
Content-Type: text/html
Last-Modified: Mon, 10 Jun 1996 10:01:14 GM
Content-Length: 5327
Cache-control: max-age=604800
Content-Location: paper.1
Alternates: {"paper.1" 0.9 {type text/html}

```

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

```

{"paper.2" 0.7 {type text/html}
{"paper.3" 1.0 {type applicatio
{language en}}
Etag: "gonkyyyy;1234"
Vary: negotiate, accept, accept-language
Expires: Thu, 01 Jan 1980 00:00:00 GMT

```

Multiple Choices for Web Stat

- [Version with HTML tables](#)
- [Version without HTML tables](#)
- [Postscript version](#)

blex

The Alternates header in the above script must be on a new line. The script always generates a list response code, which ensures compatibility with non-HTTP clients.

code, which ensures compatibility with non-IE agents.

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

21.2 Direct support by HTTP servers

Sophisticated HTTP servers could make a trans module available to content authors. Such a a remote variant selection algorithm and an i algorithm for generating choice responses (se definition of interfaces to such modules is b specification.

21.3 Web publishing tools

Web publishing tools could automatically gene a document (for example the original TeX vers tables, a HTML version without tables, and a together with an appropriate variant list in a HTTP server transparent negotiation module. documents to be published as transparently ne

22 Appendix: Example of choice response construc

The following is an example of the constructi by a proxy cache which supports HTTP/1.1 and negotiation. The use of the HTTP/1.1 conditi is also shown.

Assume that a user agent has cached a variant "1234" for the negotiable resource http://x.o that it has cached responses from two neighbo entity tags "gonkyyyy" and W/"a;b". Assume t cache entries are stale: they would need to b user agent can use them. If http://x.org/pap situation, the user agent could send the foll proxy cache:

```
GET /paper HTTP/1.1
```

```

Host: x.org
User-Agent: WuxtaWeb/2.4
Negotiate: 1.0
Accept: text/html, application/postscript;q
Accept-Language: en
If-None-Match: "gonkyyyy;1234", W/"a;b;1234

```

Assume that the proxy cache has cached the sa user agent, but that it has revalidated the v ago, so that the list is still fresh for the the proxy can run a remote variant selection and the incoming request.

Holtman & Mutz

Experimental

RFC 2295

Transparent Content Negotiat

Assume that the remote algorithm is able to c the best variant. The proxy can now construc using the algorithm in section 10.2. In step algorithm, the proxy can construct the follow on the best variant, and send it to the origi

```

GET /paper.html.en HTTP/1.1
Host: x.org
User-Agent: WuxtaWeb/2.4
Negotiate: 1.0
Accept: text/html, application/postscript;q
Accept-Language: en
If-None-Match: "gonkyyyy", W/"a;b"
Via: 1.1 fred

```

On receipt of the response

```

HTTP/1.1 304 Not Modified
Date: Tue, 11 Jun 1996 20:05:31 GMT
Etag: "gonkyyyy"

```

from the origin server, the proxy can use its paper.html.en cache entry to expand the respo response:

```

HTTP/1.1 200 OK
Date: Tue, 11 Jun 1996 20:05:31 GMT
Content-Type: text/html

```

Last-Modified: Mon, 10 Jun 1996 10:01:14 GM
Content-Length: 5327
Cache-control: max-age=604800
Etag: "gonkyyyy"
Via: 1.1 fred
Age: 0