

8149938



# THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

*August 24, 2021*

**THIS IS TO CERTIFY THAT ANNEXED IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF THE FILE WRAPPER AND CONTENTS OF:**

**APPLICATION NUMBER: 15/957,945**  
**FILING DATE: April 20, 2018**  
**PATENT NUMBER: 10257319**  
**ISSUE DATE: April 09, 2019**



Certified by

Under Secretary of Commerce  
for Intellectual Property  
and Director of the United States  
Patent and Trademark Office

Data Co Exhibit 1002  
Data Co v. Bright Data

**CERTIFICATION AND REQUEST FOR PRIORITIZED EXAMINATION  
 UNDER 37 CFR 1.102(e) (Page 1 of 1)**

First Named Inventor:	Derry Shribman	Nonprovisional Application Number (if known):	
Title of Invention:	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

**APPLICANT HEREBY CERTIFIES THE FOLLOWING AND REQUESTS PRIORITIZED EXAMINATION FOR THE ABOVE-IDENTIFIED APPLICATION.**

1. The processing fee set forth in 37 CFR 1.17(i), the prioritized examination fee set forth in 37 CFR 1.17(c), and if not already paid, the publication fee set forth in 37 CFR 1.18(d) have been filed with the request. The basic filing fee, search fee, examination fee, and any required excess claims and application size fees are filed with the request or have been already been paid.
2. The application contains or is amended to contain no more than four independent claims and no more than thirty total claims, and no multiple dependent claims.
3. The applicable box is checked below:

**I.  Original Application (Track One) - Prioritized Examination under § 1.102(e)(1)**

- i. (a) The application is an original nonprovisional utility application filed under 35 U.S.C. 111(a). This certification and request is being filed with the utility application via EFS-Web.  
 ---OR---  
 (b) The application is an original nonprovisional plant application filed under 35 U.S.C. 111(a). This certification and request is being filed with the plant application in paper.
- ii. An executed oath or declaration under 37 CFR 1.63 is filed with the application.

**II.  Request for Continued Examination - Prioritized Examination under § 1.102(e)(2)**

- i. A request for continued examination has been filed with, or prior to, this form.
- ii. If the application is a utility application, this certification and request is being filed via EFS-Web.
- iii. The application is an original nonprovisional utility application filed under 35 U.S.C. 111(a), or is a national stage entry under 35 U.S.C. 371.
- iv. This certification and request is being filed prior to the mailing of a first Office action responsive to the request for continued examination.
- v. No prior request for continued examination has been granted prioritized examination status under 37 CFR 1.102(e)(2).

Signature /Yehuda Binder/	Date 2018-04-20
Name (Print/Typed) Yehuda Binder	Practitioner Registration Number 73612
<i>Note: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required in accordance with 37 CFR 1.33 and 11.18. Please see 37 CFR 1.4(d) for the form of the signature. If necessary, submit multiple forms for more than one signature, see below*.</i>	
<input checked="" type="checkbox"/> *Total of <u>1</u> forms are submitted.	

## Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

# **SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION**

## **CROSS-REFERENCE TO RELATED APPLICATIONS**

The present application is a continuation application of U.S. non-provisional patent application no. 14/025,109, filed Sep. 12, 2013, which is a divisional application of U.S. non-provisional patent application entitled "SYSTEM AND METHOD FOR PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION" having Ser. No. 12/836,059, filed Jul. 14, 2010 and issued as U.S. Patent No. 8,560,604 on Oct. 15, 2013, and claims priority to U.S. provisional patent application entitled "SYSTEM AND METHOD FOR REDUCING INTERNET CONGESTION," having Ser. No. 61/249,624, filed Oct. 8, 2009, which are hereby incorporated herein by reference in their entirety.

## **FIELD OF THE INVENTION**

The present invention is related to Internet communication, and more particularly, to improving data communication speed and bandwidth efficiency on the Internet.

## **BACKGROUND OF THE INVENTION**

There are several trends in network and Internet usage, which tremendously increase the bandwidth that is being used on the Internet. One such trend is that more and more video is being viewed on demand on the Internet. Such viewing includes the viewing of both large and short video clips. In addition, regular shows and full-featured films may be viewed on the Internet. Another trend that is increasing the traffic on the Internet is that Web sites (such as shopping portals, news portals, and social networks) are becoming global, meaning that the Web sites are serving people in many diverse places on the globe, and thus the data is traversing over longer stretches of the Internet, increasing the congestion.

The increase in bandwidth consumption has created several major problems, a few of which are described below:

The problem for users – the current Internet bandwidth is not sufficient, and thus the effective ‘speed’ experienced by users is slow;

The problem for content owners – the tremendous amount of data being viewed by users is costing large amounts of money in hosting and bandwidth costs; and

The problem for Internet Service Providers (ISPs) – the growth in Internet traffic is requiring the ISPs to increase the infrastructure costs (communication lines, routers, etc.) at tremendous financial expense.

The need for a new method of data transfer that is fast for the consumer, cheap for the content distributor and does not require infrastructure investment for ISPs, has become a major issue which is yet unsolved.

There have been many attempts at making the Internet faster for the consumer and cheaper for the broadcaster. Each such attempt is lacking in some aspect to become a widespread, practical solution, or is a partial solution in that it solves only a subset of the major problems associated with the increase in Internet traffic. Most of the previous solutions require billions of dollars in capital investment for a comprehensive solution. Many of these attempts are lacking in that much of the content on the Internet has become dynamically created per the user and the session of the user (this is what used to be called the “Web2.0” trend). This may be seen on the Amazon Web site and the Salesforce Web site, for example, where most of the page views on these Web sites is tailored to the viewer, and is thus different for any two viewers. This dynamic information makes it impossible for most of the solutions offered to date to store the content and provide it to others seeking similar content.

One solution that has been in use is called a “proxy”. FIG. 1 is a schematic diagram providing an example of use of a proxy within a network 2. A proxy, or proxy server 4, 6, 8 is a device that is placed between one or more clients, illustrated in FIG. 1 as client devices 10, 12, 14, 16, 18, 20, that request data, via the Internet 22, and a Web server or Web servers 30, 32, 34 from which they are requesting the data. The proxy server 4, 6, 8 requests the data from the Web servers 30, 32, 34 on their behalf, and caches the responses from the Web servers 30, 32, 34, to provide to other client devices that make similar requests. If the proxy server 4, 6, 8 is geographically close enough to the client devices 10, 12, 14, 16, 18, 20, and if the storage and bandwidth of the proxy server 4, 6, 8 are large enough, the proxy server 4, 6, 8 will speed up the requests for the client devices 10, 12, 14, 16, 18, 20 that it is serving.

It should be noted, however, that to provide a comprehensive solution for Internet surfing, the proxy servers of FIG. 1 would need to be deployed at every point around the world where the Internet is being consumed, and the storage size of the proxy servers at each location would need to be near the size of all the data stored anywhere on the Internet. The abovementioned would lead to massive costs that are impractical. In addition, these proxy solutions cannot deal well with dynamic data that is prevalent now on the Web.

There have been commercial companies, such as Akamai, that have deployed such proxies locally around the world, and that are serving a select small group of sites on the Internet. If all sites on the Web were to be solved with such a solution, the capital investment would be in the range of billions of dollars. In addition, this type of solution does not handle dynamic content.

To create large distribution systems without the large hardware costs involved with a proxy solution, “peer-to-peer file sharing” solutions have been introduced, such as, for example, BitTorrent. FIG. 2 is a schematic diagram providing an example of a peer-to-peer file transfer network 50. In the network 50, files are stored on computers of consumers, referred to herein as

client devices 60. Each consumer can serve up data to other consumers, via the Internet 62, thus taking the load of serving off of the distributors and saving them the associated costs, and providing the consumer multiple points from which to download the data, referred to herein as peers 70, 72, 74, 76, 78, thus increasing the speed of the download. However, each such peer-to-peer solution must have some sort of index by which to find the required data. In typical peer-to-peer file sharing systems, because the index is on a server 80, or distributed among several servers, the number of files available in the system is not very large (otherwise, the server costs would be very large, or the lookup time would be very long).

The peer-to-peer file sharing solution is acceptable in file sharing systems, because there are not that many media files that are of interest to the mass (probably in the order of magnitude of millions of movies and songs that are of interest). Storing and maintaining an index of millions of entries is practical technically and economically. However, if this system were to be used to serve the hundreds of billions of files that are available on the Internet of today, the cost of storing and maintaining such an index would be again in the billions of dollars. In addition, these types of peer-to-peer file sharing systems are not able to deal with dynamic HTTP data.

In conclusion, a system does not exist that enables fast transmission of most of the data on the Internet, that does not incur tremendous costs, and/or that provides only a very partial solution to the problem of Internet traffic congestion. Thus, a heretofore unaddressed need exists in the industry to address the aforementioned deficiencies and inadequacies.

#### **SUMMARY OF THE INVENTION**

The present system and method provides for faster and more efficient data communication within a communication network. Briefly described, in architecture, one embodiment of the system, among others, can be implemented as follows. A network is provided

for accelerating data communication, wherein the network contains: at least one client communication device for originating a data request for obtaining the data from a data server; at least one agent communication device which is assigned to the data server for receiving the data request from the client communication device, wherein the agent keeps track of which client communication devices have received responses to data requests from the assigned data server; at least one peer communication device for storing portions of data received in response to the data request by the at least one client communication device, wherein the portions of data may be transmitted to the at least one client communication device upon request by the client communication device; and at least one acceleration server for deciding which agent communication device is to be assigned to which data server and providing this information to the at least one client communication device.

The present system and method also provides a communication device within a network, wherein the communication device contains: a memory; and a processor configured by the memory to perform the steps of: originating a data request for obtaining data from a data server; being assigned to a data server, referred to as an assigned data server; receiving a data request from a separate device within the network, and keeping track of which client communication devices within the network have received responses to data requests from the assigned data server; and storing portions of data received in response to the originated data request, wherein the portions of data may be transmitted to communication device upon request by the communication device.

Other systems, methods, features, and advantages of the present invention will be or become apparent to one with skill in the art upon examination of the following drawings and detailed description. It is intended that all such additional systems, methods, features, and advantages be included within this description, be within the scope of the present invention, and be protected by the accompanying claims.



### **BRIEF DESCRIPTION OF THE DRAWINGS**

Many aspects of the invention can be better understood with reference to the following drawings. The components in the drawings are not necessarily to scale, emphasis instead being placed upon clearly illustrating the principles of the present invention. Moreover, in the drawings, like reference numerals designate corresponding parts throughout the several views.

FIG. 1 is a schematic diagram providing a prior art example of use of a proxy within a network.

FIG. 2 is a schematic diagram providing a prior art example of a peer-to-peer file transfer network.

FIG. 3 is a schematic diagram providing an example of a communication network in accordance with the present invention.

FIG. 4 is a schematic diagram further illustrating a communication device of the communication network of FIG. 3.

FIG. 5 is a schematic diagram further illustrating the memory of FIG. 4.

FIG. 6 is a schematic diagram further illustrating elements of the acceleration application of FIG. 5, as well as communication paths of the acceleration application.

FIG. 7 is a chart further illustrating two of the main databases utilized within the communication network.

FIG. 8 is a flowchart illustrating operation of the acceleration system initializer module.

FIG. 9 is a flowchart further illustrating communication between different elements of the communication network.

FIG. 10 is a flowchart continuing the flowchart of FIG. 9 and focused on agent response to the HTTP request.

FIG. 11 is a flowchart continuing the flowchart of FIG. 10, which illustrates actions taken upon receipt of the list of peers, or single peer listing, from the agent.

FIG. 12 is a flowchart illustrating steps taken by an agent, client, or peer to determine whether a certain HTTP request is still valid.

FIG. 13 is a flowchart outlining operation of the acceleration server.

FIG. 14 is a flowchart further illustrating TCPIP acceleration in accordance with an alternative embodiment of the invention.

FIG. 15 is a flowchart further illustrating TCPIP acceleration in accordance with an alternative embodiment of the invention, detailing the communication between the client and the TCPIP server (read and write commands) after the connect phase has completed successfully.

#### **DETAILED DESCRIPTION**

The present system and method provides for faster and more efficient data communication within a communication network. An example of such a communication network 100 is provided by the schematic diagram of FIG. 3. The network 100 of FIG. 3 contains multiple communication devices. Due to functionality provided by software stored within each communication device, which may be the same in each communication device, each communication device may serve as a client, peer, or agent, depending upon requirements of the network 100, as is described in detail herein. It should be noted that a detailed description of a communication device is provided with regard to the description of FIG. 4.

Returning to FIG. 3, the exemplary embodiment of the network 100 illustrates that one of the communication devices is functioning as a client 102. The client 102 is capable of communication with one or more peers 112, 114, 116 and one or more agents 122. For exemplary purposes, the network contains three peers and one agent, although it is noted that a client can communicate with any number of agents and peers.

The communication network 100 also contains a Web server 152. The Web server 152 is the server from which the client 102 is requesting information and may be, for example, a typical HTTP server, such as those being used to deliver content on any of the many such servers on the Internet. It should be noted that the server 152 is not limited to being an HTTP server. In fact, if a different communication protocol is used within the communication network, the server may be a server capable of handling a different protocol. It should also be noted that while the present description refers to the use of HTTP, the present invention may relate to any other communication protocol and HTTP is not intended to be a limitation to the present invention.

The communication network 100 further contains an acceleration server 162 having an acceleration server storage device 164. As is described in more detail herein, the acceleration server storage device 164 has contained therein an acceleration server database. The acceleration server database stores Internet protocol (IP) addresses of communication devices within the communication network 100 having acceleration software stored therein. Specifically, the acceleration server database contains stored therein a list of communication devices having acceleration software stored therein that are currently online within the communication network 100. For each such agent, the acceleration server assigns a list of IP addresses.

In the communication network 100 of FIG. 3, the application in the client 102 is requesting information from the Web server 152, which is why the software within the communication device designated this communication device to work as a client. In addition, since the agent 122 receives the request from the client 102 as the communication device closest

to the Web server 152, functionality of the agent 122, as provided by the software of the agent 122, designates this communication device to work as an agent. It should be noted, that in accordance with an alternative embodiment of the invention, the agent need not be the communication device that is closest to the Web server. Instead, a different communication device may be selected to be the agent.

Since the peers 112, 114, 116 contain at least portions of the information sought by the client 102 from the Web server 152, functionality of the peers 112, 114, 116, as provided by the software of the peers 112, 114, 116, designates these communication devices to work as peers. It should be noted that the process of designating clients, agents, and peers is described in detail herein. It should also be noted that the number of clients, agents, peers, acceleration servers, Web servers, and other components of the communication network 100 may differ from the number illustrated by FIG. 3. In fact, the number of clients, agents, peers, acceleration servers, Web servers, and other components of the communication network 100 are not intended to be limited by the current description.

Prior to describing functionality performed within a communication network 100, the following further describes a communication device 200, in accordance with a first exemplary embodiment of the invention. FIG. 4 is a schematic diagram further illustrating a communication device 200 of the communication network 100, which contains general components of a computer. As previously mentioned, it should be noted that the communication device 200 of FIG. 4 may serve as a client, agent, or peer.

Generally, in terms of hardware architecture, as shown in FIG. 4, the communication device 200 includes a processor 202, memory 210, at least one storage device 208, and one or more input and/or output (I/O) devices 240 (or peripherals) that are communicatively coupled via a local interface 250. The local interface 250 can be, for example but not limited to, one or more buses or other wired or wireless connections, as is known in the art. The local interface 250

may have additional elements, which are omitted for simplicity, such as controllers, buffers (caches), drivers, repeaters, and receivers, to enable communications. Further, the local interface 250 may include address, control, and/or data connections to enable appropriate communications among the aforementioned components.

The processor 202 is a hardware device for executing software, particularly that stored in the memory 210. The processor 52 can be any custom made or commercially available processor, a central processing unit (CPU), an auxiliary processor among several processors associated with the communication device 200, a semiconductor based microprocessor (in the form of a microchip or chip set), a macroprocessor, or generally any device for executing software instructions.

The memory 210, which is further illustrated and described by the description of FIG. 5, can include any one or combination of volatile memory elements (*e.g.*, random access memory (RAM, such as DRAM, SRAM, SDRAM, *etc.*)) and nonvolatile memory elements (*e.g.*, ROM, hard drive, tape, CDROM, *etc.*). Moreover, the memory 210 may incorporate electronic, magnetic, optical, and/or other types of storage media. Note that the memory 210 can have a distributed architecture, where various components are situated remote from one another, but can be accessed by the processor 202.

The software 212 located within the memory 210 may include one or more separate programs, each of which contains an ordered listing of executable instructions for implementing logical functions of the communication device 200, as described below. In the example of FIG. 4, the software 212 in the memory 210 at least contains an acceleration application 220 and an Internet browser 214. In addition, the memory 210 may contain an operating system (O/S) 230. The operating system 230 essentially controls the execution of computer programs and provides scheduling, input-output control, file and data management, memory management, and communication control and related services. It should be noted that, in addition to the

acceleration application 220, Internet browser 214, and operating system 230, the memory 210 may contain other software applications.

While the present description refers to a request from the client originating from an Internet browser, the present invention is not limited to requests originating from Internet browsers. Instead, a request may originate from an email program or any other program that would be used to request data that is stored on a Web server, or other server holding data that is requested by the client device.

Functionality of the communication device 200 may be provided by a source program, executable program (object code), script, or any other entity containing a set of instructions to be performed. When a source program, then the program needs to be translated via a compiler, assembler, interpreter, or the like, which may or may not be included within the memory 210, so as to operate properly in connection with the operating system 230. Furthermore, functionality of the communication device 200 can be written as (a) an object oriented programming language, which has classes of data and methods, or (b) a procedure programming language, which has routines, subroutines, and/or functions.

The I/O devices 240 may include input devices, for example but not limited to, a keyboard, mouse, scanner, microphone, *etc.* Furthermore, the I/O devices 240 may also include output devices, for example but not limited to, a printer, display, *etc.* Finally, the I/O devices 240 may further include devices that communicate via both inputs and outputs, for instance but not limited to, a modulator/demodulator (modem; for accessing another device, system, or network), a radio frequency (RF) or other transceiver, a telephonic interface, a bridge, a router, *etc.*

When the communication device 200 is in operation, the processor 202 is configured to execute the software 212 stored within the memory 210, to communicate data to and from the memory 210, and to generally control operations of the communication device 200 pursuant to

the software 212. The software 212 and the O/S 230, in whole or in part, but typically the latter, are read by the processor 202, perhaps buffered within the processor 202, and then executed.

When functionality of the communication device 200 is implemented in software, as is shown in FIG. 4, it should be noted that the functionality can be stored on any computer readable medium for use by or in connection with any computer related system or method. In the context of this document, a computer readable medium is an electronic, magnetic, optical, or other physical device or means that can contain or store a computer program for use by or in connection with a computer related system or method. The functionality of the communication device 200 can be embodied in any computer-readable medium for use by or in connection with an instruction execution system, apparatus, or device, such as a computer-based system, processor-containing system, or other system that can fetch the instructions from the instruction execution system, apparatus, or device and execute the instructions. In the context of this document, a "computer-readable medium" can be any means that can store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

The computer readable medium can be, for example but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, device, or propagation medium. More specific examples (a non-exhaustive list) of the computer-readable medium would include the following: an electrical connection (electronic) having one or more wires, a portable computer diskette (magnetic), a random access memory (RAM) (electronic), a read-only memory (ROM) (electronic), an erasable programmable read-only memory (EPROM, EEPROM, or Flash memory) (electronic), an optical fiber (optical), and a portable compact disc read-only memory (CDROM) (optical). Note that the computer-readable medium could even be paper or another suitable medium upon which the program is printed, as the program can be electronically captured, via for instance optical scanning of the paper or other medium, then

compiled, interpreted or otherwise processed in a suitable manner if necessary, and then stored in a computer memory.

In an alternative embodiment, where the functionality of the communication device 200 is implemented in hardware, the functionality can be implemented with any or a combination of the following technologies, which are each well known in the art: a discrete logic circuit(s) having logic gates for implementing logic functions upon data signals, an application specific integrated circuit (ASIC) having appropriate combinational logic gates, a programmable gate array(s) (PGA), a field programmable gate array (FPGA), *etc.*

The at least one storage device 208 of the communication device 200 may be one of many different categories of storage device. As is described in more detail herein, the storage device 208 may include a configuration database 280 and a cache database 282. Alternatively, the configuration database 280 and cache database 282 may be located on different storage devices that are in communication with the communication device 200. The description that follows assumes that the configuration database 280 and cache database 282 are located on the same storage device, however, it should be noted that the present invention is not intended to be limited to this configuration.

The configuration database 280 stores configuration data that is common to all elements of the communication network 100 and is used to provide set up and synchronization information to different modules of the acceleration application 220 stored within the memory 210, as is described in further detail herein. The cache database 282 stores responses to HTTP requests that the communication device 200 has dispatched, either for its own consumption or on behalf of other elements of the communication network 100. As is explained in additional detail herein, the responses to HTTP requests are stored within the cache database 282 for future use by this communication device 200, or for other communication devices within the



communication network 100 that need to retrieve this information and will use this communication device as either a peer or an agent.

In addition to the abovementioned, as is explained in further detail herein, the cache database 282 has stored therein a list of URLs that the communication device is aware of (i.e., has seen requests for). For each URL, the cache database 282 has stored therein the URL itself, HTTP headers returned by the Web Server for this URL, when the last time was that the contents of this URL was loaded directly from the Web Server, when the contents of the URL had last changed on the Web Server, as well as a list of chunks that contain the contents of this URL, and the chunks of data themselves. Chunks in the present description are defined as equally sized pieces of data that together form the whole content of the URL. It should be noted that while the present description provides for chunks being equally sized pieces of data, in accordance with an alternative embodiment of the invention, the chunks may instead be of different size.

FIG. 5 is a schematic diagram further illustrating the memory 210 of FIG. 4. As shown by FIG. 5, the memory 210 may be separated into two basic levels, namely, an operating system level 260 and an application level 270. The operating system level 260 contains the operating system 230, wherein the operating system 230 further contains at least one device driver 262 and at least one communication stack 264. The device drivers 262 are software modules that are responsible for the basic operating commands for various hardware devices of the communication device 200, such as the processor 202, the storage device 208 and the I/O devices 240. In addition, the communication stacks 264 provide applications of the communication device 200 with a means of communicating within the network 100 by implementing various standard communication protocols.

The application level 270 includes any application that is running on the communication device 200. As a result, the application level 270 includes the Internet browser 214, which is used to view information that is located on remote Web servers, the acceleration application 220,

as described in more detail below, and any other applications 216 stored on the communication device 200.

As is explained in additional detail below, the acceleration application 220 intercepts the requests being made by applications of the communication device (client) that use the Internet, in order to modify the requests and route the requests through the communication network. There are various methods that may be used to intercept such requests. One such method is to create an intermediate driver 272, which is also located within the memory 210, that attaches itself to all communication applications, intercepts outgoing requests of the communication applications of the communication device 200, such as the Internet browser 214, and routes the requests to the acceleration application 220. Once the acceleration application 220 modifies the requests, routes the requests to other system elements on the communication network 100, and receives replies from other system elements of the communication network 100, the acceleration application 220 returns the replies to the intermediate driver 272, which provides the replies back to the requesting communication application.

FIG. 6 is a schematic diagram further illustrating elements of the acceleration application 220, as well as communication paths of the acceleration application 220. The acceleration application 220 contains an acceleration system initializer module 222, which is called when the acceleration application 220 is started. The acceleration system initializer module 222 is capable of initializing all elements of the communication device 200. The acceleration application 220 also contains three separate modules that run in parallel, namely, a client module 224, a peer module 226, and an agent module 228, each of which comes into play according to the specific role that the communication device 200 is partaking in the communication network 100 at a given time. The role of each module is further described herein.

The client module 224 provides functionality required when the communication device 200 is requesting information from the Web server 152, such as, for example, but not limited to,

Web pages, data, video, or audio. The client module 224 causes the communication device 200 having the client module 224 therein to intercept the information request and pass the information request on to other elements of the communication network 100, such as, servers, agents or peers. This process is further described in detail herein.

The peer module 226 provides functionality required by the communication device 200 when answering other clients within the communication network 100 and providing the other clients with information that they request, which this communication device 200, having this peer module 226 therein, has already downloaded at a separate time. This process is further described in detail herein.

The agent module 228 provides functionality required when other communication devices of the communication network 100 acting as clients query this communication device 200, having this agent module 228 therein, as an agent, to obtain a list of peers within the communication network 100 that contain requested information. This process is further described in detail herein.

The acceleration application 220 interacts with both the configuration database 280 and the cache database 282 of the storage device 208. As previously mentioned herein, the configuration database 280 stores configuration data that may be common to all communication devices of the communication network 100 and is used to provide setup and synchronization information to different modules 222, 224, 226, 228 of the acceleration application 220 stored within the memory 210.

The cache database 282 stores responses to information requests, such as, for example, HTTP requests, that the communication device 200 has dispatched, either for its own consumption or on behalf of other elements of the communication network 100. The responses to HTTP requests are stored within the cache database 282 for future use by this communication device 200, or for other communication devices within the communication network 100 that

need to retrieve this same information and will use this communication device 200 as either a peer or an agent. This process is described in detail herein.

Information stored within the cache database 282 may include any information associated with a request sent by the client. As an example, such information may include, metadata and actual requested data. For example, for an HTTP request for a video, the metadata may include the version of the Web server answering the request from the client and the data would be the requested video itself. In a situation where there is no more room for storage in the cache database, the software of the associated communication device may cause the communication device to erase previous data stored in order to clear room for the new data to store in the cache database. As an example, such previous data may include data that is most likely not to be used again. Such data may be old data or data that is known to no longer be valid. The communication device may choose to erase the least relevant data, according to any of several methods that are well known in the art.

FIG. 7 is a chart further illustrating two of the main databases utilized within the communication network 100, namely, the acceleration server database 164 and the cache database 282. As previously mentioned, the acceleration server database 164 stores IP addresses of communication devices located within the communication network 100, which have acceleration software stored therein. Specifically, the acceleration server database 164 contains stored therein a list of communication devices having acceleration software stored therein that are currently online within the communication network 100. The acceleration server assigns a list of IP addresses to each communication device functioning as an agent. Each communication device will be the agent for any Web servers whose IP address is in the range 'owned' by that communication device. As an example, when a first ever communication device goes online, namely, the first communication device as described herein having the acceleration application 220 therein, the acceleration server assigns all IP addresses in the world to this communication device, and this communication device will be the agent for any Web server. When a second

communication device goes online it will share the IP address list with the first communication device, so that each of the communication devices will be responsible for a different part of the world wide web servers.

The cache database 282 of the communication device 200 has stored therein a list of URLs 286 of which the communication device 200 is aware. The communication device 200 becomes aware of a URL each time that the communication device 200 receives a request for information located at a specific URL. As shown by FIG. 7, for each URL 288 within the list of URLs 286, the cache database 282 stores: the URL itself 290; HTTP headers 292 returned by the Web Server 152 for this URL; when the last time 294 was that the contents of this URL were loaded directly from the Web Server 152; when the contents of the URL last changed 296 on the Web Server 152; and a list of chunks 298 that contain the contents of this URL, and the content of the chunk. As previously mentioned, chunks, in the present description, are defined as equally sized pieces of data that together form the entire content of the URL, namely, the entire content whose location is described by the URL. As a non-limiting example, a chunk size of, for example, 16KB can be used, so that any HTTP response will be split up into chunks of 16KB. In accordance with an alternative embodiment of the invention, if the last chunk of the response is not large enough to fill the designated chunk size, such as 16KB for the present example, the remaining portion of the chunk will be left empty.

For each such chunk 300, the cache database 282 includes the checksum of the chunk 302, the data of the chunk 304 itself, and a list of peers 306 that most likely have the data for this chunk. As is described in additional detail herein, the data for the chunk may be used by other clients within the communication network 100 when other communication devices of the communication network 100 serve as peers to the clients, from which to download the chunk data.

For each chunk, a checksum is calculated and stored along side of the chunk itself. The checksum may be calculated in any of numerous ways known to those in the art. The purpose of having the checksum is to be able to identify data uniquely, whereas the checksum is the “key” to the data, where the data is the chunk. As an example, a client may want to load the contents of a URL, resulting in the agent that is servicing this request sending the checksums of the chunks to the client, along with the peers that store these chunks. It is to be noted that there could be a different peer for every different chunk. The client then communicates with each such peer, and provides the checksum of the chunk that it would like the peer to transmit back to the client. The peer looks up the checksum (the key) in its cache database, and provides back the chunk (data) that corresponds to this checksum (the key). As shown by FIG. 7, for each peer 308 within the list of peers 306, the cache database 282 includes the peer IP address 310, as well as the connection status 312 of the peer, which represents whether the peer 308 is online or not.

In accordance with one embodiment of the invention, the cache database 282 may be indexed by URL and by Checksum. Having the cache database indexed in this manner is beneficial due to the following reason. When the agent is using the cache database, the agent receives a request from a client for the URL that the client is looking for. In such a case the agent needs the cache database to be indexed by the URL, to assist in finding a list of corresponding peers that have the chunks of this URL. When the peers are using this cache database, the peers obtain a request from the client for a particular checksum, and the peers need the database to be indexed by the checksum so that they can quickly find the correct chunk. Of course, as would be understood by one having ordinary skill in the art, the cache database may instead be indexed in any other manner.

Having described components of the communication network 100, the following further describes how such components interact and individually function. FIG. 8 is a flowchart 300 illustrating operation of the acceleration system initializer module 222 (hereafter referred to as the initializer 222 for purposes of brevity). It should be noted that any process descriptions or

blocks in flowcharts should be understood as representing modules, segments, portions of code, or steps that include one or more instructions for implementing specific logical functions in the process, and alternative implementations are included within the scope of the present invention in which functions may be executed out of order from that shown or discussed, including substantially concurrently or in reverse order, depending on the functionality involved, as would be understood by those reasonably skilled in the art of the present invention.

The initializer 222 is the first element of the communication device 200 to operate as the communication device 200 starts up (block 302). As the initializer 222 starts, it first communicates with the acceleration server 162 to sign up with the acceleration server 162. This is performed by providing the acceleration server 162 with the hostname, and all IP addresses and media access control (MAC) addresses of the interfaces on the communication device 200 having the initializer 222 thereon.

In accordance with an alternative embodiment of the invention, as shown by block 304, the initializer 222 checks with the acceleration server 162 whether a more updated version of the acceleration application software is available. This may be performed by any one of many known methods, such as, but not limited to, by providing the version number of the acceleration application software to the acceleration server 162. The message received back from the acceleration server 162 indicates whether there is a newer version of the acceleration application software or not. If a newer version of the acceleration application software exists, the initializer 222 downloads the latest version of the acceleration application software from the acceleration server 162, or from a different location, and installs the latest version on the communication device 200. In addition to the abovementioned, the initializer 222 may also schedule additional version checks for every set period of time thereafter. As an example, the initializer 222 may check for system updates every two days.

As shown by block 306, the initializer 222 then redirects outgoing network traffic from the communication device 200 to flow through the acceleration application 162. As previously mentioned, one way to redirect the outgoing network traffic is to insert an intermediate driver 212 that intercepts and redirects the traffic. It should be noted that there are many other ways to implement this redirection, which are well known to those having ordinary skill in the art.

As shown by block 308, the initializer 222 then launches the client module 224 of the communication device 200, and configures the client module 224 of the communication device 200 to intercept to all outgoing network communications of the communication device 200 and route the outgoing network communications to the client module 224, from the intermediate driver 272 or other routing method implemented. This is performed so that the client module 224 is able to receive all network traffic coming from the network applications, modify the network traffic if necessary, and re-route the traffic. As is known by those having ordinary skill in the art, in order to re-route the traffic, the traffic needs to be modified, as an example, to change the destination of requests.

As shown by block 310, the initializer 222 then launches the agent module 228 and the peer module 226 to run on the communication device 200. The agent module 228 and peer module 226 listen on pre-determined ports of the communication device 200, so that incoming network traffic on these ports gets routed to the agent module 228 and peer module 226. As is explained in further detail herein, the abovementioned enables the communication device 200 to function as an agent and as a peer for other communication devices within the communication network 100, as needed.

FIG. 9 is a flowchart 350 further illustrating communication between different elements of the communication network 100, in accordance with the present system and method for providing faster and more efficient data communication.



As shown by block 352, an application running on the client 200 initiates a request for a resource on a network. Such a request may be, for example, “GET http://www.aol.com/index.html HTTP/1.1”. The request may come from an Internet browser 214 located on the client 200, where the Internet browser 214 is loading a page from the Internet, an application that wants to download information from the Internet, fetch or send email, or any other network communication request.

Through the intermediate driver 272, or other such mechanism as may be implemented that is re-routing the communication to the client module 224 of the client 200, the resource request is intercepted by the client module 224 that is running on the client 200 (block 354). The client module 224 then looks up the IP address of the server 152 that is the target of the resource request (e.g., the IP address of the Web server that is the host of www.aol.com in the example above), and sends this IP address to the acceleration server 162 (block 356) in order to obtain a list of communication devices that the client 200 can use as agents (hereafter referred to as agents). It should be noted that the process of performing an IP lookup for a server is known by one having ordinary skill in the art, and therefore is not described further herein.

In response to receiving the IP address of the server 152, the acceleration server 162 prepares a list of agents that may be suitable to handle the request from this IP address (block 358). The size of the list can differ based on implementation. For exemplary purposes, the following provides an example where a list of five agents is prepared by the acceleration server 162. The list of agents is created by the acceleration server 162 by finding the communication devices of the communication network 100 that are currently online, and whose IP address is numerically close to the IP of the destination Web server 152. A further description of the abovementioned process is described here in.

As shown by block 360, the client module 224 then sends the original request (e.g., “GET http://www.aol.com/index.html HTTP/1.1”) to all the agents in the list received from the

acceleration server 162 in order to find out which of the agents in the list is best suited to be the one agent that will assist with this request.

It should be noted that, in accordance with an alternative embodiment of the invention, the communication device 200 may be connected to a device that is actually requesting data. In such an alternative embodiment, the communication device would be a modular device connected to a requesting device, where the requesting device, such as, for example, a personal data assistant (PDA) or other device, would request data, and the communication device connected thereto, either through a physical connection, wireless connection, or any other connection, would receive the data request and function as described herein. In addition, as previously mentioned, it should be noted that the HTTP request may be replaced by any request for resources on the Web.

FIG. 10 is a flowchart continuing the flowchart 380 of FIG. 9 and focused on agent response to the request. As shown by block 382, upon receiving the request from the client 200, each agent that received the request from the client responds to the client 200 with whether it has information regarding the request, which can help the client to download the requested information from peers in the network. Specifically, each agent responds with whether the agent has seen a previous request for this resource that has been fulfilled. In such a case, the agent may then provide the client with the list of peers and checksums of the chunks that each of them have.

As shown by block 384, the client then decides which of the agents in the list to use as its agent for this particular information request. To determine which agent in the list to use as its agent for the particular information request, the client may consider multiple factors, such as, for example, factoring the speed of the reply by each agent and whether that agent does or does not have the information required. There are multiple ways to implement this agent selection, one practical way being to start a timer of a small window of time, such as, for example, 5ms, after receiving the first response from the agents, and after the small window, choosing from the list

of agents that responded, the agent that has the information about the request, or in the case that none of the agents responded, to choose the first agent from the list received from the acceleration server 162.

As shown by block 386, after selecting an agent, the client notifies the selected agent that it is going to use it for this request, and notifies the other agents that they will not be used for this request. The client then sends the selected agent a request for the first five chunks of data of the original information request (block 388). By specifying to the selected agent the requested chunks by their order in the full response, the client receives the peer list and checksums of the requested chunks from the selected agent. As an example, for the first five chunks the client will ask the selected agent for chunks one through five, and for the fourth batch of five chunks the client will ask the agent for chunks sixteen through twenty. As previously mentioned, additional or fewer chunks may be requested at a single time.

As shown by block 390, after receiving the request from the client, the selected agent determines whether it has information regarding the requested chunks of data by looking up the request in its cache database and determining if the selected agent has stored therein information regarding peers of the communication network that have stored the requested data of the request, or whether the selected agent itself has the requested data of the request stored in its memory. In addition to determining if the selected agent contains an entry for this request in its database, the selected agent may also determine if this information is still valid. Specifically, the selected agent determines whether the data that is stored within the memory of the selected agent or the memory of the peers, still mirrors the information that would have been received from the server itself for this request. A further description of the process utilized by the selected agent to determine if the information is still valid, is described in detail herein.

As shown by block 392, if the information (requested data of the request) exists and is still valid, then the agent prepares a response to the client, which includes for each of the chunks:

(i) the checksum of the chunk; (ii) a list of peers that according to the database of the selected agent contains these chunks; and (iii) if these are the first five chunks of the information, then the selected agent also provides the specific protocol's headers that would have been received from the server, had the initial request from the client been made directly to the server.

As shown by block 394, the list of peers for each chunk is sorted by geographical proximity to the requesting client. In accordance with the present example, only the five closest peers are kept in the list for every chunk, and the rest of the peers are discarded from this list. As shown by block 396, the prepared response, namely, the list of closest peers, is sent back to the client. It should be noted that, if this were the last set of chunks to be provided for this request, then it would be beneficial to include information about this to the client.

If the selected agent discovers that it does not have information about this request, or if the selected agent discovers that the information it has is no longer valid, the selected agent needs to load the information directly from the server in order to be able to provide an answer to the requesting client. As shown by block 400, the selected agent then sends the request directly to the server. The selected agent then stores the information it receives from the server (both the headers of the request, as well as chunks of the response itself) in its database, for this particular response to the client, as well as for future use to other clients that may request this data (block 402). The selected agent then prepares a response (list) for the client, where the response includes the protocol headers (if these are the first five chunks), and the checksums of the five chunks, and provides itself as the only peer for these chunks (block 404). This list is then sent back to the client (block 406).

FIG. 11 is a flowchart 420 continuing the flowchart of FIG. 10, which illustrates actions taken upon receipt of the list of peers, or single peer listing, from the agent. As shown by block 422, the client receives the response from the agent (including the list of chunks and their corresponding data, including peers and other information previously mentioned) and, for each

of the five chunks, the client sends a request to each of the peers listed for the chunk to download the chunk. The chunk request that the client sends to each of the peers is the checksum of the data that the client seeks to receive, which is the key (identifier) of the chunk.

As shown by block 424, the peers then respond regarding whether they still have the data of the chunk. As an example, some of the peers may not currently be online, some may be online but may have discarded the relevant information, and some may still have the relevant information, namely, the chunk. As shown by block 426, the client then selects the quickest peer that responds with a positive answer regarding the requested information, the client lets that peer know that it is chosen to provide the client with the chunk, and the client notifies the other peers that they are not chosen.

As shown by block 428, the chosen peer then sends the chunk to the client. It should be noted that if no peers answer the request of the client, the client goes back to the agent noting that the peers were all negative, and the agent either provides a list of 5 other agents, if they exist, or the agent goes on to download the information directly from the Web server as happens in the case where no peers exist as described above.

The client then stores the chunks in its cache for future use (block 430), when the client may need to provide the chunks to a requesting communication device when acting as a peer for another client that is looking for the same information. As shown by block 432, if some of the chunks were not loaded from any of the peers, the client requests the chunks again from the agent in a next round of requests, flagging these chunks as chunks that were not loadable from the client list of peers. In this situation, the agent will load the data directly from the server and provide it back to the client.

The client then acknowledges to the agent which of the chunks it received properly (block 434). The agent then looks up these chunks in the database of the agent, and adds the

client to the list of peers for these chunks, specifically, since this client is now storing these chunks, and can provide these chunks to other clients that turn to it as a peer (block 436).

As shown by block 438, the client then passes the data on to the Web browser or other application of the client that made the original request, for it to use as it had originally intended. The client then checks whether all of the chunks for this request were received (block 440), by checking the flag set by the agent. Specifically, when the agent is providing the list of the last 5 chunks, the agent includes that information as part of its reply to the client, which is referred to herein as a flag. This information is what enables the client to know that all information has been received for a particular resource request.

If the last received chunks were not the last chunks for this request, the processing flow of the client continues by returning to the functionality of block 384 of FIG. 10, but instead sending the chosen agent a request for the next five chunks of data of the original information request. Alternatively, if all chunks for this request were received, the request is complete, and the flow starts again at block 352 of FIG. 9.

FIG. 12 is a flowchart 500 illustrating steps taken by an agent, client, or peer to determine whether a certain HTTP request is still valid. Specifically, the following provides an example of how the agent, client, or peer can determine whether particular data that is stored within the memory of the agent, or the memory of a peer or client, still mirrors the information that is currently on the Web server. As shown by block 502, the HTTP request is looked up in the cache database of the agent, client or peer that is checking the validity of the HTTP request. As an example, the HTTP protocol, defined by RFC 2616, outlines specific methods that Web servers can define within the HTTP headers signifying the validity of certain data, such as, but not limited to, by using HTTP header information such as “max age” to indicate how long this data may be cached before becoming invalid, “no cache” to indicate that the data may never be cached, and using other information.

As shown by block 504, these standard methods of validation are tested on the HTTP request information in question. As shown by block 506, a determination is made whether the requested information that is stored is valid or not. If the requested information is valid, a “VALID” response is returned (block 508). Alternatively, if the requested information is not valid, an HTTP conditional request is sent to the relevant Web server, to determine if the data stored for this request is still valid (block 510). If the data stored for this request is still valid, a “VALID” response is returned (block 508). Alternatively, if the data stored for this request is not valid, an “INVALID” response is returned (block 514). It should be noted, that the abovementioned description with regard to FIG. 12 is an explanation of how to check if HTTP information is still valid. There are similar methods of determining validity for any other protocol, which may be utilized, and which those having ordinary skill in the art would appreciate and understand.

FIG. 13 is a flowchart 550 outlining operation of the acceleration server, whose main responsibility in the present system and method is to provide clients with information regarding which agents serve which requests, and to keep the network elements all up to date with the latest software updates. As shown by block 552, the acceleration server sends “keep alive” signals to the network elements, and keeps track within its database as to which network elements are online. As shown by block 554, the acceleration server continues to wait for a client request and continues to determine if one is received.

Once a request is received, the acceleration server tests the type of request received (block 556). If the client request is to sign up the client within the network, an event that happens every time that the client starts running on its host machine, then that client is added to the list of agents stored on the acceleration server, sorted by the IP address of the client (block 558).

If the request is to find an agent to use for a particular request, the acceleration server creates a new agent list, which is empty (block 560). The acceleration server then searches the

agent database for the next 5 active agents whose IP address is closest to the IP address of the server who is targeted in the request (block 562). In this context, 192.166.3.103 is closer to 192.166.3.212 than to 192.167.3.104. The acceleration server then sends this agent list to the client (block 564).

If instead, the request is to check the version of the latest acceleration software then the acceleration server sends that network element (client, peer or agent) the version number of the latest existing acceleration software version, and a URL from where to download the new version, for the case that the element needs to upgrade to the new version (block 566).

While the abovementioned example is focused on HTTP requests for data, as previously mentioned, other protocol requests are equally capable of being handled by the present system and method. As an example, in separate embodiments the acceleration method described may accelerate any communication protocol at any OSI layer (SMTP, DNS, UDP, ETHERNET, etc.). In the following alternative embodiment, it is illustrated how the acceleration method may accelerate TCPIP. As is known by those having ordinary skill in the art, TCPIP is a relatively low-level protocol, as opposed to HTTP, which is a high level protocol. For purposes of illustration of TCPIP communication, reference may be made to FIG. 3, wherein the Web server is a TCPIP server.

In TCPIP there are three communication commands that are of particular interest, namely, connect, write, and read. Connect is a command issued by an application in the communication device that is initiating the communication to instruct the TCPIP stack to connect to a remote communication device. The connect message includes the IP address of the communication device, and the port number to connect to. An application uses the write command to instruct the TCPIP stack to send a message (i.e., data) to a communication device to which it is connected. In addition, an application uses the read command to ask the TCPIP stack to provide the message that was sent from the remote communication device to which it is



connected. A communication session typically exists of a connect, followed by a read and write on both sides.

FIG. 14 is a flowchart 600 further illustrating TCPIP acceleration in accordance with this alternative embodiment of the invention. As shown by blocks 601 and 602 when an application of the communication device makes a request to the communications stack to connect with the TCPIP server, that communication is intercepted by the acceleration application.

To find an agent, upon receiving that connect message from the communication device application, which includes the IP address of the TCPIP server and the port to connect to, the acceleration application in the client makes a request to the acceleration server to find out who the agent for the communication with the TCPIP server is. This step is performed in a similar manner to that described with regard to the main HTTP embodiment of the invention (block 604). As shown by block 606, the server then provides the client with a list of agents, for example, a primary agent and four others.

To establish a connection, as shown by block 608, the client issues a TCPIP connect with the primary agent or one of the other agents if the primary agent does not succeed, to create a connection with the agent. The client then sends to the agent the IP address of the TCPIP server and connection port that were provided by the communication device application (block 610). As shown by block 612, that agent in turn issues a TCPIP connect to the TCPIP server to the port it received from the client, to create a connection with the agent.

FIG. 15 is a flowchart 800 further illustrating TCPIP acceleration in accordance with this alternative embodiment of the invention, detailing the communication between the client and the TCPIP server (read and write commands) after the connect phase has completed successfully.

As shown by block 802, if the network application within the client wants to send a message to the TCPIP server, the network application within the client writes the message to the TCPIP stack in the operating system of the client. This WRITE command is received by the acceleration application of the client and handled in the manner described below. If the TCPIP server wants to send a message to the client, the TCPIP server writes the message to the TCPIP stack of TCPIP operating system, on the connection to the agent, since this agent is where the server received the original connection. This WRITE command is received by the acceleration application of the agent and handled in the manner described below.

When the acceleration application of the client receives a message from the network application of the client to be sent to the agent, or when the acceleration application of the agent receives a message from the connection to the TCPIP server that is to be sent to the client, the acceleration application proceeds to send the message to the communication device on the other side. For instance, if the client has intercepted the message from the communication application, the client sends the message to the agent, and if it is the agent that intercepted the message from the connection to the TCPIP server, such as the TCPIP server sending a message that is intended for the communication with client, the agent sends the message to the client in the following manner:

As shown by block 804, the acceleration application breaks up the content of the message to chunks and calculates the corresponding checksums, in the same manner as in the main embodiment described herein. The acceleration application then looks up each checksum in its cache database (block 806). As shown by block 808, the acceleration application checks if the checksum exists in the cache database. If it does, then, as shown by block 810, the acceleration

application prepares a list of peers that have already received the chunk of the checksum in the past (if any), and adds the communication device of the other side to the list of communication devices that have received this chunk (adds it to the peer list of the checksum in its database), to be provided to other communication devices requesting this information in the future. As shown by block 812, the list of peers is sent to the receiving communication device, which, as shown by block 814 retrieves the chunks from the peers in the list received, in the same manner as in the main embodiment.

If the checksum does not exist within the cache database of the sending communication device then, as shown by block 820, the acceleration application adds the checksum and chunk to its cache database, sends the chunk to the communication device on the other side, and adds the other communication device to the list of peers for that checksum in its database.

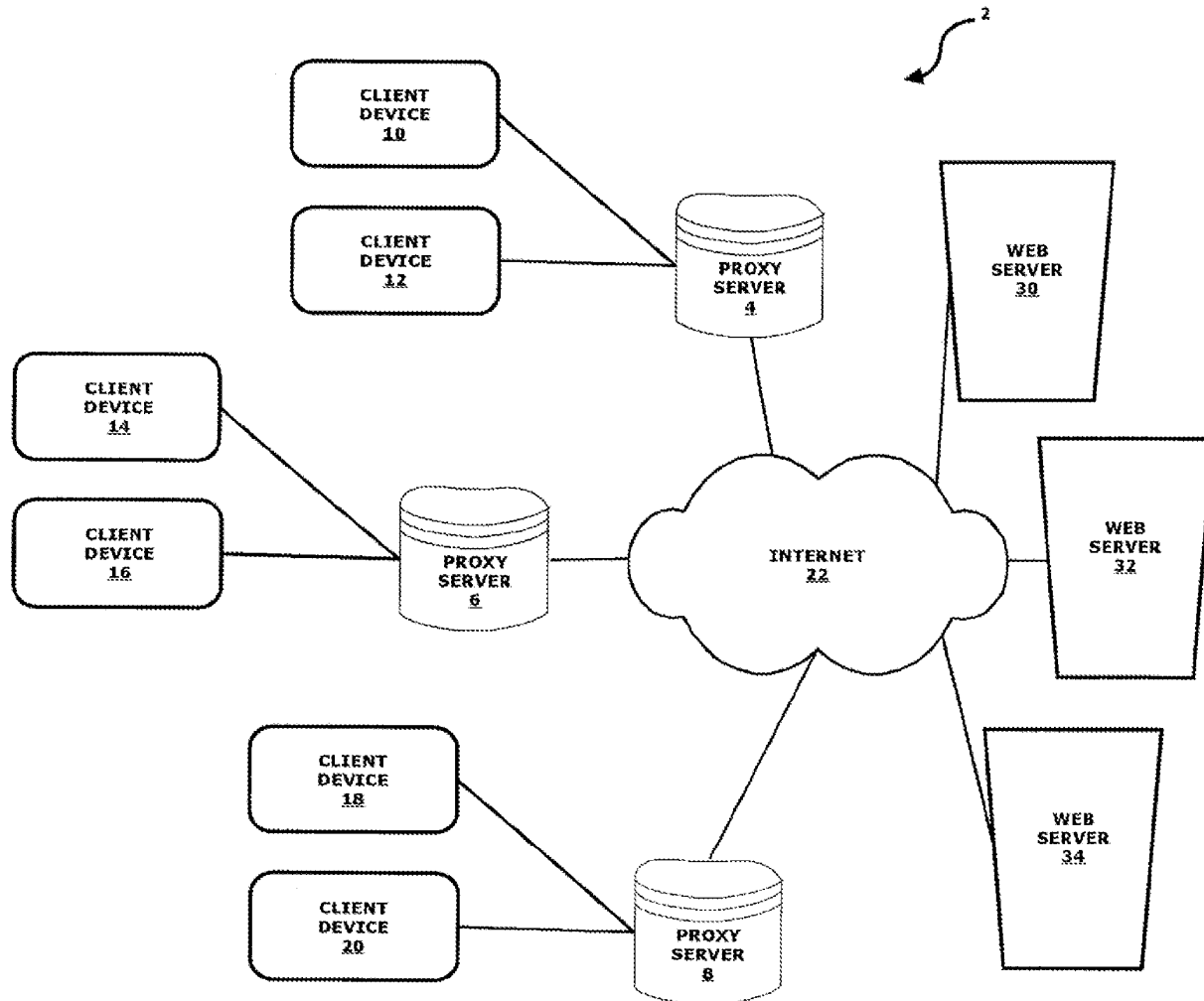
As shown by block 816, a determination is then made as to whether all chunks have been received. If all chunks have not been received, the process continues on again from block 806.

Once all data has been received, as shown by block 818, the acceleration application passes the data on to the requester. Specifically, in the client, the acceleration application passes on the complete data to the communication application, and in the agent, the acceleration application passes on the complete data to the requesting TCP/IP server.

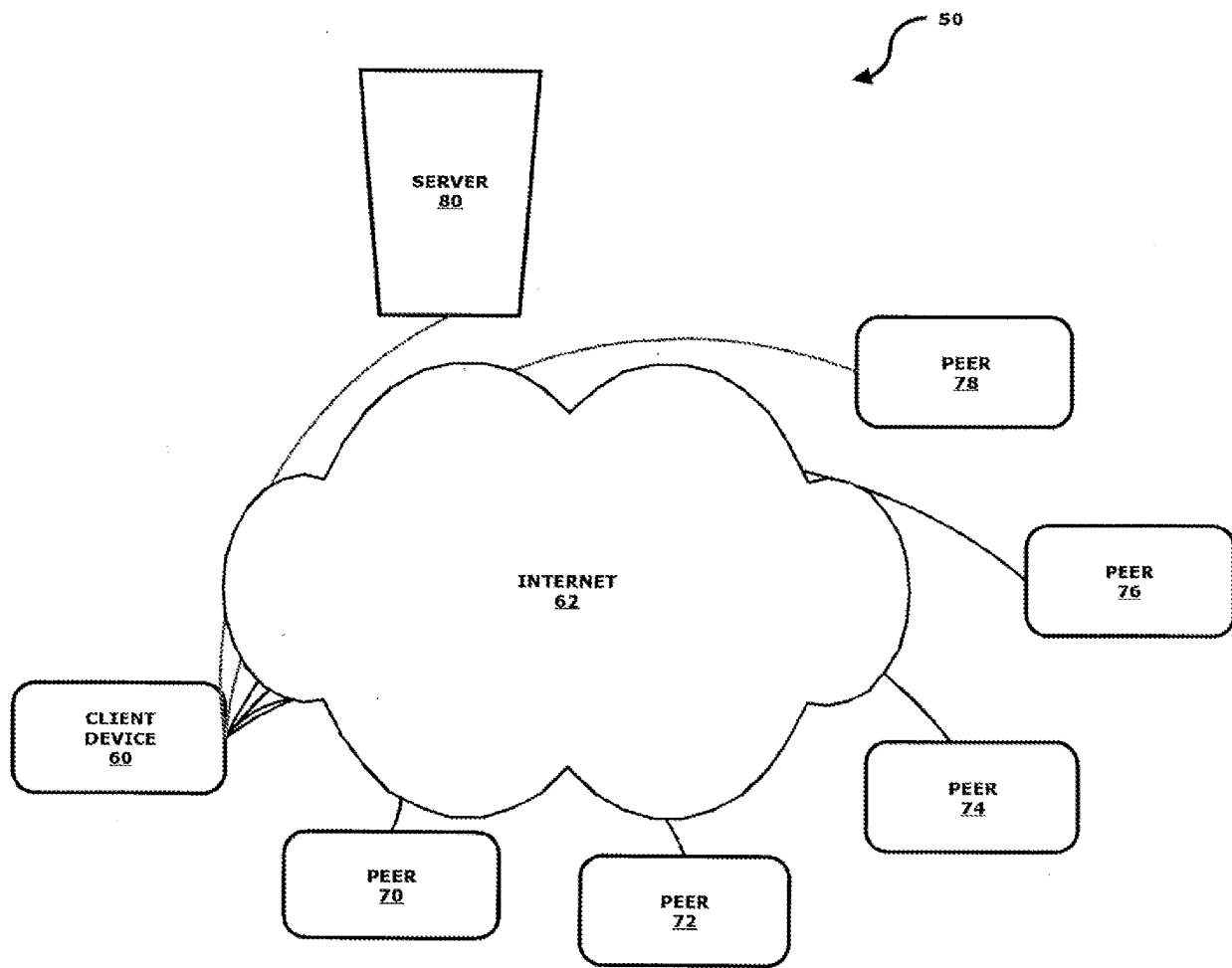
It should be emphasized that the above-described embodiments of the present invention are merely possible examples of implementations, merely set forth for a clear understanding of the principles of the invention. Many variations and modifications may be made to the above-described embodiments of the invention without departing substantially from the spirit and principles of the invention. All such modifications and variations are intended to be included

Attorney Docket No. 19459-6105P

herein within the scope of this disclosure and the present invention and protected by the following claims.



**FIG. 1**



**FIG. 2**

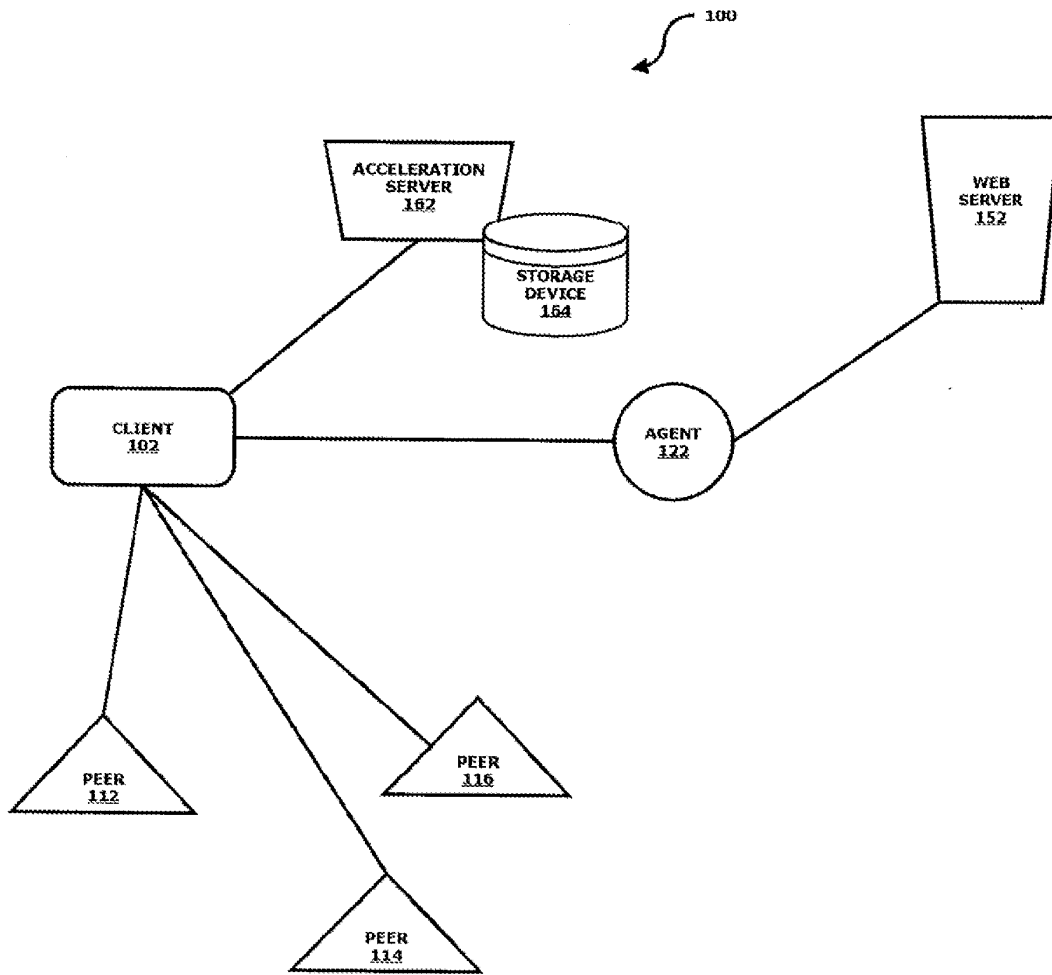
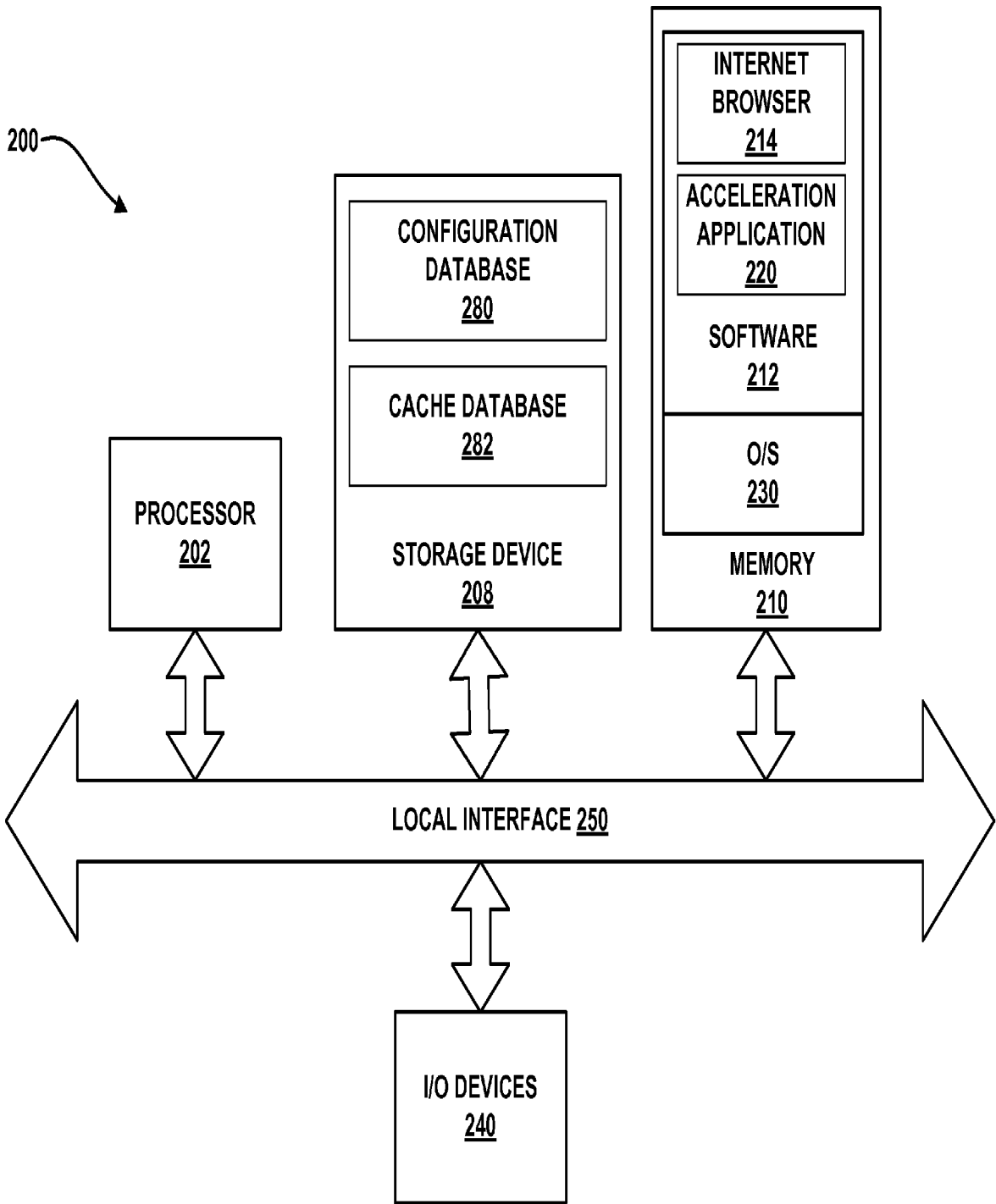
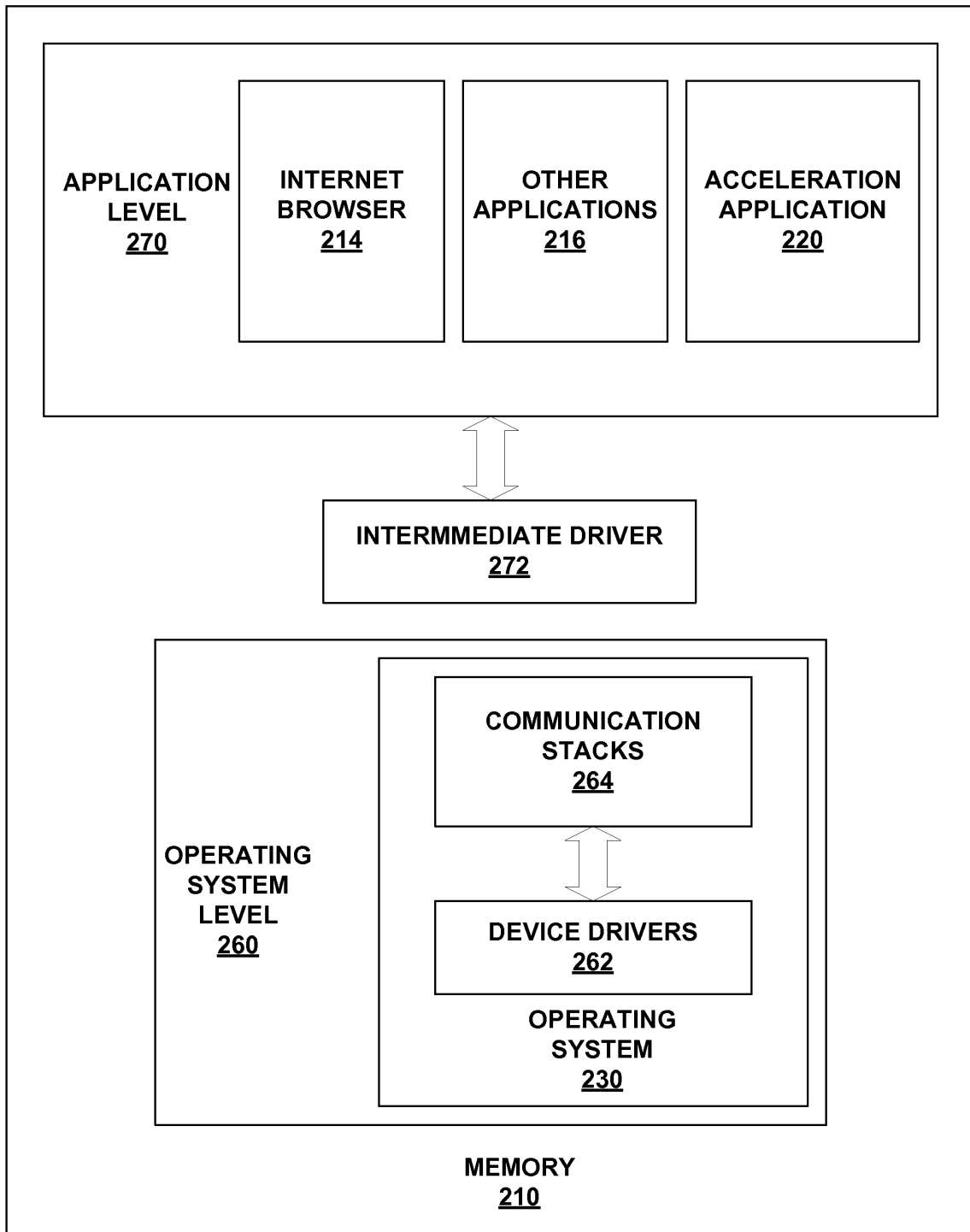


FIG. 3

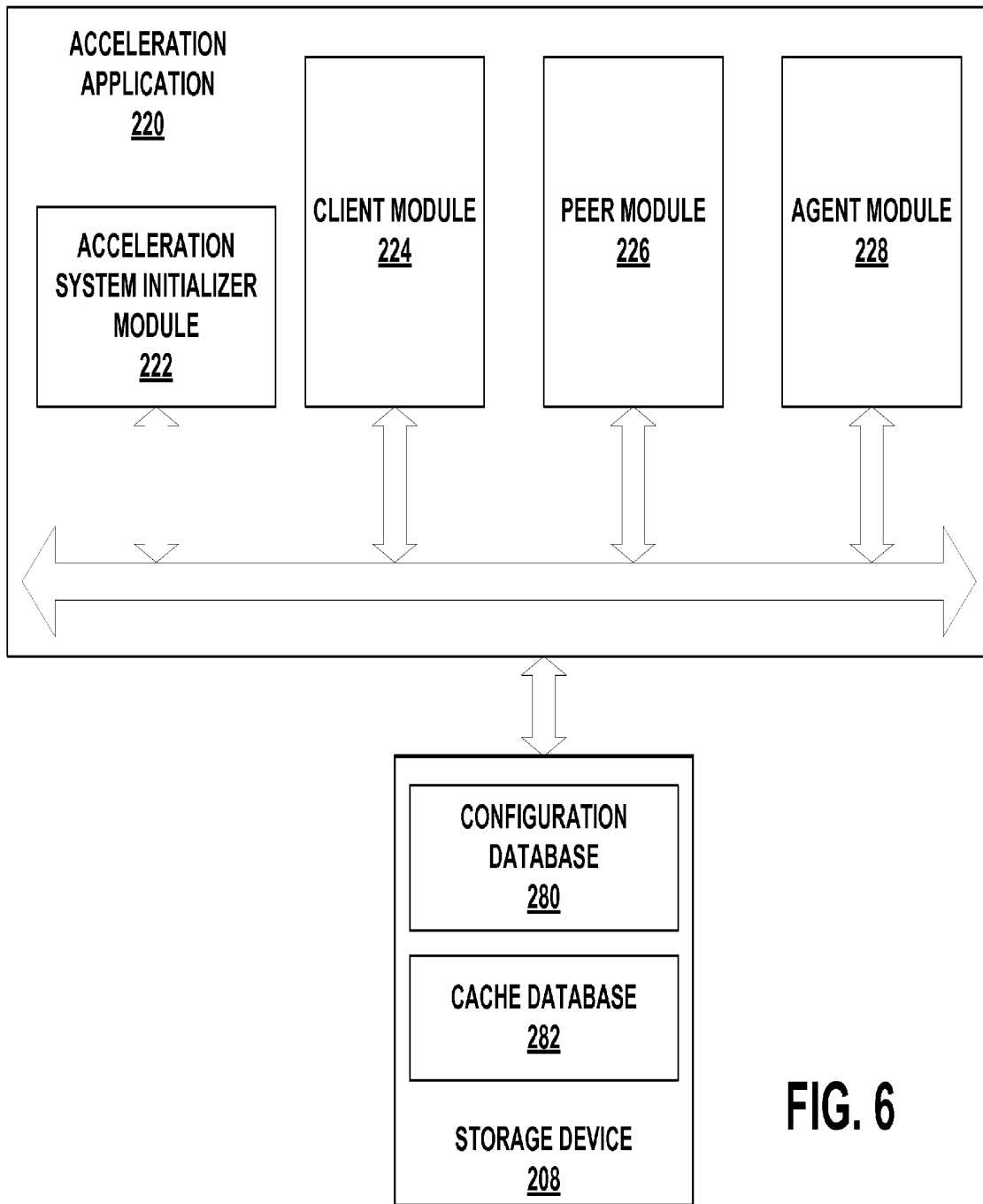


**FIG. 4**



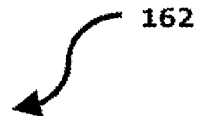


**FIG. 5**



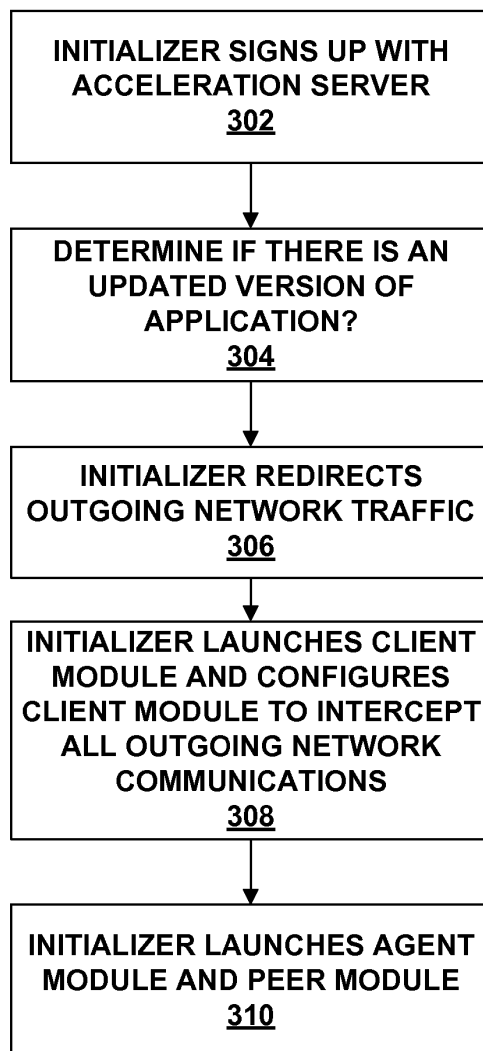
**FIG. 6**

ACCELERATION DATABASE 164			
166	AGENT IP A ONLINE/OFFLINE		
>>> INDEXED BY: AGENT IP ADDRESS			
CACHE DATABASE 282			
286	LIST OF URLS:		
288	URL 1		
	290 URL		
	292 URL HTTP HEADERS		
	294 LAST CHECKED ON SERVER		
	296 LAST CHANGED ON SERVER		
	298 LIST OF CHUNKS FOR THIS URL:		
	300 CHUNK 1		
		302 CHUNK CHECKSUM	
		304 CHUNK DATA	
		306 LIST OF PEERS:	
		308 PEER 1	
			310 PEER 1 IP ADDRESS
			312 PEER 2 CONNECTION STATUS

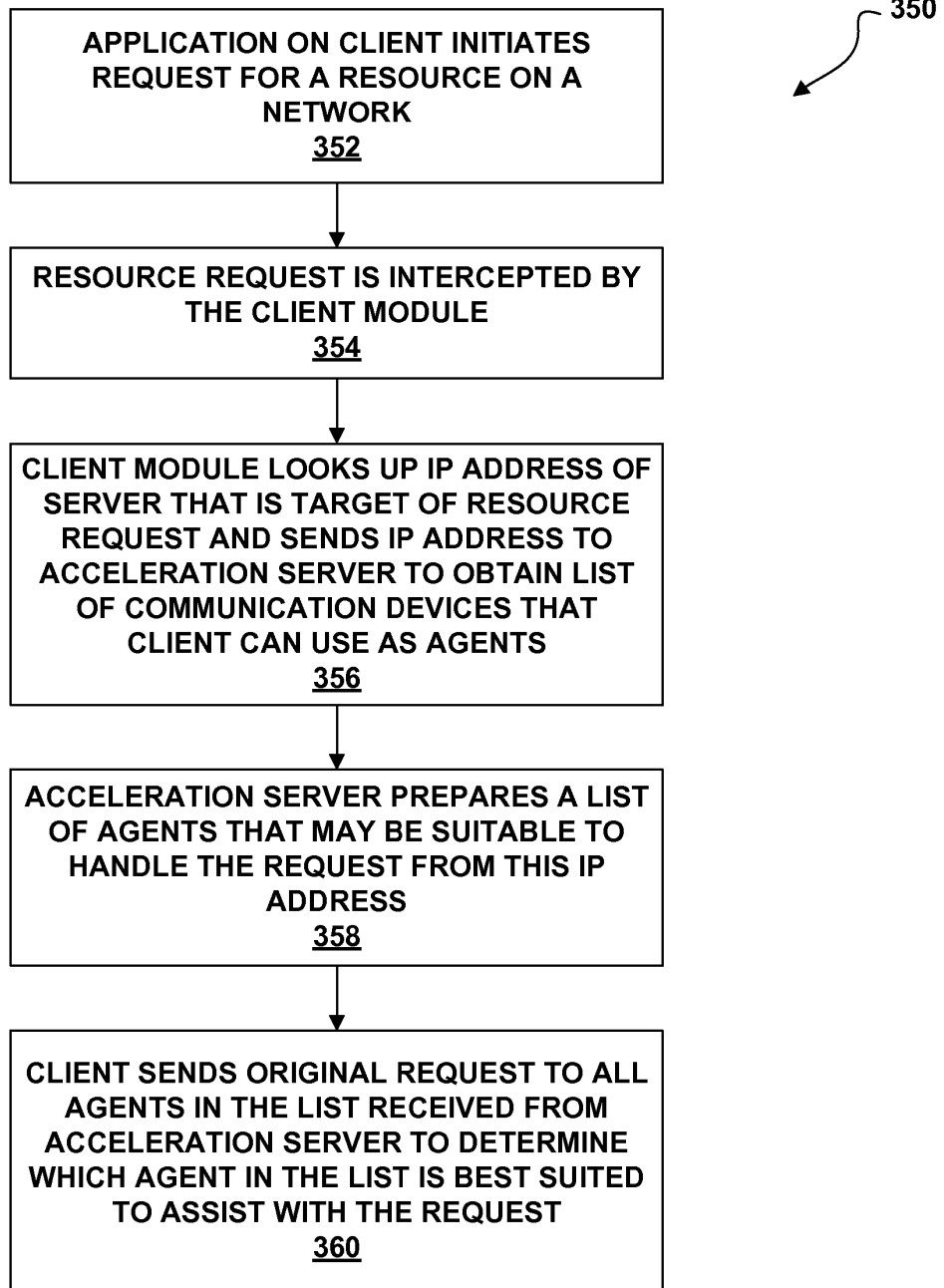


**FIG. 7**

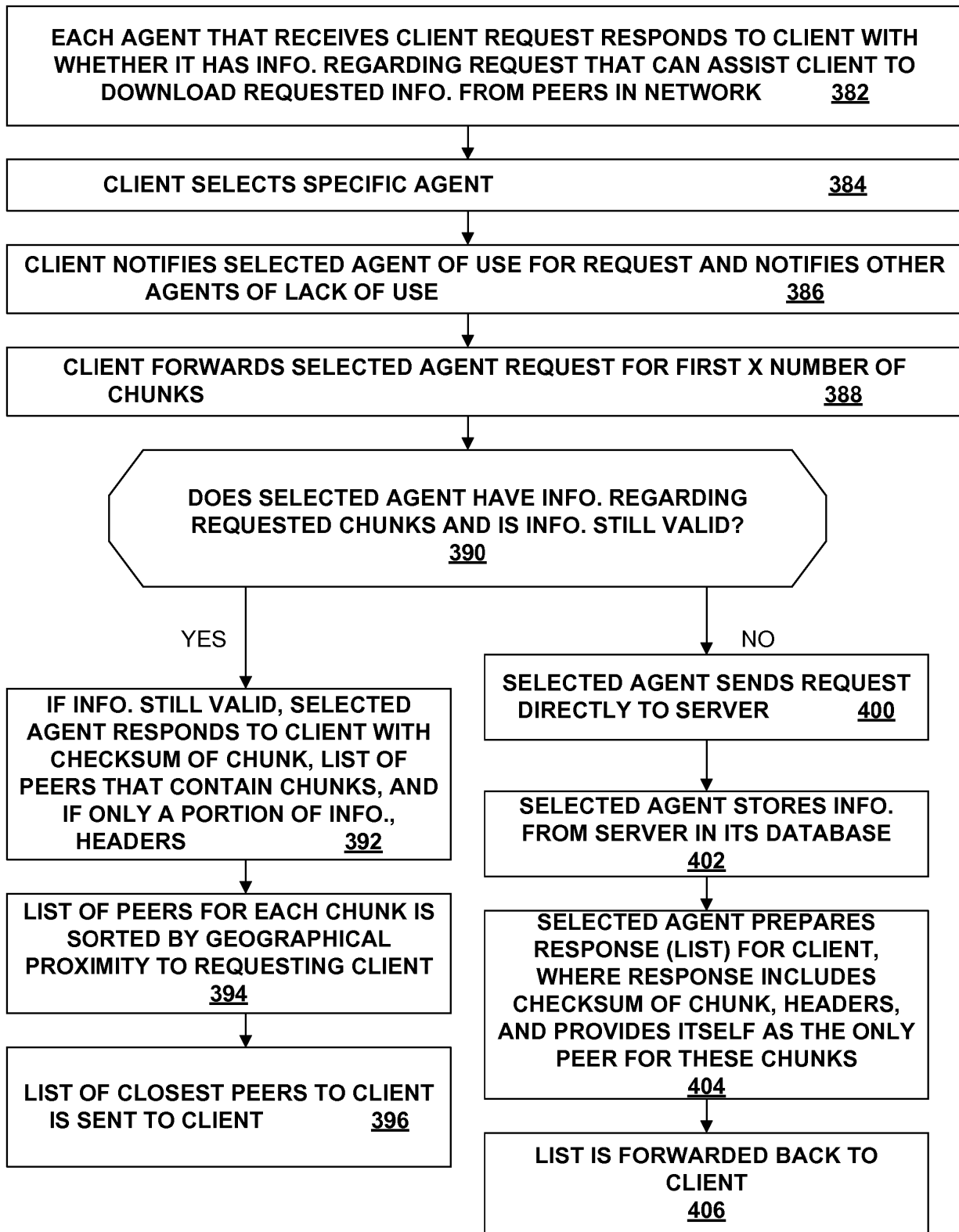
300



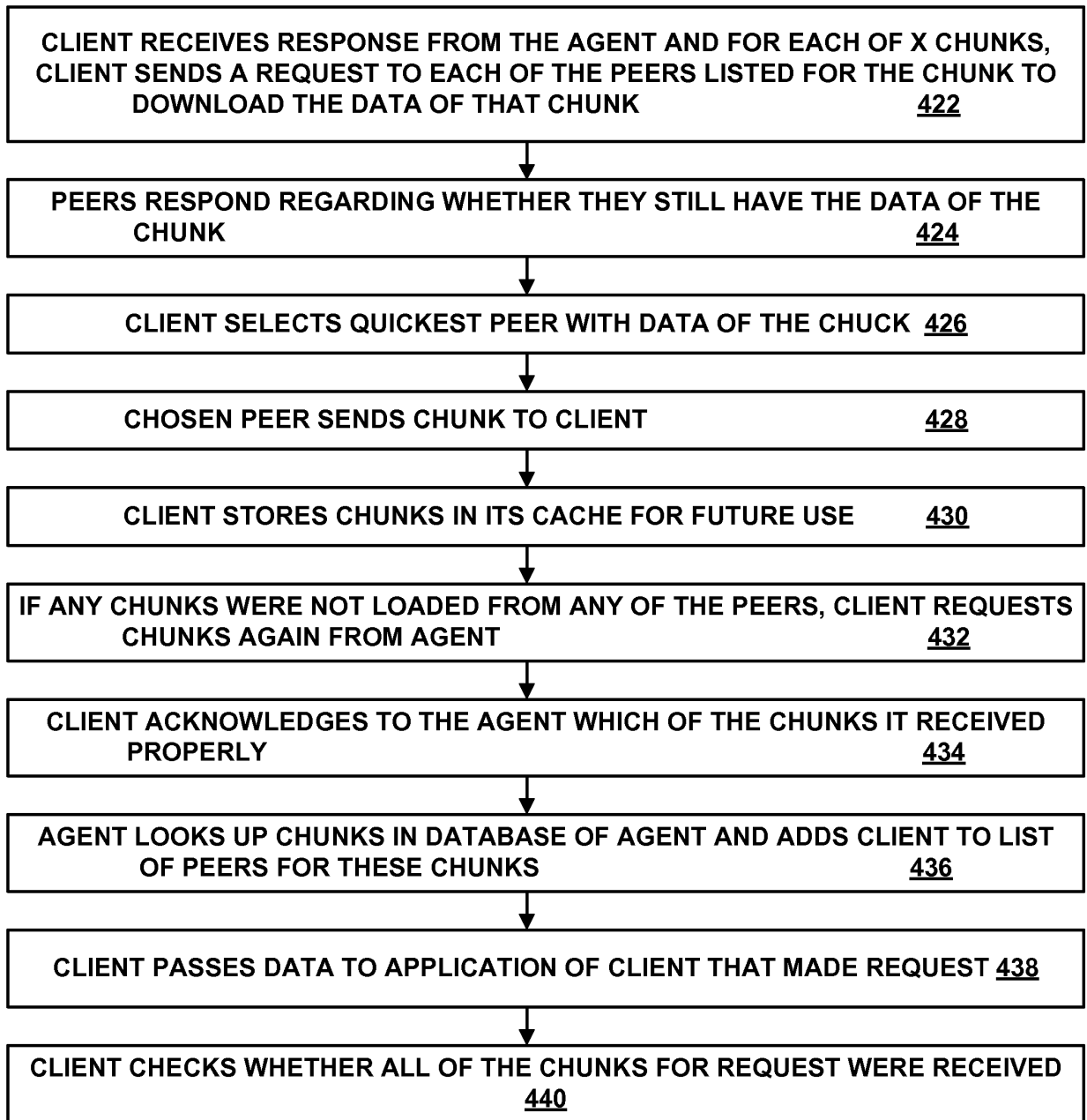
**FIG. 8**



**FIG. 9**

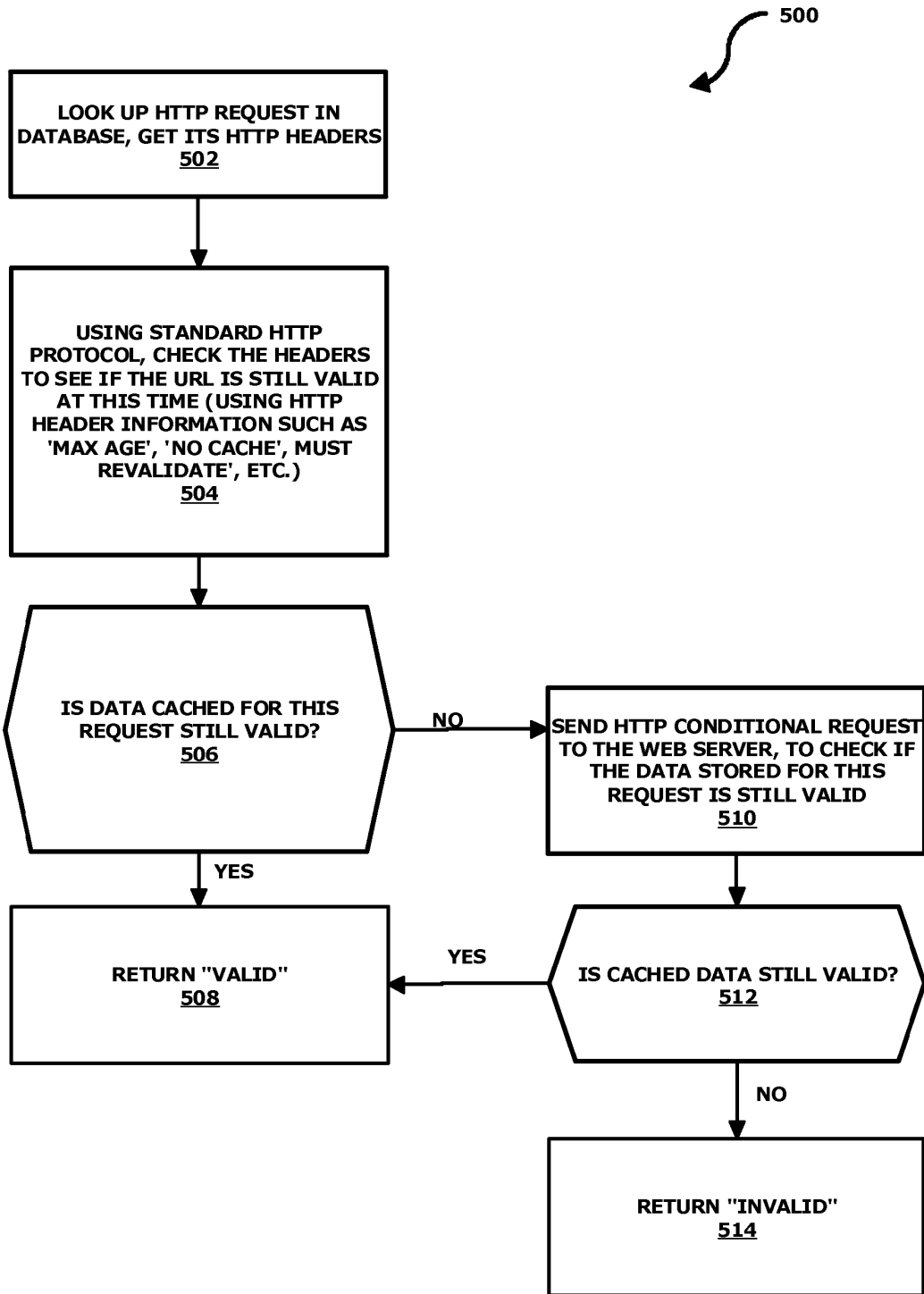


**FIG. 10**



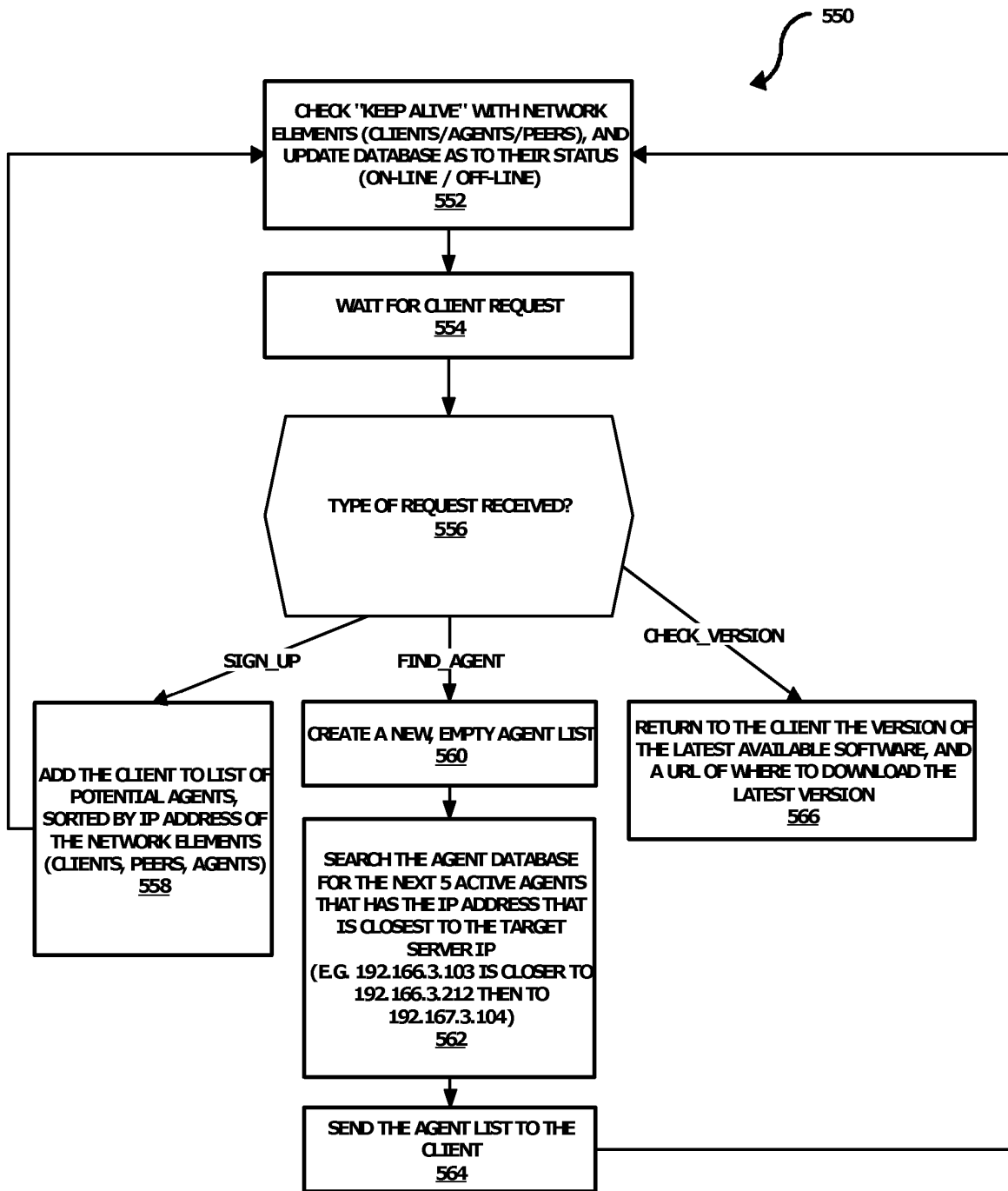
**FIG. 11**

420

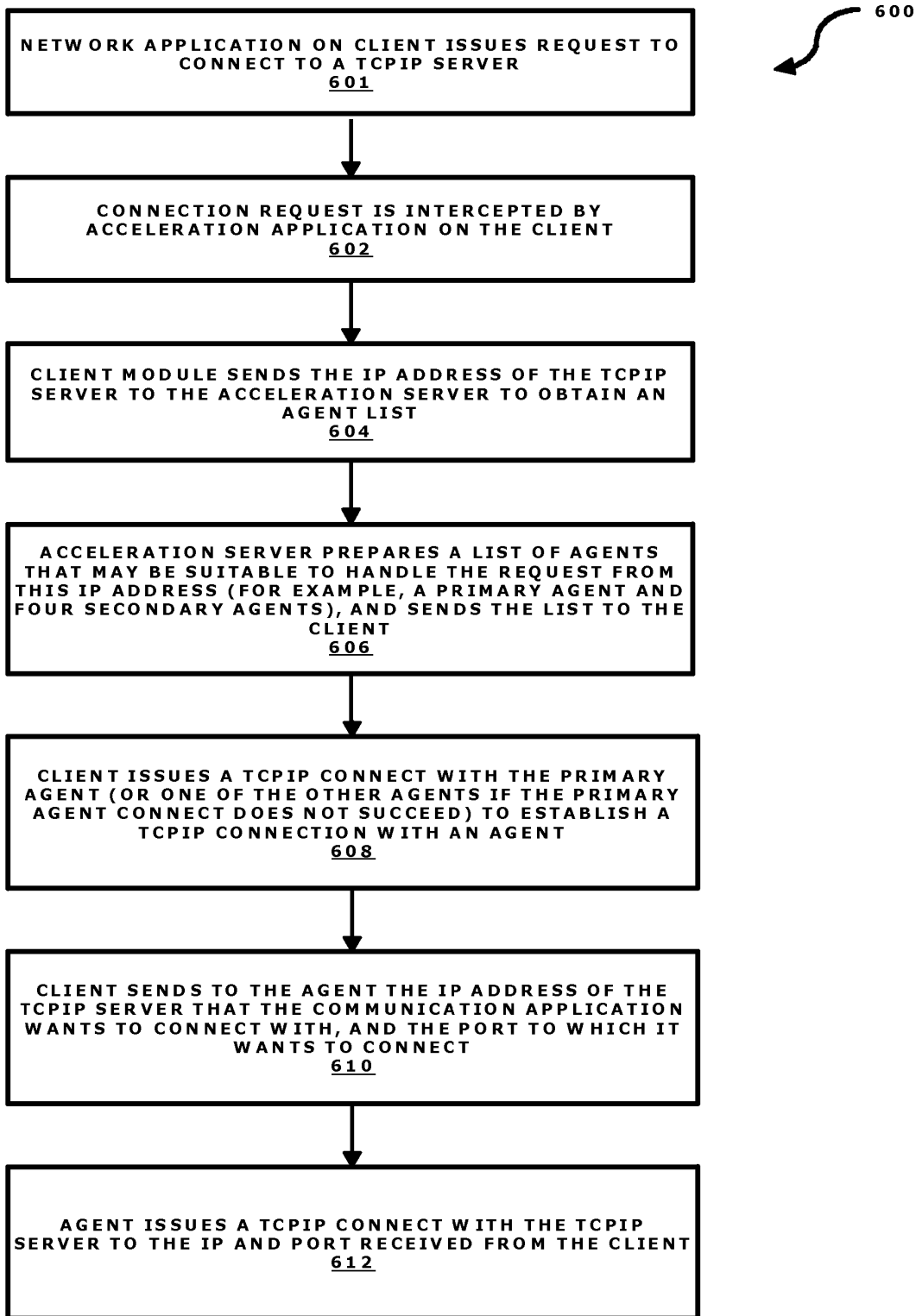


**FIG. 12**





**FIG 13**



**FIG. 14**

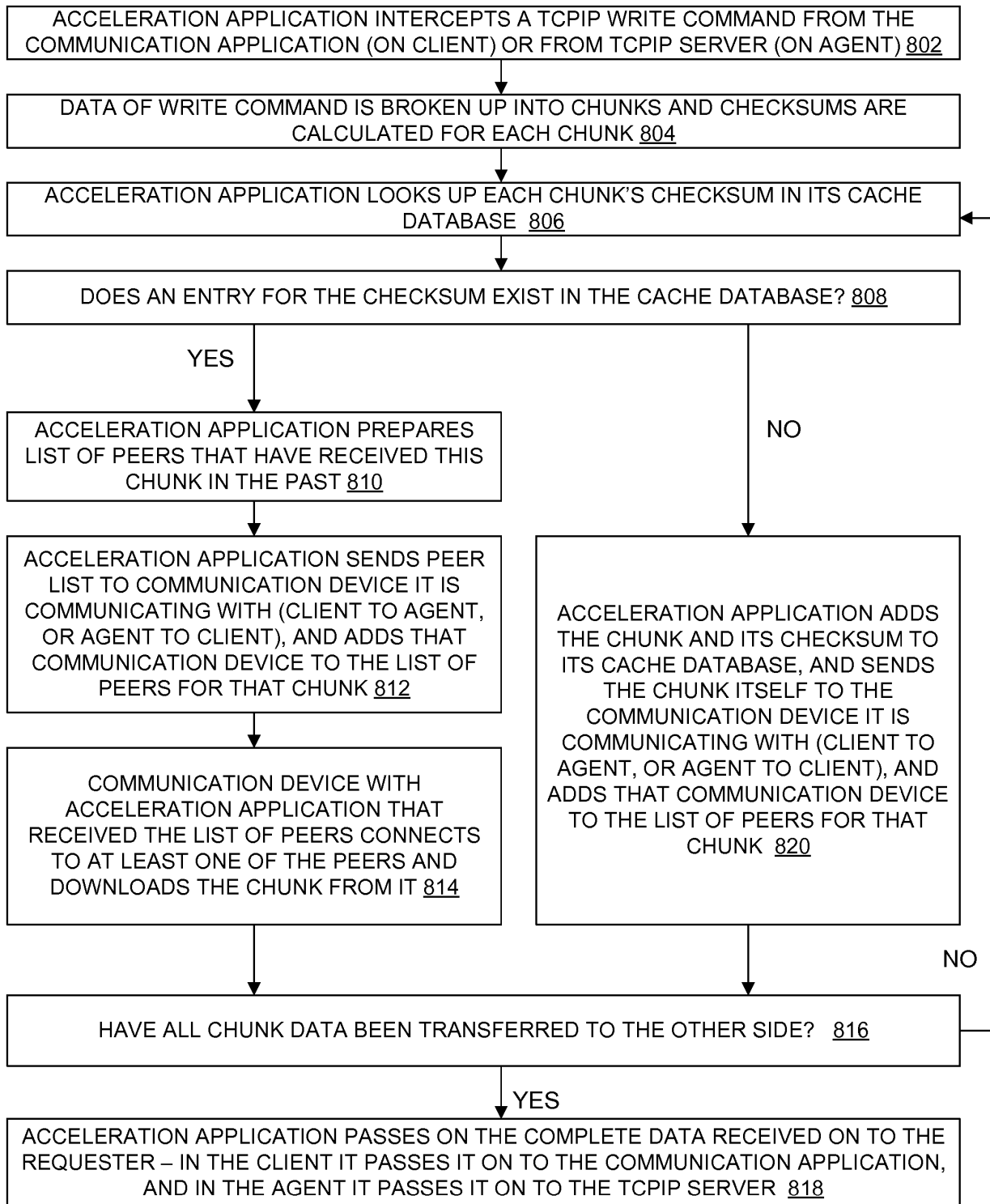


FIG. 15

800

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		
The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76. This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.			

### Secrecy Order 37 CFR 5.2:

Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

### Inventor Information:

Inventor	1				Remove
Legal Name					
Prefix	Given Name	Middle Name	Family Name	Suffix	
	Derry		Shribman		
Residence Information (Select One)    US Residency <input type="radio"/> Non US Residency    Active US Military Service					
City	Tel Aviv	Country of Residence <sup>i</sup>	L		
Mailing Address of Inventor:					
Address 1	9/6 Beylinson St.,				
Address 2					
City	Tel Aviv	State/Province			
Postal Code	6356709	Country <sup>i</sup>	IL		
Inventor	2				Remove
Legal Name					
Prefix	Given Name	Middle Name	Family Name	Suffix	
	Ofer		Vilenski		
Residence Information (Select One)    US Residency <input checked="" type="radio"/> Non US Residency    Active US Military Service					
City	Moshav Hadar Am	Country of Residence <sup>i</sup>	L		
Mailing Address of Inventor:					
Address 1	8 Hahollandim Street				
Address 2					
City	Moshav Hadar Am	State/Province			
Postal Code	42935	Country <sup>i</sup>	IL		
All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the Add button.					Add

### Correspondence Information:

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

Enter either Customer Number or complete the Correspondence Information section below.  
For further information see 37 CFR 1.33(a).

An Address is being provided for the correspondence information of this application.

Customer Number	131926		
Email Address		<input type="button" value="Add Email"/>	<input type="button" value="Remove Email"/>

### Application Information:

Title of the Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		
Attorney Docket Number	HOLA-005-US4	Small Entity Status Claimed	<input checked="" type="checkbox"/>
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	15	Suggested Figure for Publication (if any)	

### Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

### Publication Information:

Request Early Publication (Fee required at time of Request 37 CFR 1.219)

**Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

### Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32). Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer number will be used for the Representative Information during processing.

Please Select One:	<input checked="" type="radio"/> Customer Number	<input type="radio"/> US Patent Practitioner	<input type="radio"/> Limited Recognition (37 CFR 11.9)
Customer Number	131926		

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

### Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.

When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status	Pending					Remove
Application Number	Continuity Type		Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)		
	Continuation of		14/025109	2013-09-12		
Prior Application Status	Patented					Remove
Application Number	Continuity Type	Prior Application Number	Filing Date (YYYY-MM-DD)	Patent Number	Issue Date (YYYY-MM-DD)	
14/025109	Division of	12/836059	2010-07-14	8560604	2013-10-15	
Prior Application Status	Expired					Remove
Application Number	Continuity Type		Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)		
12/836059	Claims benefit of provisional		61/249624	2009-10-08		
Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the <b>Add</b> button.						Add

### Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)<sup>i</sup> the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Application Number	Country <sup>i</sup>	Filing Date (YYYY-MM-DD)	Access Code <sup>i</sup> (if applicable)	Remove
Additional Foreign Priority Data may be generated within this form by selecting the <b>Add</b> button.				Add

### Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

<p>This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.</p> <p><input type="checkbox"/> NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.</p>
--

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

## Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

**NOTE:** This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

### 1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

**A. Priority Document Exchange (PDX)** - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

**B. Search Results from U.S. Application to EPO** - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

### 2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

**NOTE:** Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.



Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

**Applicant Information:**

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.			
<b>Applicant</b>	1	<input type="button" value="Remove"/>	
If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.			
<input type="button" value="Clear"/>			
<input checked="" type="radio"/> Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor	
Person to whom the inventor is obligated to assign.		Person who shows sufficient proprietary interest	
If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:			
▼			
Name of the Deceased or Legally Incapacitated Inventor: <input type="text"/>			
If the Applicant is an Organization check here. <input checked="" type="checkbox"/>			
Organization Name	HOLA NEWCO LTD.		
<b>Mailing Address Information For Applicant:</b>			
Address 1	3 Hamahshev St.,		
Address 2			
City	Netanya	State/Province	
Country	IL	Postal Code	42507
Phone Number		Fax Number	
Email Address			
Additional Applicant Data may be generated within this form by selecting the Add button. <input type="button" value="Add"/>			

**Assignee Information including Non-Applicant Assignee Information:**

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

<b>Assignee</b>	1
-----------------	---

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

If the Assignee or Non-Applicant Assignee is an Organization check here.

Prefix	Given Name	Middle Name	Family Name	Suffix

**Mailing Address Information For Assignee including Non-Applicant Assignee:**

Address 1				
Address 2				
City		State/Province		
Country i		Postal Code		
Phone Number		Fax Number		
Email Address				

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button.

**Signature:**

**NOTE:** This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

<b>Signature</b>	/Yehuda Binder/		Date (YYYY-MM-DD)	2018-04-20
First Name	Yehuda	Last Name	BINDER	Registration Number
				73612

Additional Signature may be generated within this form by selecting the Add button.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>Application Data Sheet 37 CFR 1.76</b>		Attorney Docket Number	HOLA-005-US4
		Application Number	
Title of Invention	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION		

This collection of information is required by 37 CFR 1.76. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 23 minutes to complete, including gathering, preparing, and submitting the completed application data sheet form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

## Claims

1. A method for use with a first client device, for use with a first server that comprises a web server that is a Hypertext Transfer Protocol (HTTP) server that responds to HTTP requests, the first server stores a first content identified by a first content identifier, and for use with a second server, the method by the first client device comprising:

receiving, from the second server, the first content identifier;

sending, to the first server over the Internet, a Hypertext Transfer Protocol (HTTP) request that comprises the first content identifier;

receiving, the first content from the first server over the Internet in response to the sending of the first content identifier; and

sending, the first content by the first client device to the second server, in response to the receiving of the first content identifier.

2. The method according to claim 1, further comprising storing, by the first client device in response to the receiving from the first server, the first content, and wherein the sending, of the HTTP request is in response to the receiving of the first content identifier.

3. The method according to claim 2, further comprising:

receiving, from a second client device, the first content identifier; and

sending, the stored first content by the first client device to the second client device, in response to the

receiving of the first content identifier from the second client device.

4. The method according to claim 1, wherein the first or second server is a Transmission Control Protocol / Internet Protocol (TCP/IP) server that communicates over the Internet based on, or according to, using TCP/IP protocol or connection, and wherein the first client device is a Transmission Control Protocol / Internet Protocol (TCP/IP) client that communicates respectively with the first or second server over the Internet based on, or according to, TCP/IP protocol or connection.

5. The method according to claim 1, wherein the first client device communicates over the Internet based on, or according to, one out of UDP, DNS, TCP, FTP, POP#, SMTP, or SQL standards.

6. The method according to claim 1, wherein the first content comprises web-page, audio, or video content, wherein the first content identifier comprises a Uniform Resource Locator (URL), and wherein the method further comprising executing, by the first client device, a web browser application or an email application.

7. The method according to claim 1, for use with a third server that comprises a web server that is Hypertext Transfer Protocol (HTTP) server, the third server responds to HTTP requests and stores a second content identified by a second content identifier, the method by the first client device further comprising:

identifying, an HTTP request for the second content;

sending, to the second server over the Internet in response to the identifying, the second content identifier and a criterion; and

receiving, over the Internet in response to the sending, from a second client device selected from a plurality of client devices according to the criterion, a part of, or a whole of, the second content.

8. The method according to claim 7, further comprising executing, by the first client device, a web browser application or an email application, and wherein the identifying comprises intercepting, by a driver in the first client device, the request for the second content respectively from the web browser application or the email application.

9. The method according to claim 7, wherein the criterion is stored in the first client device, and the method further comprising selecting, by the first client device, the second client device from the plurality of client devices, according to the stored criterion.

10. The method according to claim 9, wherein the criterion is based on, or comprises, the geographical location of the plurality of client devices, or a response time when communicating with the first client device.

11. The method according to claim 9, wherein the second client device is the quickest to respond to queries from the first client device.

12. The method according to claim 9, further comprising sending, by the first client device, a notification message to a device from the plurality of client devices that was not selected as part of the selecting.

13. The method according to claim 1, further comprising periodically communicating between the second server and the first client device.

14. The method according to claim 13, wherein the periodically communicating comprises exchanging 'keep alive' messages.

15. The method according to claim 1, further comprising establishing, by the first client device, a Transmission Control Protocol (TCP) connection with the second server using TCP/IP protocol.

16. The method according to claim 1, wherein the first client device is identified by a Media Access Control (MAC) address or a hostname, and wherein the method further comprising sending, by the first client device, during, as part of, or in response to, a start-up of the first client device, a first message to the second server, and wherein the first messages comprises the first IP address, the MAC address, or the hostname.

17. The method according to claim 16, for use with a first application stored in the first client device and associated with a first version number, wherein the first message comprises the first version number.

18. The method according to claim 17, for use with a second application that is a version of the first application, is stored in the second server, and is associated with a second version number, wherein the method further comprising receiving, by the first client device from the second server, in response to the first message, a second message that comprises the second version number.

19. The method according to claim 18, wherein the method further comprising downloading over the Internet, by the first



client device from the second server, in response to the first message, the second application from the second server, and installing the second application in the first client device as a replacement for the first application.

20. The method according to claim 1, wherein the first or second server is a Transmission Control Protocol / Internet Protocol (TCP/IP) server, wherein the first client device communicates over the Internet with the first or second server based on, or according to, using TCP/IP protocol or connection.

21. The method according to claim 1, further comprising determining, by the first client device, that the received first content, is valid.

22. The method according to claim 22, wherein the determining is based on the received HTTP header according to, or based on, IETF RFC 2616.

23. The method according to claim 22, further comprising:  
    sending, a message over the Internet in response to the determining that the received first content, is not valid; and  
    receiving, over the Internet in response to the sending of the message, from the second server or from a second client device selected from a plurality of client devices, the first content.

24. The method according to claim 1, further comprising storing, operating, or using, a client operating system.

25. The method according to claim 1, wherein the steps are sequentially executed.

26. The method according to claim 1, for use with a software application that includes computer instructions that, when

executed by a computer processor, cause the processor to perform the sending of the Hypertext Transfer Protocol (HTTP) request, the receiving and storing of the first content, the receiving of the first content identifier, and the sending of the part of, or the whole of, the stored first content, the method is further preceded by:

    downloading, by the first client device from the Internet, the software application; and

    installing, by the first client device, the downloaded software application.

27. The method according to claim 26, wherein the software application is downloaded from the second server.

28. A non-transitory computer readable medium containing computer instructions that, when executed by a computer processor, cause the processor to perform the method according to claim 1.

29. A client device comprising a non-transitory computer readable medium containing computer instructions that, when executed by a computer processor, cause the processor to perform the method according to claim 1.

## **Abstract**

A system designed for increasing network communication speed for users, while lowering network congestion for content owners and ISPs. The system employs network elements including an acceleration server, clients, agents, and peers, where communication requests generated by applications are intercepted by the client on the same machine. The IP address of the server in the communication request is transmitted to the acceleration server, which provides a list of agents to use for this IP address. The communication request is sent to the agents. One or more of the agents respond with a list of peers that have previously seen some or all of the content which is the response to this request (after checking whether this data is still valid). The client then downloads the data from these peers in parts and in parallel, thereby speeding up the Web transfer, releasing congestion from the Web by fetching the information from multiple sources, and relieving traffic from Web servers by offloading the data transfers from them to nearby peers.

---

**MAY PATENTS LTD.**  
Yehuda BINDER, U.S. Patent Agent  
*B.Sc.E.E.; M.Sc.E.E.; M.B.A*

---

April 20, 2018

U.S. Patent and Trademark Office (USPTO)  
Customer Service Window  
Mail Stop Patent Application  
401 Dulany Street  
Alexandria, VA 22314

Re: **New Utility Patent Application in U.S.**  
Applicant(s): **HOLA NEWCO LTD.**  
**Title: SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA  
COMMUNICATION**  
Attorney Docket No.: HOLA-005-US4

Sir/Madam,

Attached herewith is the above-identified application for Letters Patent including:

1. A request for prioritized examination under 37 CFR 1.102(e) (PTO/SB/424);
2. Applicant asserts small entity status. See 37 CFR 1.27.
3. Application Data Sheet (PTO/AIA/14);
4. Specification (33 pages), Claims 1-29 (6 pages) and abstract (1 page).
5. Fifteen (15) sheets of Drawings (Figures 1-15).
6. Declaration  
[ X ] Newly executed [ ] Copy from prior application no. \_\_\_\_\_
7. Assignment submitted through EPAS  
[ X ] Newly executed [ ] Copy from prior application no. \_\_\_\_\_
8. Power of Attorney
9. Information Disclosure Statement (PTO/SB/08).

Certain documents were previously filed or cited to the USPTO in the prior application 14/025,109, which is relied upon under 35 U.S.C. § 120.

Applicant(s) identify these documents by attaching an Information Disclosure Statement listing these documents and request that they be considered and made of record in accordance with 37 CFR § 1.98(d). Per Section 1.98(d), copies of these documents need not be filed in the application.

*e-mail: [yehuda@maypatents.com](mailto:yehuda@maypatents.com); Mobile: +972-54-4444577*

.....  
**MAY PATENTS LTD.**  
 Yehuda BINDER, U.S. Patent Agent  
 B.Sc.E.E.; M.Sc.E.E; M.B.A  
 .....

10. Electronic Payment in the amount of US\$ 3,435 is being made by deposit account no. 506726 to cover filing fee calculated as follows:

Application as Filed				Fee (US\$)
Basic Filing Fee				75.00
Search Fee				330.00
Examination Fee				380.00
Publication Fee				130.00
Prioritized Examination Processing Fee				70.00
Prioritized Examination Fee				2,000.00
<b>Total Sheets</b>	<b>Extra Sheets</b>		<b>Rate</b>	
55/100	--		200.00	--
<b>Claims</b>	<b># Filed</b>	<b># Extra</b>	<b>Rate</b>	
Total Claims	1-29	9	50.00	450.00
Independent Claims	3	--	230.00	--
Multiple Dependent Claim			410.00	--
<b>Total Filing Fee</b>				<b>3,435.00</b>

1. As in the prior application 14/025,109, please associate the above referenced application with **Customer No. 131926**.
2. The Correspondence Address associated with **Customer No. 131926**.

Submitted by,  
May Patents Ltd.

By: /Yehuda Binder/  
Yehuda Binder  
Registration No. 73,612

*e-mail: [yehuda@maypatents.com](mailto:yehuda@maypatents.com); Mobile: +972-54-4444577*

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		
	Filing Date		
	First Named Inventor	Derry Shribman	
	Art Unit		
	Examiner Name		
	Attorney Docket Number	HOLA-005-US4	

U.S.PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
	1	8479251	B2	2013-07-02	Feinleib et al	
	2	8499059	B2	2013-07-30	Stoyanov	
	3	7970835	B2	2011-28-01	Xerox Corporation	
	4	8832179	B2	2014-09-09	Owen , et al.	
	5	6173330	B1	2001-09-01	Guo , et al.	
	6	8769035	B2	2014-01-07	Resch , et al.	
	7	8171101	B2	2012-05-01	Gladwin , et al.	
	8	7558942	B2	2009-07-07	Chen , et al.	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

9	4937781	A	1990-06-26	Lee , et al.
10	7970835	B2	2011-06-28	Robert St. Jacques

If you wish to add additional U.S. Patent citation information please click the Add button.

**U.S.PATENT APPLICATION PUBLICATIONS**

Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20150067819	A1	2015-03-05	Hola Networks Ltd.	
	2	20120254456	A1	2012-10-04	Misharam Zubair et al.	
	3	20080222291	A1	2008-09-11	Weller et al.	
	4	20100235438	A1	2010-09-16	Narayanan et al.	
	5	20120124239	A1	2012-05-17	Shribman et al.	
	6	20130166768	A1	2013-06-27	Thomson Licensing	
	7	20020065930	A1	2002-30-05	Rhodes, David L.	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

8	20030204602	A1	2003-10-30	Hudson Michael D.
9	20120099566	A1	2012-04-26	Laine; Tuomas ; et al.
10	20130201316	A1	2013-08-08	BINDER; Yehuda ; et al.
11	20080125123	A1	2008-05-29	Dorenbosch; Jheroen P. ; et al.
12	20140301334	A1	2014-10-09	Labranche; Miguel ; et al.
13	20070239655	A1	2007-10-11	Agetsuma; Masakuni ; et al.
14	20070226810	A1	2007-09-27	Hotti; Timo
15	20100094970	A1	2010-04-15	Zuckerman; Gal ; et al.
16	20020120874	A1	2002-29-08	Shu, Li ; et al.
17	20100115063	A1	2010-06-05	GLADWIN; S. CHRISTOPHER ; et al.
18	20100154044	A1	2010-17-06	Manku; Tajinder



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

19	20100293555	A1	2010-15-11	VEPSALAINEN; Ari M.
20	20130272519	A1	2013-17-10	Huang; Lawrence P.
21	20030115364	A1	2003-06-19	Shu Li et al.
22	20090217122	A1	2009-27-08	Yokokawa; Takashi ; et al.
23	20010033583	A1	2001-25-10	Rabenko, Theodore F. ; et al.
24	20080109446	A1	2008-05-08	Wang Matrix XIN
25	20020133621	A1	2002-09-19	Talmon Marco et al
26	20040107242	A1	2004-06-03	John Vert et al
27	20070073878	A1	2007-03-29	Alfredo C. Issa
28	20090319502	A1	2009-12-24	Olivier Chalouhi et al
29	20060212584	A1	2006-09-21	Mingjian Yu et al

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

If you wish to add additional U.S. Published Application citation information please click the Add button.

**FOREIGN PATENT DOCUMENTS**

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2</sup> i	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
	1	2015034752	WO	A1	2015-03-12	Akamai Technologies INC		
	2	2000/018078	WO	A1	2000-03-30	Sopuch David. J		
	3	0948176	EP	A2	1999-10-06	Siemens Inf &Comm Networks		
	4	2597869	EP	A1	2015-05-29	Sharp Kabushiki Kaisha Osaka-shi		
	5	2010090562	WO	A1	2010-08-12	Telefonaktiebolaget L M Ericsson		
	6	2007280388	JP		2007-25-10	Xerox Corporation		
	7	1020090097034	KR		2009-15-09	KT Corporation		
	8	2343536	RU	C2	2009-10-01	Microsoft Corporation		
	9	101075242	CN	A	2007-11-21	TENGXUN SCIENCE & TECHNOLOGY		

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		
	Filing Date		
	First Named Inventor	Derry Shribman	
	Art Unit		
	Examiner Name		
	Attorney Docket Number	HOLA-005-US4	

	10	101179389	CN	A	2008-05-14	Wang Matrix XIN		
--	----	-----------	----	---	------------	-----------------	--	--

If you wish to add additional Foreign Patent Document citation information please click the Add button

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
	1	R. Fielding et al, RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1, June 1999, retrieved from the Internet <a href="http://rfc-editor.org">http://rfc-editor.org</a> [retrieved Apr. 15, 2002] (114 pages)	
	2	"On the Leakage of Personally Identifiable Information via Online Social Networks"-Wills et al, AT&T, Apr. 2009 <a href="http://www2.research.att.com/~bala/papers/wosn09.pdf">http://www2.research.att.com/~bala/papers/wosn09.pdf</a> .	
	3	Notice of Preliminary Rejection in KR Application No. 10-2012-7011711 dated July 15, 2016	
	4	KEI SUZUKI, a study on Cooperative Peer Selection Method in P2P Video Delivery, Vol. 109, No. 37, IEICE Technical Report, The Institute of Electronics, Information and Communication Engineers, May 14, 2009	

If you wish to add additional non-patent literature document citation information please click the Add button

**EXAMINER SIGNATURE**

Examiner Signature	Date Considered
--------------------	-----------------

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		
	Filing Date		
	First Named Inventor	Derry Shribman	
	Art Unit		
	Examiner Name		
	Attorney Docket Number	HOLA-005-US4	

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2018-04-19
Name/Print	Yehuda BINDER	Registration Number	73612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	
	Filing Date	
	First Named Inventor	Derry Shribman
	Art Unit	
	Examiner Name	
	Attorney Docket Number	HOLA-005-US4

U.S.PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
	1	3922494	A	1975-11-25	Cooper , et al.	
	2	5758195	A	1998-05-26	Balmer; Keith	
	3	6061278	A	2000-05-09	Kato , et al.	
	4	6466470	B1	2002-10-15	Houn Chang	
	5	7865585		2011-01-04	Allen Samuels, et al.	
	6	7120666		2006-10-10	Steven McCanne, et al.	
	7	7203741		2007-04-10	Talmon Marco, et al.	
If you wish to add additional U.S. Patent citation information please click the Add button.						Add
<b>U.S.PATENT APPLICATION PUBLICATIONS</b>						Remove

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
	1	20030009518	A1	2003-01-09	Harrow, Ivan P. ; et al.	
	2	20030074403	A1	2003-04-17	Harrow, Ivan P. ; et al.	
	3	20140082260	A1	2014-03-20	OH; HakJune ; et al.	
	4	20110314347	A1	2011-12-22	NAKANO; Rikizo ; et al.	
	5	20100329270	A1	2010-12-30	Asati; Rajiv ; et al.	
	6	20100085977	A1	2010-04-08	Khalid; Mohamed ; et al.	
	7	20100066808	A1	2010-03-18	Tucker; Curtis E. ; et al.	
	8	20090279559	A1	2009-11-12	Wong; Yuen Fai ; et al.	
	9	20080025506	A1	2008-01-31	Muraoka; Jochiku	
	10	20040264506	A1	2004-12-30	Furukawa, Rei	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		
Filing Date		
First Named Inventor	Derry Shribman	
Art Unit		
Examiner Name		
Attorney Docket Number	HOLA-005-US4	

11	20020123895	A1	2002-09-05	Sergey Potekhin
12	20150033001	A1	2015-01-29	Ivanov; Vladimir
13	20150358648	A1	2015-12-10	Limberg; Allen LeRoy
14	20160021430	A1	2016-01-21	LaBosco; Mark ; et al.
15	20110087733	A1	2011-04-14	Derry Shribman; et al.
16	20030174648	A1	2003-09-18	Mea Wang; et al.
17	20080008089	A1	2008-01-10	Claudson F. Bornstein; et al.
18	20040088646	A1	2004-05-06	William J. Yeager; et al.
19	20030009583	A1	2003-01-09	Chung Chan; et al.
20	20080235391	A1	2008-09-25	Christopher Painter; et al.
21	20070156855	A1	2007-07-05	Moses Johnson



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	
Filing Date	
First Named Inventor	Derry Shribman
Art Unit	
Examiner Name	
Attorney Docket Number	HOLA-005-US4

22	20020007413	A1	2002-01-17	JJ Garcia-Luna-Aceves, et al.
23	20030210694	A1	2003-11-13	Suresh Jayaraman, et al.
24	20030200307	A1	2003-10-23	Jyoti Raju, et al.

If you wish to add additional U.S. Published Application citation information please click the Add button.

**FOREIGN PATENT DOCUMENTS**

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2</sup>	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button.

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
	1	International Search Report issued in PCT Application No. PCT/US2010/051881 dated 09 December 2010	
	2	Supplementary European Search Report issued in EP Application No. 10822724 dated 24 April 2013	

If you wish to add additional non-patent literature document citation information please click the Add button.

**EXAMINER SIGNATURE**

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	
Filing Date	
First Named Inventor	Derry Shribman
Art Unit	
Examiner Name	
Attorney Docket Number	HOLA-005-US4

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	
Filing Date	
First Named Inventor	Derry Shribman
Art Unit	
Examiner Name	
Attorney Docket Number	HOLA-005-US4

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2018-04-19
Name/Print	Yehuda BINDER	Registration Number	73612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.



**SUPPLEMENTARY  
EUROPEAN SEARCH REPORT**

Application Number  
EP 10 82 2724

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	US 2008/109446 A1 (WANG MATRIX XIN [CN]) 8 May 2008 (2008-05-08) * paragraphs [0010] - [0011] * * paragraphs [0024] - [0035] * * figures 3-5 * -----	1-14	INV. G01R31/08 G06F11/00 G08C15/00 H04L29/08 H04L12/24
X	US 2007/156855 A1 (JOHNSON MOSES [US]) 5 July 2007 (2007-07-05) * paragraphs [0013] - [0016] * * paragraphs [0022] - [0036] * * figure 1 * -----	1-14	
X	US 2008/235391 A1 (PAINTER CHRISTOPHER [US] ET AL) 25 September 2008 (2008-09-25) * paragraphs [0022] - [0043] * * figure 1 * -----	1-14	
			TECHNICAL FIELDS SEARCHED (IPC)
			H04L
The supplementary search report has been based on the last set of claims valid and available at the start of the search.			
Place of search <b>Munich</b>		Date of completion of the search <b>24 April 2013</b>	Examiner <b>Veloso González, J</b>
<p><b>CATEGORY OF CITED DOCUMENTS</b></p> <p>X : particularly relevant if taken alone  Y : particularly relevant if combined with another document of the same category  A : technological background  O : non-written disclosure  P : intermediate document</p> <p>T : theory or principle underlying the invention  E : earlier patent document, but published on, or after the filing date  D : document cited in the application  L : document cited for other reasons  .....  &amp; : member of the same patent family, corresponding document</p>			

EPO FORM 1503 03/02 (P04C04) 2



Europäisches  
Patentamt  
European  
Patent Office  
Office européen  
des brevets

Application Number  
EP 10 82 2724

### CLAIMS INCURRING FEES

The present European patent application comprised at the time of filing claims for which payment was due.

- Only part of the claims have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due and for those claims for which claims fees have been paid, namely claim(s):
- No claims fees have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due.

### LACK OF UNITY OF INVENTION

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

see sheet B

- All further search fees have been paid within the fixed time limit. The present European search report has been drawn up for all claims.
- As all searchable claims could be searched without effort justifying an additional fee, the Search Division did not invite payment of any additional fee.
- Only part of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the inventions in respect of which search fees have been paid, namely claims:
- None of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims, namely claims:
- The present supplementary European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims (Rule 164 (1) EPC).



**LACK OF UNITY OF INVENTION  
SHEET B**

Application Number  
EP 10 82 2724

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

1. claims: 1-14

Network of peers holding portions of data of a data server where the peers transmit data to a client device upon request.

---

2. claims: 15-20

Method for a requesting client device to choose an agent device to mediate in the delivery of data on a peer to peer network.

---

**ANNEX TO THE EUROPEAN SEARCH REPORT  
ON EUROPEAN PATENT APPLICATION NO.**

EP 10 82 2724

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on  
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

24-04-2013

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008109446 A1	08-05-2008	CN 101179389 A US 2008109446 A1	14-05-2008 08-05-2008
-----			
US 2007156855 A1	05-07-2007	NONE	
-----			
US 2008235391 A1	25-09-2008	CN 101641685 A EP 2145257 A1 JP 2010522472 A US 2008235391 A1 US 2011191419 A1 US 2011191420 A1 WO 2008118252 A1	03-02-2010 20-01-2010 01-07-2010 25-09-2008 04-08-2011 04-08-2011 02-10-2008
-----			

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82



# PCT

## INTERNATIONAL SEARCH REPORT

(PCT Article 18 and Rules 43 and 44)

Applicant's or agent's file reference 17459-8101	<b>FOR FURTHER ACTION</b>	see Form PCT/ISA/220 as well as, where applicable, item 5 below.
International application No. PCT/US 10/51881	International filing date ( <i>day/month/year</i> ) 08 October 2010 (08.10.2010)	(Earliest) Priority Date ( <i>day/month/year</i> ) 08 October 2009 (08.10.2009)
Applicant HOLA NETWORKS, LTD.		

This international search report has been prepared by this International Searching Authority and is transmitted to the applicant according to Article 18. A copy is being transmitted to the International Bureau.

This international search report consists of a total of 3 sheets.

It is also accompanied by a copy of each prior art document cited in this report.

**1. Basis of the report**

a. With regard to the **language**, the international search was carried out on the basis of:

the international application in the language in which it was filed.

a translation of the international application into \_\_\_\_\_ which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1(b)).

b.  This international search report has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43.6bis(a)).

c.  With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, see Box No. I.

2.  **Certain claims were found unsearchable** (see Box No. II).

3.  **Unity of invention is lacking** (see Box No. III).

4. With regard to the **title**,

the text is approved as submitted by the applicant.

the text has been established by this Authority to read as follows:

5. With regard to the **abstract**,

the text is approved as submitted by the applicant.

the text has been established, according to Rule 38.2, by this Authority as it appears in Box No. IV. The applicant may, within one month from the date of mailing of this international search report, submit comments to this Authority.

6. With regard to the **drawings**,

a. the figure of the **drawings** to be published with the abstract is Figure No. 3

as suggested by the applicant.

as selected by this Authority, because the applicant failed to suggest a figure.

as selected by this Authority, because this figure better characterizes the invention.

b.  none of the figures is to be published with the abstract.

**Box No. IV Text of the abstract (Continuation of item 5 of the first sheet)**

A system for increasing network communication speed for users, while lowering network congestion for content owners and ISPs. The system employs network elements including an acceleration server, clients, agents, and peers, where communication requests generated by applications are intercepted by the client on the same machine. The IP address of the server is transmitted to the acceleration server, which provides a list of agents to use for this IP address. One or more of the agents respond with a list of peers that have previously seen some or all of the content which is the response to this request. The client then downloads the data from these peers in parts and in parallel, thereby speeding up the Web transfer.

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC(8) - G01R 31/08, G06F 11/00, G08C 15/00 (2010.01) USPC - 370/230 According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) USPC: 370/230  Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched USPC: 370/229, 230, 351, 389, 400, 401, 464, 465, 468; 709/217, 219; 707/705, 781, 783, 784, 785, 786, 788, E17.031, E17.12, 999.002, 999.01; 718/100, 102 (keyword limited - see search terms below)  Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) PubWEST (PGPB, USPT, USOC, EPAB, JPAB); GOOGLE; Google Scholar Terms: internet, protocol, speed, faster, content, server, web, request, agent, module, url, http, header, packet, network, fetch, proxy, chunk, portion, cooperate, ftp, smtp, load, geography, optimal, maximize, cache, server, checksum.		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2003/0174648 A1 (WANG et al.) 18 September 2003 (18.09.2003) entire document, especially abstract, para [0003], [0007], [0014], [0030], [0031], [0033], [0043], [0092], [0204], [0219], [0232], [0246], [0417], [0573], [0598], [0610], [0823], [1006], [1146], [1150], [1156], [1177].	1-45
Y	US 2008/0008089 A1 (BORNSTEIN et al.) 10 January 2008 (10.01.2008) entire document, especially abstract, para [0007], [0010], [0011], [0030], [0033], [0084], [0086].	1-45
A	US 2004/0088646 A1 (YEAGER et al.) 06 May 2004 (06.05.2004) entire document, especially abstract, para [0009], [0010], [0077], [0078], [0081], [0084], [0088].	1-45
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/>		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 22 November 2010 (22.11.2010)		Date of mailing of the international search report <b>09 DEC 2010</b>
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201		Authorized officer: Lee W. Young  PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

## Electronic Patent Application Fee Transmittal

<b>Application Number:</b>				
<b>Filing Date:</b>				
<b>Title of Invention:</b>	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION			
<b>First Named Inventor/Applicant Name:</b>	Derry Shribman			
<b>Filer:</b>	Yehuda Binder			
<b>Attorney Docket Number:</b>	HOLA-005-US4			
Filed as Small Entity				
<b>Filing Fees for Track I Prioritized Examination - Nonprovisional Application under 35 USC 111(a)</b>				
Description	Fee Code	Quantity	Amount	Sub-Total in USD(\$)
<b>Basic Filing:</b>				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	75	75
UTILITY SEARCH FEE	2111	1	330	330
UTILITY EXAMINATION FEE	2311	1	380	380
REQUEST FOR PRIORITIZED EXAMINATION	2817	1	2000	2000
<b>Pages:</b>				
<b>Claims:</b>				
CLAIMS IN EXCESS OF 20	2202	9	50	450
<b>Miscellaneous-Filing:</b>				

Description	Fee Code	Quantity	Amount	Sub-Total in USD(\$)
PUBL. FEE- EARLY, VOLUNTARY, OR NORMAL	1504	1	0	0
PROCESSING FEE, EXCEPT PROV. APPLS.	2830	1	70	70
<b>Petition:</b>				
<b>Patent-Appeals-and-Interference:</b>				
<b>Post-Allowance-and-Post-Issuance:</b>				
<b>Extension-of-Time:</b>				
<b>Miscellaneous:</b>				
<b>Total in USD (\$)</b>				<b>3305</b>

## Electronic Acknowledgement Receipt

<b>EFS ID:</b>	32393125
<b>Application Number:</b>	15957945
<b>International Application Number:</b>	
<b>Confirmation Number:</b>	7917
<b>Title of Invention:</b>	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION
<b>First Named Inventor/Applicant Name:</b>	Derry Shribman
<b>Customer Number:</b>	131926
<b>Filer:</b>	Yehuda Binder
<b>Filer Authorized By:</b>	
<b>Attorney Docket Number:</b>	HOLA-005-US4
<b>Receipt Date:</b>	20-APR-2018
<b>Filing Date:</b>	
<b>Time Stamp:</b>	07:33:36
<b>Application Type:</b>	Utility under 35 USC 111(a)

### Payment information:

Submitted with Payment	yes
Payment Type	DA
Payment was successfully received in RAM	\$3305
RAM confirmation Number	042018INTEFSW00008084506726
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

<b>File Listing:</b>					
<b>Document Number</b>	<b>Document Description</b>	<b>File Name</b>	<b>File Size(Bytes)/ Message Digest</b>	<b>Multi Part /.zip</b>	<b>Pages (if appl.)</b>
1	TrackOne Request	sb0424.pdf	140370	no	2
			bad6222ea626436f20a0d09262275157c8b223ed		
<b>Warnings:</b>					
<b>Information:</b>					
2	Specification	Spec.pdf	3057660	no	33
			5b9f3ae3e121bf653db2f7921f15a64c3ec77b53		
<b>Warnings:</b>					
<b>Information:</b>					
3	Drawings-only black and white line drawings	Drawings-14025109.pdf	258016	no	15
			fa77c1f6c6442922c86a9e390fa2a1ba55639fe3		
<b>Warnings:</b>					
<b>Information:</b>					
4	Application Data Sheet	ADS.pdf	1823472	no	9
			3318c3858daae06245fad40a8362282c1dc5fc5d		
<b>Warnings:</b>					
<b>Information:</b>					
5	Oath or Declaration filed	Signed-oath-Derry.pdf	862135	no	2
			6a2192d0180f49adf2112bf067a659cb3f1f8144		
<b>Warnings:</b>					
<b>Information:</b>					
6	Oath or Declaration filed	Signed-oath-Ofer.pdf	871562	no	2
			164fbaa7776b17398cb08bf75bfa2cac4e62d2d		
<b>Warnings:</b>					
<b>Information:</b>					

7	Power of Attorney	Signed-POA.pdf	888347	no	2
			74a4d32b0579bb0e33a0f6ea26cf96059db06d46		
<b>Warnings:</b>					
<b>Information:</b>					
8	Claims	Claims-Tunnel.pdf	103352	no	6
			8241ec969ea347d0a5388bd8f0367c3fc90fbc38		
<b>Warnings:</b>					
<b>Information:</b>					
9	Abstract	Abstract.pdf	119286	no	1
			23f763b6b527ef1a4ee2e1116d2eac110f917c06		
<b>Warnings:</b>					
<b>Information:</b>					
10	Power of Attorney	Transmittal-letter.pdf	456932	no	2
			f9bcf858c57b0d0e3f54685e975d356584b41333		
<b>Warnings:</b>					
<b>Information:</b>					
11	Information Disclosure Statement (IDS) Form (SB08)	IDS.pdf	1037437	no	8
			03c57837ce4ca170bd1ce47f6e04ed4c71674797		
<b>Warnings:</b>					
<b>Information:</b>					
12	Information Disclosure Statement (IDS) Form (SB08)	IDS2.pdf	1036361	no	7
			88c5fa37ce2739924d60266233d8f46f0a518aec		
<b>Warnings:</b>					
<b>Information:</b>					
13	Non Patent Literature	EP-SR.pdf	98640	no	4
			e1c76ee63a0e4422c734cb5c1bd1e4cbcd033232		
<b>Warnings:</b>					
<b>Information:</b>					



14	Non Patent Literature	WO2011044402-ISR.pdf	129414	no	3
			57c68e1f16f744e4d0bdfb6bc9924ac6d44a1adff		
<b>Warnings:</b>					
<b>Information:</b>					
15	Fee Worksheet (SB06)	fee-info.pdf	41369	no	2
			78e9ad00e063712dca4d32be9020aed62235fc3d		
<b>Warnings:</b>					
<b>Information:</b>					
<b>Total Files Size (in bytes):</b>			10924353		
<p><b>This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.</b></p> <p><b><u>New Applications Under 35 U.S.C. 111</u></b>  <b>If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.</b></p> <p><b><u>National Stage of an International Application under 35 U.S.C. 371</u></b>  <b>If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.</b></p> <p><b><u>New International Application Filed with the USPTO as a Receiving Office</u></b>  <b>If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.</b></p>					

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

<b>POWER OF ATTORNEY OR REVOCATION OF POWER OF ATTORNEY WITH A NEW POWER OF ATTORNEY AND CHANGE OF CORRESPONDENCE ADDRESS</b>	Application Number	
	Filing Date	
	First Named Inventor	Dery Shribman
	Title	SYSTEM PROVIDING FASTER AND MORE EFFICIENT
	Art Unit	
	Examiner Name	
	Attorney Docket Number	HOLA-005-US3

I hereby revoke all previous powers of attorney given in the above-identified application.

A Power of Attorney is submitted herewith.

**OR**

I hereby appoint Practitioner(s) associated with the following Customer Number as my/our attorney(s) or agent(s) to prosecute the application identified above, and to transact all business in the United States Patent and Trademark Office connected therewith:

131,926

**OR**

I hereby appoint Practitioner(s) named below as my/our attorney(s) or agent(s) to prosecute the application identified above, and to transact all business in the United States Patent and Trademark Office connected therewith:

Practitioner(s) Name	Registration Number

Please recognize or change the correspondence address for the above-identified application to:

The address associated with the above-mentioned Customer Number.

**OR**

The address associated with Customer Number:

Firm or Individual Name

Address

City State Zip

Country

Telephone Email

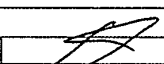
I am the:

Applicant/Inventor.

**OR**

Assignee of record of the entire interest. See 37 CFR 3.71.  
 Statement under 37 CFR 3.73(b) (Form PTO/SB/96) submitted herewith or filed on \_\_\_\_\_

**SIGNATURE of Applicant or Assignee of Record**

Signature		Date	March 13, 2018
Name	Dery Shribman	Telephone	
Title and Company	CEO of HOLA NEWCO LTD.		

**NOTE:** Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below\*.

\*Total of 1 forms are submitted.

This collection of information is required by 37 CFR 1.31, 1.32 and 1.33. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 3 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

## Privacy Act Statement

**The Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

**DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)**

<b>Title of Invention</b>	<b>SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION</b>
---------------------------	--

As the below named inventor, I hereby declare that:

This declaration is directed to:  The attached application, or  
 United States application or PCT international application number \_\_\_\_\_  
filed on \_\_\_\_\_.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

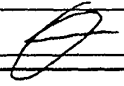
I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

**WARNING:**

*Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.*

**LEGAL NAME OF INVENTOR**

Inventor: Derry Shribman Date (Optional): \_\_\_\_\_

Signature: 

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

This collection of information is required by 35 U.S.C. 115 and 37 CFR 1.63. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 1 minute to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

## Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the *principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.*

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

**DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)**

<b>Title of Invention</b>	<b>SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION</b>
---------------------------	--

As the below named inventor, I hereby declare that:

This declaration is directed to:  The attached application, or  
 United States application or PCT international application number \_\_\_\_\_  
filed on \_\_\_\_\_

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

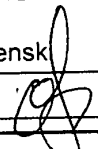
I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

**WARNING:**

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

**LEGAL NAME OF INVENTOR**

Inventor: Ofer Vilensk Date (Optional) : \_\_\_\_\_

Signature: 

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

This collection of information is required by 35 U.S.C. 115 and 37 CFR 1.63. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 1 minute to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

*If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.*

## Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

Table with 7 columns: APPLICATION NUMBER, FILING or 371(c) DATE, GRP ART UNIT, FIL FEE REC'D, ATTY DOCKET NO, TOT CLAIMS, IND CLAIMS. Row 1: 15/957,945, 04/20/2018, 2447, 1235, HOLA-005-US4, 29, 1

CONFIRMATION NO. 7917

FILING RECEIPT

131926
May Patents Ltd. c/o Dorit Shem-Tov
P.O.B 7230
Ramat-Gan, 5217102
ISRAEL



Date Mailed: 05/14/2018

Receipt is acknowledged of this non-provisional patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections

Inventor(s)

Derry Shribman, Tel Aviv, ISRAEL;
Ofer Vilenski, Moshav Hadar Am, ISRAEL;

Applicant(s)

HOLA NEWCO LTD., Netanya, ISRAEL;

Power of Attorney: The patent practitioners associated with Customer Number 131926

Domestic Priority data as claimed by applicant

This application is a CON of 14/025,109 09/12/2013
which is a DIV of 12/836,059 07/14/2010 PAT 8560604
which claims benefit of 61/249,624 10/08/2009

Foreign Applications for which priority is claimed (You may be eligible to benefit from the Patent Prosecution Highway program at the USPTO. Please see http://www.uspto.gov for more information.) - None.

Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: Yes

Permission to Access Search Results: Yes

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 05/14/2018



The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 15/957,945**

**Projected Publication Date:** 08/23/2018

**Non-Publication Request:** No

**Early Publication Request:** No

**\*\* SMALL ENTITY \*\***

**Title**

SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION

**Preliminary Class**

709

**Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications:** No

### **PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES**

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at <http://www.uspto.gov/web/offices/pac/doc/general/index.html>.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, <http://www.stopfakes.gov>. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

**LICENSE FOR FOREIGN FILING UNDER**  
**Title 35, United States Code, Section 184**  
**Title 37, Code of Federal Regulations, 5.11 & 5.15**

**GRANTED**

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign Assets Control, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

**NOT GRANTED**

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

---

***SelectUSA***

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit <http://www.SelectUSA.gov> or call +1-202-482-6800.

**PATENT APPLICATION FEE DETERMINATION RECORD**

Substitute for Form PTO-875

Application or Docket Number  
15/957,945

**APPLICATION AS FILED - PART I**

(Column 1)		(Column 2)	SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
FOR	NUMBER FILED	NUMBER EXTRA	RATE(\$)	FEE(\$)		RATE(\$)	FEE(\$)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	75		N/A	
SEARCH FEE (37 CFR 1.16(k), (j), or (m))	N/A	N/A	N/A	330		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	380		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	29 minus 20 = *	9	x 50 =	450	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	1 minus 3 = *		x 230 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is \$310 (\$155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			0.00			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	1235		TOTAL	

**APPLICATION AS AMENDED - PART II**

(Column 1)		(Column 2)	(Column 3)	SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
AMENDMENT A	CLAIMS REMAINING AFTER AMENDMENT	HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE(\$)	ADDITIONAL FEE(\$)		RATE(\$)	ADDITIONAL FEE(\$)
	Total (37 CFR 1.16(i))	* Minus **	=	x =		OR	x =	
	Independent (37 CFR 1.16(h))	* Minus ***	=	x =		OR	x =	
	Application Size Fee (37 CFR 1.16(s))					OR		
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					OR		
			TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE		
AMENDMENT B	CLAIMS REMAINING AFTER AMENDMENT	HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE(\$)	ADDITIONAL FEE(\$)		RATE(\$)	ADDITIONAL FEE(\$)
	Total (37 CFR 1.16(i))	* Minus **	=	x =		OR	x =	
	Independent (37 CFR 1.16(h))	* Minus ***	=	x =		OR	x =	
	Application Size Fee (37 CFR 1.16(s))					OR		
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					OR		
			TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE		

\* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.  
 \*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".  
 \*\*\* If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".  
 The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

ATTY.'S DOCKET: HOLA-005-US4

In re Application of: ) Confirmation No. 7917  
Derry SHRIBMAN ) Art Unit:  
Appln. No.: 15/957,945 ) Examiner:  
Filed: April 20, 2018 ) Washington, D.C.  
For: SYSTEM PROVIDING FASTER )  
AND MORE EFFICIENT DATA )  
COMMUNICATION ) May 29, 2018

**PRELIMINARY AMENDMENT:**

Honorable Commissioner for Patents  
U.S. Patent and Trademark Office  
Randolph Building, Mail Stop Amendments  
401 Dulany Street  
Alexandria, VA 22314

Sir:

Amendments to the drawings begin on page 2 of this  
paper.

Appln. No. 15/957,945  
Filed April 20, 2018

**Amendments to the drawings**

Submitted herewith is Figure 7 wherein the line quality has been improved.

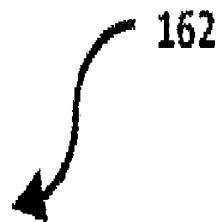
No new matter was added.

Respectfully submitted,

By           / Yehuda Binder /  
Yehuda Binder  
Registration No. 73,612

Tel: +972-54-4444577  
Fax: +972-9-7442619  
e-mail: yehuda@maypatents.com

ACCELERATION DATABASE 164					
166	AGENT IP A	ONLINE/OFFLINE			
>>> INDEXED BY: AGENT IP ADDRESS					
CACHE DATABASE 282					
286	LIST OF URIS:				
288	URL 1				
	290	URL			
	292	URL HTTP HEADERS			
	294	LAST CHECKED ON SERVER			
	296	LAST CHANGED ON SERVER			
	298	LIST OF CHUNKS FOR THIS URL:			
		300	CHUNK 1		
			302	CHUNK CHECKSUM	
			304	CHUNK DATA	
			306	LIST OF PEERS:	
				308	PEER 1
					310 PEER 1 IP ADDRESS
					312 PEER 2 CONNECTION STATUS



**FIG. 7**

## Electronic Acknowledgement Receipt

<b>EFS ID:</b>	32739321
<b>Application Number:</b>	15957945
<b>International Application Number:</b>	
<b>Confirmation Number:</b>	7917
<b>Title of Invention:</b>	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION
<b>First Named Inventor/Applicant Name:</b>	Derry Shribman
<b>Customer Number:</b>	131926
<b>Filer:</b>	Yehuda Binder
<b>Filer Authorized By:</b>	
<b>Attorney Docket Number:</b>	HOLA-005-US4
<b>Receipt Date:</b>	29-MAY-2018
<b>Filing Date:</b>	20-APR-2018
<b>Time Stamp:</b>	11:50:11
<b>Application Type:</b>	Utility under 35 USC 111(a)

### Payment information:

Submitted with Payment	no
------------------------	----

### File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Preliminary Amendment	Preliminary-Amendment.pdf	103501 7f90ad4004d8dd27b352eb38d5f3b9ac7a00fea	no	2

### Warnings:

<b>Information:</b>					
2	Drawings-only black and white line drawings	Corrected_FIG7.pdf	52370	no	1
			b56d80d6e2a6ebcd36986a16a5372c049653e6a7		
<b>Warnings:</b>					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
<b>Information:</b>					
<b>Total Files Size (in bytes):</b>			155871		
<p><b>This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.</b></p> <p><b><u>New Applications Under 35 U.S.C. 111</u></b>  <b>If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.</b></p> <p><b><u>National Stage of an International Application under 35 U.S.C. 371</u></b>  <b>If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.</b></p> <p><b><u>New International Application Filed with the USPTO as a Receiving Office</u></b>  <b>If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.</b></p>					





UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
**United States Patent and Trademark Office**  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/957,945	04/20/2018	Derry Shribman	HOLA-005-US4	7917
131926	7590	05/30/2018	EXAMINER	
May Patents Ltd. c/o Dorit Shem-Tov P.O.B 7230 Ramat-Gan, 5217102 ISRAEL			ART UNIT	PAPER NUMBER
			2459	
			MAIL DATE	DELIVERY MODE
			05/30/2018	PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

<b><i>Decision Granting Request for Prioritized Examination (Track I)</i></b>	<b>Application No.</b> 15/957,945	<b>Applicant(s)</b> Shribman et al.	
	<b>Examiner</b> JoAnne Burke	<b>Art Unit</b> OPET	<b>AIA (First Inventor to File) Status</b> No
<p>1. THE REQUEST FILED <u>20 April 2018</u> IS <b>GRANTED</b> .</p> <p>The above-identified application has met the requirements for prioritized examination</p> <p>A. <input checked="" type="checkbox"/> for an original nonprovisional application (Track I).</p> <p>B. <input type="checkbox"/> for an application undergoing continued examination (RCE).</p> <p>2. <b>The above-identified application will undergo prioritized examination.</b> The application will be accorded special status throughout its entire course of prosecution until one of the following occurs:</p> <p>A. filing a <b><u>petition for extension of time</u></b> to extend the time period for filing a reply;</p> <p>B. filing an <b><u>amendment to amend the application to contain more than four independent claims, more than thirty total claims</u></b>, or a multiple dependent claim;</p> <p>C. filing a <b><u>request for continued examination</u></b> ;</p> <p>D. filing a notice of appeal;</p> <p>E. filing a request for suspension of action;</p> <p>F. mailing of a notice of allowance;</p> <p>G. mailing of a final Office action;</p> <p>H. completion of examination as defined in 37 CFR 41.102; or</p> <p>I. abandonment of the application.</p> <p>Telephone inquiries with regard to this decision should be directed to JoAnne Burke at (571)272-4584. In his/her absence, calls may be directed to Petition Help Desk at (571) 272-3282.</p>			
/JOANNE L BURKE/ Paralegal Specialist, OPET			



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

Table with 4 columns: APPLICATION NUMBER (15/957,945), FILING OR 371(C) DATE (04/20/2018), FIRST NAMED APPLICANT (Derry Shribman), ATTY. DOCKET NO./TITLE (HOLA-005-US4)

CONFIRMATION NO. 7917

PUBLICATION NOTICE

131926
May Patents Ltd. c/o Dorit Shem-Tov
P.O.B 7230
Ramat-Gan, 5217102
ISRAEL



Title:SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION

Publication No.US-2018-0241851-A1

Publication Date:08/23/2018

NOTICE OF PUBLICATION OF APPLICATION

The above-identified application will be electronically published as a patent application publication pursuant to 37 CFR 1.211, et seq. The patent application publication number and publication date are set forth above.

The publication may be accessed through the USPTO's publically available Searchable Databases via the Internet at www.uspto.gov. The direct link to access the publication is currently http://www.uspto.gov/patft/.

The publication process established by the Office does not provide for mailing a copy of the publication to applicant. A copy of the publication may be obtained from the Office upon payment of the appropriate fee set forth in 37 CFR 1.19(a)(1). Orders for copies of patent application publications are handled by the USPTO's Public Records Division. The Public Records Division can be reached by telephone at (571) 272-3150 or (800) 972-6382, by facsimile at (571) 273-3250, by mail addressed to the United States Patent and Trademark Office, Public Records Division, Alexandria, VA 22313-1450 or via the Internet.

In addition, information on the status of the application, including the mailing date of Office actions and the dates of receipt of correspondence filed in the Office, may also be accessed via the Internet through the Patent Electronic Business Center at www.uspto.gov using the public side of the Patent Application Information and Retrieval (PAIR) system. The direct link to access this status information is currently https://portal.uspto.gov/pair/PublicPair. Prior to publication, such status information is confidential and may only be obtained by applicant using the private side of PAIR.

Further assistance in electronically accessing the publication, or about PAIR, is available by calling the Patent Electronic Business Center at 1-866-217-9197.

Office of Data Management, Application Assistance Unit (571) 272-4000, or (571) 272-4200, or 1-888-786-0101



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
**United States Patent and Trademark Office**  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/957,945	04/20/2018	Derry Shribman	HOLA-005-US4	7917
131926	7590	09/05/2018	EXAMINER	
May Patents Ltd. c/o Dorit Shem-Tov			NGUYEN, MINH CHAU	
P.O.B 7230			ART UNIT	PAPER NUMBER
Ramat-Gan, 5217102			2459	
ISRAEL			MAIL DATE	DELIVERY MODE
			09/05/2018	PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

<b>Office Action Summary</b>	<b>Application No.</b> 15/957,945	<b>Applicant(s)</b> Shribman et al.	
	<b>Examiner</b> MINH CHAU N NGUYEN	<b>Art Unit</b> 2459	<b>AIA Status</b> No

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTHS FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1)  Responsive to communication(s) filed on 20 April 2018.  
 A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on \_\_\_\_.
- 2a)  This action is **FINAL**.                                    2b)  This action is non-final.
- 3)  An election was made by the applicant in response to a restriction requirement set forth during the interview on \_\_\_\_; the restriction requirement and election have been incorporated into this action.
- 4)  Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims\***

- 5)  Claim(s) 1-29 is/are pending in the application.  
5a) Of the above claim(s) \_\_\_\_ is/are withdrawn from consideration.
- 6)  Claim(s) \_\_\_\_ is/are allowed.
- 7)  Claim(s) 1-29 is/are rejected.
- 8)  Claim(s) \_\_\_\_ is/are objected to.
- 9)  Claim(s) \_\_\_\_ are subject to restriction and/or election requirement

\* If any claims have been determined allowable, you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see [http://www.uspto.gov/patents/init\\_events/pph/index.jsp](http://www.uspto.gov/patents/init_events/pph/index.jsp) or send an inquiry to [PPHfeedback@uspto.gov](mailto:PPHfeedback@uspto.gov).

**Application Papers**

- 10)  The specification is objected to by the Examiner.
- 11)  The drawing(s) filed on See Continuation Sheet is/are: a)  accepted or b)  objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

**Priority under 35 U.S.C. § 119**

- 12)  Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

**Certified copies:**

- a)  All    b)  Some\*\*    c)  None of the:
  - 1.  Certified copies of the priority documents have been received.
  - 2.  Certified copies of the priority documents have been received in Application No. \_\_\_\_.
  - 3.  Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\*\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1)  Notice of References Cited (PTO-892)
- 2)  Information Disclosure Statement(s) (PTO/SB/08a and/or PTO/SB/08b)  
Paper No(s)/Mail Date 04/20/2018.
- 3)  Interview Summary (PTO-413)  
Paper No(s)/Mail Date \_\_\_\_\_.
- 4)  Other: \_\_\_\_\_.

Continuation of Application Papers 11): 20 April 2018

***Notice of Pre-AIA or AIA Status***

The present application is being examined under the pre-AIA first to invent provisions.

**DETAILED ACTION**

This action is responsive to the application 15/957,945 filed on April 20, 2018. Claims 1-29 are pending.

***Claim Rejections - 35 USC § 101***

35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

**Claims 1-29** are rejected under 35 U.S.C. 101 because the claimed invention is not directed to statutory subject matter.

The framework for determining whether a claim invention is directed to patent eligible subject matter under 35 U.S.C. §101 stands as:

**Step 1:** *Determining whether the claims are directed to one of the four patent-eligible subject matter categories (process, machine, manufacture, or composition of matter).*

**Claims 1-27** recite a method comprising steps. Accordingly, claims 1-26 are directed to a process.

**Claim 28** recites an article of manufacture comprising a non-transitory computer readable medium containing computer instructions that when executed by a computer processor to cause the processor to perform the method. Accordingly, claim 28 is directed to an article of manufacture.

**Claim 29** recites an apparatus comprising elements of a system which including non-transitory computer readable medium and computer processor. Accordingly, claim 29 is directed to a machine.

**Step 2A:** *Determining whether the claims whole embrace a judicially recognized exception: laws of nature, natural phenomena, or abstract idea.*

**Claim 1** is directed to “*receiving from second server the first content identifier; sending to the first server over the internet, a request that comprises the first content identifier; receiving, the first content from the first server in response to the sending of the first content identifier; and sending, the first content by the first client device to the second server*”, which recites an abstract idea.

These steps describe the abstract idea which similar to the concept of remotely accessing and retrieving user specified information that have been identified as abstract by the courts in **Int. Vent. V. Erie Indemnity ‘002 patent**.

**Step 2B:** *Do the claim elements, individually or in combination, amount to significantly more than the exception?*

The claims do not include additional elements that are sufficient to amount to significantly more than the abstract idea because the additional computer elements, e.g. “*first and second servers comprises a web server; and first and second client devices*” recited in **claim 1**, which are recited at a high level of generality, provide conventional computer functions that do not add meaningful limits to practicing the abstract idea. There is no indication that the combination of elements improves the functioning of a computer or improves any other



technology. Their collective functions merely provide conventional computer implementation. Merely adding a generic computer, generic computer components, or a programmed computer to perform generic computer functions does not automatically overcome an eligibility rejection.

For the reasons above, **claims 2-29** are rejected under 35 U.S.C. 101 as being directed to non-statutory subject matter. The dependent claims add steps to storing, receiving and sending to selected client device in a list of potential clients, which provided by the server, return the request content. The claims do not include additional elements, alone or in combination, that are sufficient to amount to significantly more than the abstract idea.

### ***Claim Rejections - 35 USC § 103***

The following is a quotation of pre-AIA 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under pre-AIA 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

**Claims 1-29** is/are rejected under pre-AIA 35 U.S.C. 103(a) as being unpatentable over Fang et la. (US 2006/0212542) hereinafter "**Fang**" and in view of Zaid et la. (US 2011/0035503) hereinafter "**Zaid**".

Claim 1

**Fang** teaches a method for use with a first client device, for use with a first server that comprises a web server that is a Hypertext Transfer Protocol (HTTP) server that responds to HTTP requests, the first server stores a first content identified by a first content identifier, and for use with a second server (Fang, figures 2-4), the method by the first client device comprising:

receiving, from the second server, the first content identifier [i.e. the client detects the download event associated with a URL file from sever 332] (Fang, 0037-0038);

sending, to the first server over the Internet, a Hypertext Transfer Protocol (HTTP) request that comprises the first content identifier [i.e. indexing server 331 receives a request/query includes URL of a requested content from a peer client] (Fang, 0026, 0037-0038);

receiving, the first content from the second server over the Internet in response to the sending of the first content identifier [i.e. the client receives the content file by retrieving from the file server 332] (Fang, 0038).

**Fang** fails to teach receiving, the first content from the first server; and sending, the first content by the first client device to the second server [i.e. central server], in response to the receiving of the first content identifier.

However, in an analogous art, **Zaid** teaches receiving, the first content from the first server [i.e. publishing server] and sending, the first content by the first client device to the second server, in response to the receiving of the first content identifier (Zaid, 0150-0152, 0159-0166).

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate the features of receiving, the first content from the first server; and sending, the first content by the first client device to the second server, in response to the receiving of the first content identifier, as disclosed by Zaid, into the teachings of Fang. One would be motivated to do so in order to provide an improvement for an information/communication exchange system via a network.

Claim 2

**Fang in combination with Zaid** teach the method according to claim 1, further comprising storing, by the first client device in response to the receiving from the first server, the first content (Zaid, 0150-0152, 0159-0166), and wherein the sending, of the HTTP request is in response to the receiving of the first content identifier (Fang, 0026, 0062).

Claim 3

**Fang in combination with Zaid** teach the method according to claim 2, further comprising:

receiving, from a second client device, the first content identifier (Fang, 0038, 0080-0086); and

sending, the stored first content by the first client device to the second client device, in response to the receiving of the first content identifier from the second client device (Fang, 0080-0086).

Claim 4

**Fang in combination with Zaid** teach the method according to claim 1, wherein the first or second server is a Transmission Control Protocol / Internet Protocol (TCP/IP) server that communicates over the Internet based on, or according to, using TCP/IP protocol or connection, and wherein the first client device is a Transmission Control Protocol / Internet Protocol (TCP/IP) client that communicates respectively with the first or second server over the Internet based on, or according to, TCP/IP protocol or connection (Fang, 0024).

Claim 5

**Fang in combination with Zaid** teach the method according to claim 1, wherein the first client device communicates over the Internet based on, or according to, one out of UDP, DNS, TCP, FTP, POP#, SMTP, or SQL standards (Fang, 0024, 0026).

Claim 6

**Fang in combination with Zaid** teach the method according to claim 1, wherein the first content comprises web-page, audio, or video content, wherein the first content identifier comprises a Uniform Resource Locator (URL), and wherein the method further comprising executing, by the first client device, a web browser application or an email application (Fang, 0017, 0033).

### Claim 7

**Fang in combination with Zaid** teach the method according to claim 1, for use with a third server that comprises a web server that is Hypertext Transfer Protocol (HTTP) server, the third server responds to HTTP requests and stores a second content identified by a second content identifier, the method by the first client device further comprising:

identifying, an HTTP request for the second content (Fang, 0037-0038);

sending, to the second server over the Internet in response to the identifying, the second content identifier and a criterion (Fang, 0026, 0037-0038); and

receiving, over the Internet in response to the sending, from a second client device selected from a plurality of client devices according to the criterion, a part of, or a whole of, the second content (Fang, 0037-0038).

### Claim 8

**Fang in combination with Zaid** teach the method according to claim 7, further comprising executing, by the first client device, a web browser application or an email application, and wherein the identifying comprises intercepting, by a driver in the first client device, the request for the second content respectively from the web browser application or the email application (Fang, 0034-0037).

Claim 9

**Fang in combination with Zaid** teach the method according to claim 7, wherein the criterion is stored in the first client device, and the method further comprising selecting, by the first client device, the second client device from the plurality of client devices, according to the stored criterion (Fang, 0038, 0060, 0081-0085).

Claim 10

**Fang in combination with Zaid** teach the method according to claim 9, wherein the criterion is based on, or comprises, the geographical location of the plurality of client devices, or a response time when communicating with the first client device (Fang, 0004, 0029).

Claim 11

**Fang in combination with Zaid** teach the method according to claim 9, wherein the second client device is the quickest to respond to queries from the first client device (Fang, 0038, 0084).

Claim 12

**Fang in combination with Zaid** teach the method according to claim 9, further comprising sending, by the first client device, a notification message to a device from the plurality of client devices that was not selected as part of the selecting (Fang, 0029, 0052).

Claim 13

**Fang in combination with Zaid** teach the method according to claim 1, further comprising periodically communicating between the second server and the first client device (Fang, 0029).

Claim 14

**Fang in combination with Zaid** teach the method according to claim 13, wherein the periodically communicating comprises exchanging 'keep alive' messages (Fang, 0029, 0078-0079).

Claim 15

**Fang in combination with Zaid** teach the method according to claim 1, further comprising establishing, by the first client device, a Transmission Control Protocol (TCP) connection with the second server using TCP/IP protocol (Fang, 0024).

Claim 16

**Fang in combination with Zaid** teach the method according to claim 1, wherein the first client device is identified by a Media Access Control (MAC) address or a hostname, and wherein the method further comprising sending, by the first client device, during, as part of, or in response to, a start-up of the first client device, a first message to the second server, and wherein the first messages comprises the first IP address, the MAC address, or the hostname (Fang, 0025-0026, 0037).

Claim 17

**Fang in combination with Zaid** teach the method according to claim 16, for use with a first application stored in the first client device and associated with a first version number, wherein the first message comprises the first version number (Zaid, 0058, 0126-0133).



Claim 18

**Fang in combination with Zaid** teach the method according to claim 17, for use with a second application that is a version of the first application, is stored in the second server, and is associated with a second version number, wherein the method further comprising receiving, by the first client device from the second server, in response to the first message, a second message that comprises the second version number (Zaid, 0058, 0126-0133).

Claim 19

**Fang in combination with Zaid** teach the method according to claim 18, wherein the method further comprising downloading over the Internet, by the first client device from the second server, in response to the first message, the second application from the second server, and installing the second application in the first client device as a replacement for the first application (Zaid, 0021, 0058, 0084-0085, 0126-0133).

Claim 20

**Fang in combination with Zaid** teach the method according to claim 1, wherein the first or second server is a Transmission Control Protocol / Internet Protocol (TCP/IP) server, wherein the first client device communicates over the Internet with the first or second server based on, or according to, using TCP/IP protocol or connection (Fang, 0024).

Claim 21

**Fang in combination with Zaid** teach the method according to claim 1, further comprising determining, by the first client device, that the received first content, is valid (Fang, 0065-0066).

Claim 22

**Fang in combination with Zaid** teach the method according to claim 22, wherein the determining is based on the received HTTP header according to, or based on, IETF RFC 2616 (Fang, 0071).

Claim 23

**Fang in combination with Zaid** teach the method according to claim 22, further comprising:

sending, a message over the Internet in response to the determining that the received first content, is not valid; and receiving, over the Internet in response to the sending of the message, from the second server or from a second client device selected from a plurality of client devices, the first content (Fang, 0065-0066).

Claim 24

**Fang in combination with Zaid** teach the method according to claim 1, further comprising storing, operating, or using, a client operating system (Fang, 0040, 0048-0050).

Claim 25

**Fang in combination with Zaid** teach the method according to claim 1, wherein the steps are sequentially executed (Zaid, 0021).

Claim 26

**Fang in combination with Zaid** teach the method according to claim 1, for use with a software application that includes computer instructions that, when executed by a computer processor, cause the processor to perform the sending of the Hypertext Transfer Protocol (HTTP) request, the receiving and storing of the first content, the receiving of the first content identifier, and the sending of the part of, or the whole of, the stored first content (Fang, 0026, 0062), the method is further preceded by:

downloading, by the first client device from the Internet, the software application (Zaid, 0085, 0126-0131, 0142-0147, 0199); and

installing, by the first client device, the downloaded software application (Zaid, 0021, 0084-0085).

Claim 27

**Fang in combination with Zaid** teach the method according to claim 26, wherein the software application is downloaded from the second server (Zaid, 0085, 0126-0131, 0142-0147, 0199).

Claim 28 is corresponding claim of claim 1. Therefore, it is rejected under the same rationale.

Claim 29 is corresponding claim of claim 1. Therefore, it is rejected under the same rationale.

*Correspondence Information*

Any inquiry concerning this communication or earlier communications from the examiner should be directed to MINH CHAU N NGUYEN whose telephone number is (571)272-4242. The examiner can normally be reached on M-F 8am-4pm.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at <http://www.uspto.gov/interviewpractice>.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, JEFFREY NICKERSON can be reached on (571)270-3631. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/MINH CHAU NGUYEN/  
Primary Examiner, Art Unit 2459

<b>Notice of References Cited</b>	Application/Control No. 15/957,945		Applicant(s)/Patent Under Reexamination	
	Examiner MINH CHAU N NGUYEN		Art Unit 2459	Page 1 of 1

**U.S. PATENT DOCUMENTS**

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20060212542-A1	09-2006	Fang; Han	H04L67/104	709/219
*	B	US-20110035503-A1	02-2011	ZAID; SAM	H04L63/0407	709/228
*	C	US-6868453-B1	03-2005	Watanabe; Mitsuhiro	G06F17/30861	707/E17.107
*	D	US-8595786-B2	11-2013	Choi; In Hwan	H04L1/005	725/151
*	E	US-20050097441-A1	05-2005	Herbach, Jonathan D.	G06F21/10	715/229
*	F	US-20060212584-A1	09-2006	Yu; Mingjian	G06F17/30902	709/227
	G					
	H					
	I					
	J					
	K					
	L					
	M					


**FOREIGN PATENT DOCUMENTS**

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

**NON-PATENT DOCUMENTS**

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

\*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)  
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

<b><i>Index of Claims</i></b> 	<b>Application/Control No.</b> 15/957,945	<b>Applicant(s)/Patent Under Reexamination</b> Shribman et al.
	<b>Examiner</b> MINH CHAU N NGUYEN	<b>Art Unit</b> 2459


✓	<b>Rejected</b>
=	<b>Allowed</b>

-	<b>Cancelled</b>
÷	<b>Restricted</b>

N	<b>Non-Elected</b>
I	<b>Interference</b>

A	<b>Appeal</b>
O	<b>Objected</b>

CLAIMS									
<input type="checkbox"/> Claims renumbered in the same order as presented by applicant <input type="checkbox"/> CPA <input type="checkbox"/> T.D. <input type="checkbox"/> R.1.47									
CLAIM		DATE							
Final	Original	08/21/2018							
	1	✓							
	2	✓							
	3	✓							
	4	✓							
	5	✓							
	6	✓							
	7	✓							
	8	✓							
	9	✓							
	10	✓							
	11	✓							
	12	✓							
	13	✓							
	14	✓							
	15	✓							
	16	✓							
	17	✓							
	18	✓							
	19	✓							
	20	✓							
	21	✓							
	22	✓							
	23	✓							
	24	✓							
	25	✓							
	26	✓							
	27	✓							
	28	✓							
	29	✓							

<b><i>Search Notes</i></b> 	<b>Application/Control No.</b> 15/957,945	<b>Applicant(s)/Patent Under Reexamination</b> Shribman et al.
	<b>Examiner</b> MINH CHAU N NGUYEN	<b>Art Unit</b> 2459

<b>CPC - Searched*</b>		
<b>Symbol</b>	<b>Date</b>	<b>Examiner</b>
H04L67/42	08/21/2018	MN
H04L41/046	08/21/2018	MN
H04L67/108	08/21/2018	MN
H04L67/22	08/21/2018	MN

<b>CPC Combination Sets - Searched*</b>		
<b>Symbol</b>	<b>Date</b>	<b>Examiner</b>

<b>US Classification - Searched*</b>			
<b>Class</b>	<b>Subclass</b>	<b>Date</b>	<b>Examiner</b>
709	202	08/21/2018	MN

\* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

<b>Search Notes</b>		
<b>Search Notes</b>	<b>Date</b>	<b>Examiner</b>
Search on EAST	08/21/2018	MN

<b>Interference Search</b>			
<b>US Class/CPC Symbol</b>	<b>US Subclass/CPC Group</b>	<b>Date</b>	<b>Examiner</b>

/MINH CHAU NGUYEN/ Primary Examiner, Art Unit 2459	
---	--



## EAST Search History

## EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	4	((("20060212584") or ("7865585")).PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 22:52
S2	89	servers with (communicat\$4 session\$1 receiv\$4 transmit\$4 provid\$4 send\$4) with ((list\$4 group\$4) near1 clients)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:05
S3	56	S2 and @ad< "20091008"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:06
S4	3	("20080008089").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:14
S5	0	S3 and (server\$1 near3 (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 ((content\$1 page\$1 document\$1) near (identifier\$1 address\$2))) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:18
S6	0	S3 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 ((content\$1 page\$1 document\$1) near (identifier\$1 address\$2))) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:19
S7	7758	(servers and (peers clients)) same ((content\$1 document\$1 page\$1) near3 (deliver\$4 quer\$4 inquir\$4 retriev\$4 search\$4))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:20
S8	77	S7 and (server\$1 with (list\$1 near3	US-PGPUB;	OR	OFF	2018/08/04

EAST Search History

		(node\$1 peer\$1 terminal\$1 device\$1) near3 client\$1))	USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			23:22
S9	1	S8 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 ((content\$1 page\$1 document\$1) near (identifier\$1 address\$2))) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1 node\$1 terminal\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:22
S10	0	S9 and @ad< "20091008"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:22
S11	3	S8 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1 node\$1 terminal\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:23
S12	0	S1 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1 node\$1 terminal\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:25
S13	0	S4 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1 node\$1 terminal\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:25
S14	1	S1 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with (client\$1 peer\$1 node\$1 terminal\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:26
S15	1	S4 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with (client\$1 peer\$1 node\$1 terminal\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:29
S16	3498	(server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 content\$1 page\$1 document\$1) with ((other\$1 second\$4 another differen\$4) near1 (client\$1 peer\$1 node\$1 terminal\$1)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT;	OR	OFF	2018/08/04 23:29

			IBM_TDB			
S17	1298	S16 and @ad<"20091008"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:30
S18	44	S17 and (server\$1 with (list\$1 near3 (node\$1 peer\$1 terminal\$1 device\$1) near3 client\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/04 23:31
S19	5	((("20060212584") or ("20030097408")).PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/05 08:43
S20	3	S19 and (live online geographic\$6 near\$4 close\$3 location\$1 add\$1 adding)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/05 08:44
S21	3	S19 and (live online geographic\$6 near\$4 close\$3 location\$1 add\$1 adding select\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/05 08:47
S22	5	((("20060212584") or ("20030097408")).PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:35
S23	3	S22 and (live online geographic\$6 near\$4 close\$3 location\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:35
S24	3	S22 and (live online status\$2 geographic\$6 near\$4 close\$3 location\$1 add\$1 adding select\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:38
S25	2	S22 and (remov\$4 delet\$4 drop\$4 releas\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO;	OR	OFF	2018/08/06 08:43

EAST Search History

			JPO; DERWENT; IBM_TDB			
S26	3	S22 and (live online status\$2 period\$6 interval\$1 time\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:46
S27	3	S22 and (proxim\$6 geographic\$6 near\$4 close\$3 location\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:50
S28	3	S22 and (criteri\$4 select\$4 proxim\$6 geographic\$6 near\$4 close\$3 location\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:50
S29	1	S22 and (overlap\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:52
S30	2	S22 and (mac host\$name\$1 ip)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:57
S31	1	S22 and (software\$1 version\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 08:58
S32	3	S22 and (stor\$4 using opearting)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 09:00
S33	3	S22 and (stor\$4 using operating)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 09:00
S34	2	S22 and (sequen\$6)	US-PGPUB; USPAT;	OR	OFF	2018/08/06 09:03

EAST Search History

			USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			
S35	2	S22 and (processor\$1 execut\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 09:04
S36	3	S22 and (processor\$1 execut\$4 cpu process\$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/08/06 09:05

**EAST Search History (Interference)**

< This search history is empty >

**9/ 4/ 2018 1:09:33 PM**

**C:\Users\mnguyen2\Documents\EAST\Workspaces\15957950.wsp**

## Bibliographic Data

Application No: 15/957,945

Foreign Priority claimed:  Yes  No

35 USC 119 (a-d) conditions met:  Yes  No  Met After Allowance

Verified and Acknowledged: /MINH CHAU NGUYEN/

Examiner's Signature

Initials

Title:

SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION

---

FILING or 371(c) DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
04/20/2018	709	2459	HOLA-005-US4
<b>RULE</b>			

### APPLICANTS

HOLA NEWCO LTD., Netanya, ISRAEL

### INVENTORS

Derry Shribman Tel Aviv, ISRAEL

Ofer Vilenski Moshav Hadar Am, ISRAEL

### CONTINUING DATA

This application is a CON of 14025109 09/12/2013

14025109 is a DIV of 12836059 07/14/2010 PAT 8560604

12836059 has PRO of 61249624 10/08/2009

### FOREIGN APPLICATIONS

#### IF REQUIRED, FOREIGN LICENSE GRANTED\*\*

05/14/2018

\*\* SMALL ENTITY \*\*

### STATE OR COUNTRY

ISRAEL

### ADDRESS

May Patents Ltd. c/o Dorit Shem-Tov

P.O.B 7230

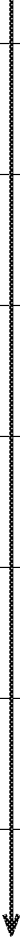
Ramat-Gan, 5217102

ISRAEL

### FILING FEE RECEIVED

\$3,305

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		15/957945
	Filing Date		04/20/2018
	First Named Inventor	Derry Shribman	
	Art Unit	2459	
	Examiner Name	MINH-CHAU NGUYEN	
	Attorney Docket Number	HOLA-005-US4	

U.S. PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
/M. N/ 	1	8479251	B2	2013-07-02	Feinleib et al	
	2	8499059	B2	2013-07-30	Stoyanov	
	3	7970835	B2	2011-28-01	Xerox Corporation	
	4	8832179	B2	2014-09-09	Owen , et al.	
	5	6173330	B1	2001-09-01	Guo , et al.	
	6	8769035	B2	2014-01-07	Resch , et al.	
	7	8171101	B2	2012-05-01	Gladwin , et al.	
	8	7558942	B2	2009-07-07	Chen , et al.	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		15/957945
Filing Date		04/20/2018
First Named Inventor	Derry Shribman	
Art Unit	2459	
Examiner Name	MINH-CHAU NGUYEN	
Attorney Docket Number	HOLA-005-US4	

/M.N/	9	4937781	A	1990-06-26	Lee , et al.
/M.N/	10	7970835	B2	2011-06-28	Robert St. Jacques

If you wish to add additional U.S. Patent citation information please click the Add button.

**U.S.PATENT APPLICATION PUBLICATIONS**

Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
/M.N/	1	20150067819	A1	2015-03-05	Hola Networks Ltd.	
	2	20120254456	A1	2012-10-04	Visharam Zubair et al.	
	3	20080222291	A1	2008-09-11	Weller et al.	
	4	20100235438	A1	2010-09-16	Narayanan et al.	
	5	20120124239	A1	2012-05-17	Shribman et al.	
	6	20130166768	A1	2013-06-27	Thomson Licensing	
	7	20020065930	A1	2002-30-05	Rhodes, David L.	



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		15/957945
Filing Date		04/20/2018
First Named Inventor	Derry Shribman	
Art Unit	2459	
Examiner Name	MINH-CHAU NGUYEN	
Attorney Docket Number	HOLA-005-US4	

/M. N/	8	20030204602	A1	2003-10-30	Hudson Michael D.
	9	20120099566	A1	2012-04-26	Laine; Tuomas ; et al.
	10	20130201316	A1	2013-08-08	BINDER; Yehuda ; et al.
	11	20080125123	A1	2008-05-29	Dorenbosch; Jheroen P. ; et al.
	12	20140301334	A1	2014-10-09	Labranche; Miguel ; et al.
	13	20070239655	A1	2007-10-11	Agetsuma; Masakuni ; et al.
	14	20070226810	A1	2007-09-27	Hotti; Timo
	15	20100094970	A1	2010-04-15	Zuckerman; Gal ; et al.
	16	20020120874	A1	2002-29-08	Shu, Li ; et al.
	17	20100115063	A1	2010-06-05	GLADWIN; S. CHRISTOPHER ; et al.
	18	20100154044	A1	2010-17-06	Manku; Tajinder

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		15/957945
Filing Date		04/20/2018
First Named Inventor	Derry Shribman	
Art Unit	2459	
Examiner Name	MINH-CHAU NGUYEN	
Attorney Docket Number	HOLA-005-US4	


/M. N/	19	20100293555	A1	2010-15-11	VEPSALAINEN; Ari M.
	20	20130272519	A1	2013-17-10	Huang; Lawrence P.
	21	20030115364	A1	2003-06-19	Shu Li et al.
	22	20090217122	A1	2009-27-08	Yokokawa; Takashi ; et al.
	23	20010033583	A1	2001-25-10	Rabenko, Theodore F. ; et al.
	24	20080109446	A1	2008-05-08	Wang Matrix XIN
	25	20020133621	A1	2002-09-19	Talmon Marco et al
	26	20040107242	A1	2004-06-03	John Vert et al
	27	20070073878	A1	2007-03-29	Alfredo C. Issa
	28	20090319502	A1	2009-12-24	Olivier Chalouhi et al
	29	20060212584	A1	2006-09-21	Mingjian Yu et al

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	15/957945
Filing Date	04/20/2018
First Named Inventor	Derry Shribman
Art Unit	2459
Examiner Name	MINH-CHAU NGUYEN
Attorney Docket Number	HOLA-005-US4

If you wish to add additional U.S. Published Application citation information please click the Add button.

**FOREIGN PATENT DOCUMENTS**

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2</sup>	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
/M.N/ 	1	2015034752	WO	A1	2015-03-12	Akamai Technologies INC		
	2	2000/018078	WO	A1	2000-03-30	Sopuch David. J		
	3	0948176	EP	A2	1999-10-06	Siemens Inf &Comm Networks		
	4	2597869	EP	A1	2015-05-29	Sharp Kabushiki Kaisha Osaka-shi		
	5	2010090562	WO	A1	2010-08-12	Telefonaktiebolaget L M Ericsson		
	6	2007280388	JP		2007-25-10	Xerox Corporation		
	7	1020090097034	KR		2009-15-09	KT Corporation		
	8	2343536	RU	C2	2009-10-01	Microsoft Corporation		
	9	101075242	CN	A	2007-11-21	TENGXUN SCIENCE & TECHNOLOGY		

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		15/957945	
	Filing Date		04/20/2018	
	First Named Inventor	Derry Shribman		
	Art Unit	2459		
	Examiner Name	MINH-CHAU NGUYEN		
	Attorney Docket Number	HOLA-005-US4		

/M.N/	10	101179389	CN	A	2008-05-14	Wang Matrix XIN		
-------	----	-----------	----	---	------------	-----------------	--	--

If you wish to add additional Foreign Patent Document citation information please click the Add button

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
/M.N/	1	R. Fielding et al, RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1, June 1999, retrieved from the Internet http://rfc-editor.org [retrieved Apr. 15, 2002] (114 pages)	
	2	"On the Leakage of Personally Identifiable Information via Online Social Networks"-Wills et al, AT&T, Apr. 2009 http://www2.research.att.com/~bala/papers/wosn09.pdf.	
	3	Notice of Preliminary Rejection in KR Application No. 10-2012-7011711 dated July 15, 2016	
	4	KEI SUZUKI, a study on Cooperative Peer Selection Method in P2P Video Delivery, Vol. 109, No. 37, IEICE Technical Report, The Institute of Electronics, Information and Communication Engineers, May 14, 2009	

If you wish to add additional non-patent literature document citation information please click the Add button

**EXAMINER SIGNATURE**

Examiner Signature	/MINH CHAU NGUYEN/	Date Considered	08/21/2018
--------------------	--------------------	-----------------	------------

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT ( Not for submission under 37 CFR 1.99)</b>	Application Number	15/957945
	Filing Date	04/20/2018
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	MINH-CHAU NGUYEN
	Attorney Docket Number	HOLA-005-US4

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2018-04-19
Name/Print	Yehuda BINDER	Registration Number	73612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS

Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (03-15)

Approved for use through 07/31/2016. OMB 0651-0031

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15/957945
	Filing Date	04/20/2018
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	MINH-CHAU NGUYEN
	Attorney Docket Number	HOLA-005-US4

U.S.PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
/M.N/ ↓	1	3922494	A	1975-11-25	Cooper , et al.	
	2	5758195	A	1998-05-26	Balmer; Keith	
	3	6061278	A	2000-05-09	Kato , et al.	
	4	6466470	B1	2002-10-15	Houn Chang	
	5	7865585		2011-01-04	Allen Samuels, et al.	
	6	7120666		2006-10-10	Steven McCanne, et al.	
	7	7203741		2007-04-10	Talmon Marco, et al.	
If you wish to add additional U.S. Patent citation information please click the Add button.						Add
<b>U.S.PATENT APPLICATION PUBLICATIONS</b>						Remove

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		15/957945
Filing Date		04/20/2018
First Named Inventor	Derry Shribman	
Art Unit	2459	
Examiner Name	MINH-CHAU NGUYEN	
Attorney Docket Number	HOLA-005-US4	

Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
/M.N/	1	20030009518	A1	2003-01-09	Harrow, Ivan P. ; et al.	
	2	20030074403	A1	2003-04-17	Harrow, Ivan P. ; et al.	
	3	20140082260	A1	2014-03-20	OH; HakJune ; et al.	
	4	20110314347	A1	2011-12-22	NAKANO; Rikizo ; et al.	
	5	20100329270	A1	2010-12-30	Asati; Rajiv ; et al.	
	6	20100085977	A1	2010-04-08	Khalid; Mohamed ; et al.	
	7	20100066808	A1	2010-03-18	Tucker, Curtis E. ; et al.	
	8	20090279559	A1	2009-11-12	Wong; Yuen Fai ; et al.	
	9	20080025506	A1	2008-01-31	Muraoka; Jochiku	
	10	20040264506	A1	2004-12-30	Furukawa, Rei	



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number		15/957945
Filing Date		04/20/2018
First Named Inventor	Derry Shribman	
Art Unit	2459	
Examiner Name	MINH-CHAU NGUYEN	
Attorney Docket Number	HOLA-005-US4	

/M. N/	11	20020123895	A1	2002-09-05	Sergey Potekhin
	12	20150033001	A1	2015-01-29	Ivanov; Vladimir
	13	20150358648	A1	2015-12-10	Limberg; Allen LeRoy
	14	20160021430	A1	2016-01-21	LaBosco; Mark ; et al.
	15	20110087733	A1	2011-04-14	Derry Shribman; et al.
	16	20030174648	A1	2003-09-18	Mea Wang; et al.
	17	20080008089	A1	2008-01-10	Claudson F. Bornstein; et al.
	18	20040088646	A1	2004-05-06	William J. Yeager; et al.
	19	20030009583	A1	2003-01-09	Chung Chan; et al.
	20	20080235391	A1	2008-09-25	Christopher Painter; et al.
	21	20070156855	A1	2007-07-05	Moses Johnson

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15/957945
	Filing Date	04/20/2018
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	MINH-CHAU NGUYEN
	Attorney Docket Number	HOLA-005-US4

/M.N/	22	20020007413	A1	2002-01-17	JJ Garcia-Luna-Aceves, et al.
↓	23	20030210694	A1	2003-11-13	Suresh Jayaraman, et al.
↓	24	20030200307	A1	2003-10-23	Jyoti Raju, et al.

If you wish to add additional U.S. Published Application citation information please click the Add button

**FOREIGN PATENT DOCUMENTS**

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2j</sup>	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
/M.N/	1	International Search Report issued in PCT Application No. PCT/US2010/051881 dated 09 December 2010	
/M.N/	2	Supplementary European Search Report issued in EP Application No. 10822724 dated 24 April 2013	

If you wish to add additional non-patent literature document citation information please click the Add button

**EXAMINER SIGNATURE**

Examiner Signature	/MINH CHAU NGUYEN/	Date Considered	08/21/2018
--------------------	--------------------	-----------------	------------

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15/957945
	Filing Date	04/20/2018
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	MINH-CHAU NGUYEN
	Attorney Docket Number	HOLA-005-US4

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT ( Not for submission under 37 CFR 1.99)</b>	Application Number	15/957945
	Filing Date	04/20/2018
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	MINH-CHAU NGUYEN
	Attorney Docket Number	HOLA-005-US4

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2018-04-19
Name/Print	Yehuda BINDER	Registration Number	73612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

## EAST Search History

## EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	10677	(servers and (peers clients)) same ((content\$1 document\$1 page\$1) with (deliver\$4 quer\$4 inquir\$4 retriev\$4 search\$4))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/03 23:18
L2	995	1 and (server\$1 with (receiv\$4 quer\$4 transmit\$4 inquir\$4 send\$4) with (url\$1 ((content\$1 page\$1 document\$1) near (identifier\$1 address\$2))) with (client\$1 peer\$1 node\$1 terminal\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/03 23:19
L3	477	2 and @ad<"20091008"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:00
L4	233	3 and (live online) and (geographic\$6 location\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:02
L5	29	4 and (software\$1 with version\$1)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:02
L6	0	5 and (server\$1 with (list\$1 near3 (node\$1 peer\$1 terminal\$1 device\$1) near3 client\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:02
L7	1	4 and (server\$1 with (list\$1 near3 (node\$1 peer\$1 terminal\$1 device\$1) near3 client\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:02
L8	7	3 and (server\$1 with (list\$1 near3 (node\$1 peer\$1 terminal\$1 device\$1) near3 client\$1))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2018/09/04 00:22
L9	7	8 and @ad<"20091008"	US-PGPUB; USPAT;	OR	OFF	2018/09/04 00:28

EAST Search History

			USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB		
--	--	--	--	--	--

**EAST Search History (Interference)**

< This search history is empty >

**9/ 4/ 2018 12:39:24 AM**

**C:\Users\mnguyen2\Documents\EAST\Workspaces\15957945.wsp**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

ATTY.'S DOCKET: HOLA-005-US4

In re Application of:	)	Confirmation No. 7917
	)	
Derry Shribman <i>et al.</i>	)	Art Unit: 2459
	)	
Appln. No.: 15/957,945	)	Examiner: Nguyen, Minh Chau
	)	
Filed: April 20, 2018	)	Washington, D.C.
	)	
For: System providing faster and more efficient data communication	)	October 18, 2018
	)	

**RESPONSE / AMENDMENT:**

Honorable Commissioner for Patents  
U.S. Patent and Trademark Office  
Randolph Building, Mail Stop Amendments  
401 Dulany Street  
Alexandria, VA 22314

Sir:

In response to the Office Action of September 5,  
2018 ("Action"):

**Remarks/Arguments** begin on page 2 of this paper.



**REMARKS / ARGUMENTS**

The examiner's action dated September 5, 2018 ("Action") has been received and its contents carefully noted.

Office Action, pages 2-4

Claims 1-29 are rejected under 35 U.S.C. 101 because the claimed invention lacks patentable utility.

Response.

a. Claim 1 is considered abstract since it describes "*abstract idea which similar to the concept of remotely accessing and retrieving user specified information*". The Action is based on over generalization of the abstract idea and oversimplification of the recited claim functions and is untethered from the actual language of the claims. The Examiner has provided no facts and/or evidence to support the Examiner's determination that the recited structure and mechanism is an abstract idea. It is noted that Alice framework cautions that "*describing the claims at such a high level of abstraction and untethered from the language of the claims all but ensures that the exceptions to § 101 swallow the rule.*" *Enfish, LLC v. Microsoft Corp.*, 822 F.3d 1327, 1337 (Fed. Cir. 2016).

In particular, claim 1 also recites sending a received content to a server, which cannot be part of "*the concept of remotely accessing and retrieving user specified information*". Hence, the Action fails to consider the claims as a whole, while it is noted that the claims should be analyzed "*... in their entirety to ascertain whether their character as a whole is directed to excluded subject matter.*" (Emphasis added) *Internet Patents Corp.*, 790 F.3d at 1346. Further, the information exchanged over the network relates to routing or

handling OTHER information, which is similar to the case of *Enfish*, 822 F.3d at 1336, 118 USPQ2d at 1689, where claims to self-referential table for a computer database were not directed to an abstract idea. It is noted that the claims here are even less abstract since the steps involves not only a single generic computer, but few types of devices (servers / clients) communicating over a network.

b. The rejection is based on the case of "*Int. Vent. V. Erie Indemnity '002 patent*". It is noted that this case (*Intellectual Ventures I LLC v. Erie Indemnity Co.*, 850 F.3d 1315, 1327 (Fed. Cir. 2017)) involved organizing and accessing records through the creation of an index-searchable database (i.e., locating information in a database). Claim 1 is not involved in any database in general, and any organizing and accessing records in particular, hence this case is not analogous to the claims herein. Specifically, the claim discloses a server receiving information from another server via a client device, which is unique and solves a specific problem such as anonymity when fetching information.

Hence, in light of the specification's description of the problem and the inventors' solution, the claimed invention, as described in paragraphs 0004-0012 of the corresponding publication 2018/0241851, solves a problem of Internet congestion, faster and more efficient content transport by improving the operation of peer-to-peer networking arrangement using a management server, in contrast to '*creating and using an index*', which is the heart of the invention - "*the heart of the claimed invention lies in creating and using an index to search for and retrieve data ... an abstract concept*" [Cf. *Intellectual Ventures I v. Erie Indemnity Company*, 850 F.3d, 1315, 1328 (Fed. Cir. 2017)].

Appln. No. 15/957,945  
Reply to Office action of September 5, 2018

Hence, similar to the PTAB decision in appeal 2017-011163 dated May 9, 2018, the Examiner's interpretation of the claims as being directed to an abstract idea of creating an index and using that index to search for and retrieve data is an oversimplification of the claims, as the claims do not even mention the creation of an index or the use of such an index or searching and retrieving data, not do the claim mention the words 'index' or 'search'.

c. As admitted in the Action, the claims involve specific networking of physical elements such as servers and clients, connected via various networks forming a specific structure and relationships, which are physical apparatuses, and are NOY a 'generic computer' as stated in the Action. Under Bilski's MoT test, a claimed process can be patent-eligible under § 101 if: (1) **it is tied to a particular machine or apparatus;** or (2) the process transforms a particular article into a different state or thing." (See Bilski, 545 F.3d at 954 (citing Gottschalk v. Benson, 409 U.S. 63, 70 (1972))).

d. The Action states that the arrangement claimed provides '*conventional computer functions*', '*conventional computer implementation*', '*generic computer*', '*generic computer components*', or a *programmed computer*'. However, the Examiner does not sufficiently establish that the "ordered combination" of the recited elements also fails to "'transform the nature of the claim' into a patent-eligible application." Alice, 134 S. Ct. at 2355. "[A]n inventive concept can be found in the non-conventional and nongeneric arrangement of known, conventional pieces," even if these pieces constitute generic computer-related components. Bascom Global Internet v. AT&T Mobility LLC, 827 F.3d 1341, 1350 (Fed. Cir. 2016). Specifically, the

claimed components as a combination perform functions that are not merely generic - It is respectfully submitted that the conventional arrangement involves fetching data by a client device from a server device, while the claims disclose a server receiving information from another server via a client device, which is unique and solves a specific problem such as anonymity when fetching information.

e. It is noted that *"For computer-implemented inventions like the claimed invention, the question in the second step is whether the computer implementation of the abstract idea involves 'more than performance of 'well-understood, routine, [and] conventional activities previously known to the industry.'"* Content Extraction & Transmission LLC v. Wells Fargo Bank. It is noted that *"Whether something is well-understood, routine, and conventional to a skilled artisan at the time of the patent is a factual determination."* Berkheimer v. HP Inc., 881 F.3d 1360, 1369 (Fed. Cir. 2018), and the Action provides no factual evidence regarding to *"well-understood, routine, and conventional"*, as required by USPTO memorandum dated April 19, 2018, entitled: *"Changes in Examination Procedure Pertaining to Subject Matter Eligibility, Recent Subject Matter Eligibility Decision (Berkheimer v. HP, Inc.)"*.

f. It is respectfully submitted that in a decision of Appeal 2017-010768 dated May 4, 2018 (from application 14/288,506), claims were similarly found to be patent eligible being *"directed to a technical improvement in the communication between computers"*. Similarly, in a decision of Appeal 2017-007566 dated May 17, 2018 (from application 14/268,145), claims were similarly found to be patent eligible since they

Appln. No. 15/957,945  
Reply to Office action of September 5, 2018

'facilitates the cooperation of "multiple different kind of computers."' , and using intermediate computers.

g. It is further noted that software is not automatically an abstract idea, even if performance of a software task involves an underlying mathematical calculation or relationship. See, e.g., *Thales Visionix, Inc. v. United States*, 850 F.3d 1343, 121 USPQ2d 1898, 1902 ("That a mathematical equation is required to complete the claimed method and system does not doom the claims to abstraction.", and also in MPEP §2106.04(a) Abstract Ideas [R-08.2017]. Further, although the claims recite 'servers and devices' the individual steps of the method are **NOT BEING** capable of being performed by a person, either mentally or with paper and pencil, which renders the method not abstract. [See *CyberSource Corp. v. Retail Decisions, Inc.*, 654 F.3d 1366, 1372 (Fed. Cir. 2011)].

h. On page 6 of the Action, the Action admits that the claims provides: "... **an improvement for an information/communication exchange system in a network**" (Emphasis added). Hence, the Action itself admits to various improvements in various technologies and fields, and to solve a problem unique to the Internet and computer technology and, thus, amounts to significantly more than an abstract idea. This is similar to *DDR Holdings, LLC v. Hotels.com, L.P.*, 773 F.3d 1245, 1257-59, 113 USPQ2d 1097, 1106-07 (Fed. Cir. 2014), the claims are admitted by the Office to be directed to a "solution [that] is necessary rooted in computer technology in order to overcome a problem specifically arising in the realm of computer networks".

Office Action, pages 5-15

Claims 1-29 are rejected under 35 U.S.C. 103 as being unpatentable over Fang *et al.* (US 2006/0212542 - hereinafter "Fang"), in view of Zaid *et al.* (US 2011/0035503 - hereinafter "Zaid").

General.

The Action is unclear at least since it is unclear from the rejection what elements of Fang (and also Zaid) are equated to the claim recited elements. For example, it is not clear if the claimed HTTP server is equated to server 332 or server 331 in Figure 3 of the Fang reference. It is noted that *"A rejection must be set forth in sufficiently articulate and informative manner as to meet the notice requirement of § 132, such as by identifying where or how each limitation of the rejected claims is met by the prior art references."* In re Jung, 637 F.3d 1356, 1363 (Fed. Cir. 2011); see also 37 C.F.R. § 1.104(c)(2) (*"When a reference is complex or shows or describes inventions other than that claimed by the applicant, the particular part relied on must be designated as nearly as practicable. The pertinence of each reference, if not apparent, must be clearly explained and each rejected claim specified."*); Gechter v. Davidson 116 F.3d 1454, 1460 (Fed. Cir. 1997) (*PTO must create a record that includes "specific fact findings for each contested limitation and satisfactory explanations for such findings."*).

Non-analogous references.

The Applicant submits that the cited references are directed towards respectively different fields and purposes, and are based on respectively different structures, and thus are not analogous to one another and cannot logically be combined. The Action does not explain, as required by the

rules, what is the reasoning behind the combining of the references, and why these references are analogous to each other. If the Action is based on the reference being in the same field, it is hereby requested that the definition of the field of both the cited references will be included in the next Action. If the Action is based on the reference being pertinent to the same problem, it is hereby requested that the definition of the problem will be included in the next Action.

Unclear combination.

The Action merely state that the missing limitation of the recited claim may be added to the Fang reference. It is not clear HOW this limitation is added to the Fang described system.

Improper rationale for combining the references

The rationale for combining the Fang and Zaid references is stated as: "... to provide an improvement for an information/communication exchange system via a network.". It is noted that this rationale is conclusory and is detached from the specific references, and is not according to MPEP 2143 teaching that: "Any rationale employed must provide **a link** between the factual findings and the legal conclusion of obviousness." (Emphasis added). It is not clear WHY adding the missing limitation provide ANY improvement to the Fang system, and WHAT is the improvement obtained. The rules further require an explicit motivation to combine because "the 'improvement' is technology independent and the combination of references results in a product or process that is more desirable, for example because it is stronger, cheaper, cleaner, faster, lighter, smaller, more durable, or more efficient." DyStar

Appln. No. 15/957,945  
Reply to Office action of September 5, 2018

Textilfarben GmbH & Co. Deutsch/and KG v. CH Patrick Co., 464  
F.3d 1356, 1367 (Fed. Cir. 2006).

Regarding claim 1.

a. Limitation not taught by the Fang reference.

Claim 1 recites the limitation of *"receiving, from the second server, the first content identifier"*. The rejection is based on paragraphs 0037-0038 that teach (according to the rejection) - *"... the client detects the download event associated with a URL file from server 332"*. It is respectfully noted that while the content may be associated with a server, the detecting a download event is performed in and by the client itself and is not related to any server in general, and in particular does not teach any receiving any information from any server. Further, paragraph 0037 itself discloses that a download event starts with a 'click', hence initiated locally by a user and NOT initiated by any EXTERNAL event.

b. Limitation not taught by the Fang reference.

Claim 1 recites the limitation of *"sending, to the first server over the Internet, a Hypertext Transfer Protocol (HTTP) request that comprises the first content identifier"*. The rejection is based on paragraphs 0026 and 0037-0038 that teach (according to the rejection) - *"... indexing server 331 receives a request/query includes URL of a requested content from a peer list"*. It is noted that paragraphs 0026 and 0037 are silent regarding any sending of any request to any server.

It seems from the Action that the claimed 'first server' is equated to the 'indexing server 331' of Fang. However, the first server is explicitly claimed as the server that stores the first content, while the 'indexing server 331'



in the Fang reference clearly ONLY stores reference to content locations, but does not include any content itself as recited in the claim.

c. Mistake in claim interpretation

Claim 1 recites the limitation of "receiving, the first content **from the first server** over the Internet in response to the sending of the first content identifier" (Emphasis added). The rejection by mistake quote "receiving, the first content **from the second server** over the Internet" (Emphasis added). It is noted that the claimed 'first server' is equated in the second limitation to the 'indexing server 331' and NOT to the file server 332 as stated in the rejection of this limitation.

Further, the claim explicitly recites that the receiving of the first content "from the first server over the Internet in **response to the sending** of the first content identifier" (Emphasis added). The Action and the Fang reference are silent regarding any such action in response to the sending of the first content identifier.

Regarding claim 2.

a. Claim 2 defines, inter alia, the limitation of "storing, by the first client device in response to the receiving from the first server, the first content". The rejection is based on paragraphs 0150-0152 and 0159-0166 of the Zaid reference. However, the Action and these cited paragraphs are silent regarding any storing of any content in general, and the recited "storing, by the first client device in response to the receiving from the first server, the first content" in particular.

b. The rejection provides no rationale for modifying the Fang reference, serving as the primary reference, to include the recited limitation. The rationale for combining with the Zaid reference in claim 1 rejection involved different limitation that is not relevant to the recited limitation in this claim.

Regarding claims 6 and 8.

Claim 6 (and 8) recites, *inter alia*, the limitation of: *"executing, by the first client device, a web browser application or an email application"*. The rejection only states paragraphs 0017 and 0033 of the Fang reference. However, the Fang reference in general, and the cited paragraphs in particular, are silent regarding any content-handling software application in general, and the executing *"a web browser application or an email application"* as recited in the claim in particular.

Regarding claim 7.

Claim 7 recites, *inter alia*, using a criterion sent from the client device, and using the criterion for selecting a client device - *"... sending, to the second server over the Internet in response to the identifying, the second content identifier **and a criterion**; and receiving, over the Internet in response to the sending, from a second client device **selected from a plurality of client devices according to the criterion, a part of, or a whole of, the second content"*** (Emphasis added). The Fang reference in general, and the cited paragraphs 0026 and 0037-0038 in particular, are silent regarding any criterion in general, and any criterion for selecting client device in particular. To the contrary, the Fang reference only teaches selecting a peer device by a human user (*"... by performing a mouse click"* in paragraph 0037), hence effectively teaches away

from any automatic selecting using a criterion as recited in the claim.

Regarding claim 10.

Claim 10 recites that *"... the criterion is based on, or comprises, the geographical location of the plurality of client devices, or a response time when communicating with the first client device"*. The Fang reference in general, and the cited paragraphs 0004 and 0029 in particular, are silent regarding any criterion in general, and any criterion for selecting client device in particular, and furthermore any geographic related or time related criterion. To the contrary, the Fang reference only teaches selecting a peer device by a human user (*"... by performing a mouse click"* in paragraph 0037), hence effectively teaches away from any automatic selecting using a criterion as recited in the claim.

Regarding claim 11.

Claim 11 recites that *"... the second client device is the quickest to respond to queries from the first client device"*. The Fang reference in general, and the cited paragraphs 0038 and 0084 in particular, are silent regarding any criterion in general, and any criterion for selecting client device in particular, and furthermore selecting any quickest device. To the contrary, the Fang reference only teaches selecting a peer device by a human user (*"... by performing a mouse click"* in paragraph 0037), hence effectively teaches away from any automatic selecting using a criterion as recited in the claim.

Regarding claim 12.

Claim 12 recites the step of "... *sending, by the first client device, a notification message to a device from the plurality of client devices that was not selected as part of the selecting*". The Fang reference in general, and the cited paragraphs 0029 and 0052 in particular, are silent regarding any communication with a non-selected peer in general, and sending any message thereto in particular.

Regarding claims 13-14.

Claim 13 recites the step of "... *periodically communicating between the second server and the first client device.*". The Fang reference in general, and the cited paragraph 0029 in particular, are silent regarding any periodic communication in general, and regarding "*periodically communicating between the second server and the first client device*" in particular. Further, the Fang reference is silent regarding any 'keep alive' messages as recited in claim 14.

Regarding claim 15.

Claim 15 recites the step of "... *establishing, by the first client device, a Transmission Control Protocol (TCP) **connection** with the second server using TCP/IP protocol*" (Emphasis added). While the Fang reference mentions in passing using TCP/IP protocol, the Fang reference in general, and the cited paragraph 0024 in particular, are silent regarding any establishing of a persistent connection in general, and TCP connection in particular.

Regarding claim 17.

The rejection of claim 17 is based on the Zaid reference. However, claim 17 depends from claim 16 that was rejected based on specific passages in the Fang reference, and

Appln. No. 15/957,945  
Reply to Office action of September 5, 2018

the cited paragraphs in the Zaid reference are not relevant to the cited Fang reference. Further, the rejection provides no rationale for modifying the Fang reference, serving as the primary reference, to include the recited limitation. The rationale for combining with the Zaid reference stated in claim 1 rejection involved different limitation that is not relevant to the recited limitation in this claim. Further, while the Zaid reference may disclose versions of software, it is silent regarding including any version related information in any message as recited in the claim.

Appln. No. 15/957,945  
Reply to Office action of September 5, 2018

The absence of a reply to a specific rejection, issue, or comment, does not signify agreement with that rejection, issue, or comment. In addition, because the arguments made above may not be exhaustive, there may be reasons for patentability of any or all pending claims that have not been expressed.

Nothing in this reply should be understood as conceding any issue with regard to any claim, except as specifically stated in this reply, and the amendment of any claims does not necessarily signify concession of unpatentability to the claim before its amendment.

In view of the foregoing, it is requested that all of the rejections be reconsidered and withdrawn and that the claims be considered allowable.

If the above arguments should not now place the application in the condition for allowance, the examiner is invited to call undersigned counsel to resolve any remaining issues.

Respectfully submitted,

By           /Yehuda Binder/            
          Yehuda Binder  
          Registration No. 73,612

Tel: +972-54-4444577  
Fax: +972-9-7442619  
e-mail: yehuda@maypatents.com

## Electronic Acknowledgement Receipt

<b>EFS ID:</b>	34043344
<b>Application Number:</b>	15957945
<b>International Application Number:</b>	
<b>Confirmation Number:</b>	7917
<b>Title of Invention:</b>	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION
<b>First Named Inventor/Applicant Name:</b>	Derry Shribman
<b>Customer Number:</b>	131926
<b>Filer:</b>	Yehuda Binder
<b>Filer Authorized By:</b>	
<b>Attorney Docket Number:</b>	HOLA-005-US4
<b>Receipt Date:</b>	18-OCT-2018
<b>Filing Date:</b>	20-APR-2018
<b>Time Stamp:</b>	06:51:16
<b>Application Type:</b>	Utility under 35 USC 111(a)

### Payment information:

Submitted with Payment	no
------------------------	----

### File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		10-2018-NFOA-reply.pdf	345788 <small>7f953ffc75f900fa246276c185ed89952f2b7800</small>	yes	15

<b>Multipart Description/PDF files in .zip description</b>		
<b>Document Description</b>	<b>Start</b>	<b>End</b>
Amendment/Req. Reconsideration-After Non-Final Reject	1	1
Applicant Arguments/Remarks Made in an Amendment	2	15
<b>Warnings:</b>		
<b>Information:</b>		
<b>Total Files Size (in bytes):</b>		345788
<p><b>This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.</b></p> <p><b><u>New Applications Under 35 U.S.C. 111</u></b>  <b>If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.</b></p> <p><b><u>National Stage of an International Application under 35 U.S.C. 371</u></b>  <b>If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.</b></p> <p><b><u>New International Application Filed with the USPTO as a Receiving Office</u></b>  <b>If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.</b></p>		



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15957945
	Filing Date	2018-04-20
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	NGUYEN, MINH CHAU
	Attorney Docket Number	HOLA-005-US4

U.S.PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	6868453	B1	2005-03-15	Mitsuhiro Watanabe	
	2	8595786	B2	2013-11-26	In Hwan Choi	

If you wish to add additional U.S. Patent citation information please click the Add button.

Add

U.S.PATENT APPLICATION PUBLICATIONS						Remove
Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030097408	A1	2003-05-22	Masahiro Kageyama	
	2	20070100839	A1	2007-05-03	Deok-ho Kim	
	3	20080256175	A1	2008-10-16	Sang-kwon Lee	
	4	20060212542	A1	2006-09-21	Han Fang	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	15957945
Filing Date	2018-04-20
First Named Inventor	Derry Shribman
Art Unit	2459
Examiner Name	NGUYEN, MINH CHAU
Attorney Docket Number	HOLA-005-US4

5	20110035503	A1	2011-02-10	SAM ZAID
6	20050097441	A1	2005-05-05	Jonathan D. Herbach

If you wish to add additional U.S. Published Application citation information please click the Add button.

**FOREIGN PATENT DOCUMENTS**

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2</sup> i	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button.

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
	1		

If you wish to add additional non-patent literature document citation information please click the Add button.

**EXAMINER SIGNATURE**

Examiner Signature	<input type="text"/>	Date Considered	<input type="text"/>
--------------------	----------------------	-----------------	----------------------

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15957945
	Filing Date	2018-04-20
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	NGUYEN, MINH CHAU
	Attorney Docket Number	HOLA-005-US4

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2018-10-18
Name/Print	Yehuda BINDER	Registration Number	73612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

## Electronic Acknowledgement Receipt

<b>EFS ID:</b>	34043388
<b>Application Number:</b>	15957945
<b>International Application Number:</b>	
<b>Confirmation Number:</b>	7917
<b>Title of Invention:</b>	SYSTEM PROVIDING FASTER AND MORE EFFICIENT DATA COMMUNICATION
<b>First Named Inventor/Applicant Name:</b>	Derry Shribman
<b>Customer Number:</b>	131926
<b>Filer:</b>	Yehuda Binder
<b>Filer Authorized By:</b>	
<b>Attorney Docket Number:</b>	HOLA-005-US4
<b>Receipt Date:</b>	18-OCT-2018
<b>Filing Date:</b>	20-APR-2018
<b>Time Stamp:</b>	07:34:03
<b>Application Type:</b>	Utility under 35 USC 111(a)

### Payment information:

Submitted with Payment	no
------------------------	----

### File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS3.pdf	1034587 <small>d551626a22daefa2e7e9a93de94016c78e143093</small>	no	4

### Warnings:

<b>Information:</b>	
<b>Total Files Size (in bytes):</b>	1034587
<p><b>This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.</b></p> <p><b><u>New Applications Under 35 U.S.C. 111</u></b>  <b>If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.</b></p> <p><b><u>National Stage of an International Application under 35 U.S.C. 371</u></b>  <b>If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.</b></p> <p><b><u>New International Application Filed with the USPTO as a Receiving Office</u></b>  <b>If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.</b></p>	

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15957945
	Filing Date	2018-04-20
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	NGUYEN, MINH CHAU
	Attorney Docket Number	HOLA-005-US4

U.S.PATENTS						Remove
Examiner Initial*	Cite No	Patent Number	Kind Code <sup>1</sup>	Issue Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear
	1	7788378		2010-08-31	Ravi T. Rao	
	2	9253164		2016-02-02	Christopher S. Gouge	
	3	7890547	B2	2011-02-15	Timo Hotti	
	4	8832179	B2	2014-09-09	Owen , et al.	
	5	7818430	B2	2010-10-19	Gal Zuckerman	
	6	6154782	A	2000-11-28	NAOHISA KAWAGUCHI	
	7	5577243	A	1996-17-11	Sherwood , et al.	
	8	8135912	B2	2012-13-03	Shribman , et al.	

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	15957945
Filing Date	2018-04-20
First Named Inventor	Derry Shribman
Art Unit	2459
Examiner Name	NGUYEN, MINH CHAU
Attorney Docket Number	HOLA-005-US4

9	8719505	B2	2014-06-05	Shribman , et al.
10	9201808	B2	2015-01-12	Shribman , et al.
11	9990295	B2	2018-06-05	Shribman , et al.

If you wish to add additional U.S. Patent citation information please click the Add button.

Add

**U.S.PATENT APPLICATION PUBLICATIONS**

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code <sup>1</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20080109446	A1	2008-05-08	Matrix Xin Wang	
	2	20110066924	A1	2011-03-17	Gregory Dorso	
	3	20110128911	A1	2011-06-02	Kamel M. Shaheen	
	4	20130157699	A1	2013-06-20	Mohit Talwar	
	5	20130326607	A1	2013-12-05	Liang Feng	
	6	20030204602	A1	2003-30-10	Hudson, Michael D. ; et al.	



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	15957945
Filing Date	2018-04-20
First Named Inventor	Derry Shribman
Art Unit	2459
Examiner Name	NGUYEN, MINH CHAU
Attorney Docket Number	HOLA-005-US4

7	20120124173	A1	2012-17-05	De; Pradipta ; et al.
8	20020069241	A1	2002-06-06	Narlikar, Girija ; et al.
9	20130201316	A1	2013-08-08	BINDER; Yehuda ; et al.
10	20120099566	A1	2012-26-04	Laine; Tuomas ; et al.
11	20120254370	A1	2012-10-04	Utz BACHER
12	20080125123	A1	2008-05-29	Jheroen P. Dorenbosch
13	20140301334	A1	2014-10-09	Miguel Labranche
14	20070239655	A1	2007-10-11	Masakuni Agetsuma
15	20070226810	A1	2007-09-27	Timo Hotti
16	20100094970	A1	2010-04-15	Gal Zuckerman
17	20130007253	A1	2013-01-03	Guohuai Li

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
( Not for submission under 37 CFR 1.99)

Application Number	15957945
Filing Date	2018-04-20
First Named Inventor	Derry Shribman
Art Unit	2459
Examiner Name	NGUYEN, MINH CHAU
Attorney Docket Number	HOLA-005-US4

18	20090037529	A1	2009-02-05	Gilad Armon-Kest
19	20090182843	A1	2009-07-16	Michael G. Hluchyj
20	20060036755	A1	2006-02-16	Ibrahim S. Abdullah
21	20140376403	A1	2014-12-25	Wenqi Shao
22	20050228964	A1	2005-13-10	Sechrest, Stuart ; et al.
23	20080086730	A1	2008-10-04	Vertes; Marc
24	20060259728	A1	2006-16-11	Chandrasekaran; Sashikanth ; et al.
25	20040254907	A1	2004-16-12	Crow, Preston F. ; et al.
26	20050015552	A1	2005-20-01	So, Kimming ; et al.
27	20050022236	A1	2005-01-27	Akihiko Ito; et al.

If you wish to add additional U.S. Published Application citation information please click the Add button.

**FOREIGN PATENT DOCUMENTS**

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15957945
	Filing Date	2018-04-20
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	NGUYEN, MINH CHAU
	Attorney Docket Number	HOLA-005-US4

Examiner Initial*	Cite No	Foreign Document Number <sup>3</sup>	Country Code <sup>2i</sup>	Kind Code <sup>4</sup>	Publication Date	Name of Patentee or Applicant of cited Document	Pages, Columns, Lines where Relevant Passages or Relevant Figures Appear	T <sup>5</sup>
	1	2597869	EP	A1	2013-18-12	Sharp Kk		
	2	2010090562	WO	A1	2010-12-08	Telefonaktiebolaget L M Ericsson (Publ)		
	3	2011068784	WO	A1	2011-09-06	Azuki Systems, Inc		

If you wish to add additional Foreign Patent Document citation information please click the Add button

**NON-PATENT LITERATURE DOCUMENTS**

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T <sup>5</sup>
	1	Screen captures from YouTube video clip entitle "nVpn.net   Double your Safety and use Socks5 + nVpn" 38 pages, last accessed 11/20/2018 < <a href="https://www.youtube.com/watch?v=L0Hct2kSnn4">https://www.youtube.com/watch?v=L0Hct2kSnn4</a> >	
	2	Screen captures from YouTube video clip entitle "Andromeda" 47 pages, publicly known and available as of at least 2011 < <a href="https://www.youtube.com/watch?v=yRRYpFLbKNU">https://www.youtube.com/watch?v=yRRYpFLbKNU</a> >	
	3	SpyEye, <a href="https://www.symantec.com/security-center/writeup/2010-020216-0135-9">https://www.symantec.com/security-center/writeup/2010-020216-0135-9</a> ; <a href="http://seuresql.info/riskyclouds/spyeye-user-manual">http://seuresql.info/riskyclouds/spyeye-user-manual</a> ; known as of at least 2010 (13 pages)	
	4	Screen captures from YouTube video clip entitle "Change Your Country IP Address & Location with Easy Hide IP Software" 9 pages, publicly known and available as of at least 2011, < <a href="https://www.youtube.com/watch?v=ulwkf1sOfdA">https://www.youtube.com/watch?v=ulwkf1sOfdA</a> and <a href="https://www.youtube.com/watch?v=iFEMT-o9DTc">https://www.youtube.com/watch?v=iFEMT-o9DTc</a> >	
	5	CoralCDN ("CoralCDN"), <a href="https://pdos.csail.mit.edu/6.824/papers/freedman-coral.pdf">https://pdos.csail.mit.edu/6.824/papers/freedman-coral.pdf</a> (14 PAGES)	

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number		15957945
	Filing Date		2018-04-20
	First Named Inventor	Derry Shribman	
	Art Unit		2459
	Examiner Name	NGUYEN, MINH CHAU	
	Attorney Docket Number		HOLA-005-US4

6	European Search Report for EP 14182547.1, dated July 30, 2015
7	R. Fielding et al, RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1, June 1999, retrieved from the Internet <a href="http://rfc-editor.org">http://rfc-editor.org</a> [retrieved Apr. 15, 2002]
8	"On the leakage of personally identifiable information via online social networks", Wills et al. AT&T, Apr. 2009 <a href="http://www2.research.att.com/-bala/papers/wosn09.pdf">http://www2.research.att.com/-bala/papers/wosn09.pdf</a>
9	"Slice Embedding Solutions for Distributed Service Architectures" - Esposito et al., Boston University, Computer Science Dept., 10/2011 <a href="http://www.cs.bu.edu/techreports/pdf/2011-025-slice-embedding.pdf">http://www.cs.bu.edu/techreports/pdf/2011-025-slice-embedding.pdf</a>
10	International Search Report of PCT/US2010/034072 dated July 01, 2010
11	YouTube video clip entitled "nVpn.net   Double your Safety and use Socks5 + nVpn" < <a href="https://www.youtube.com/watch?v=L0Hct2kSnn4">https://www.youtube.com/watch?v=L0Hct2kSnn4</a> >
12	YouTube video clip entitled "Andromeda" < <a href="https://www.youtube.com/watch?v=yRRYpFLbKNU">https://www.youtube.com/watch?v=yRRYpFLbKNU</a> >
13	YouTube video clip entitled "Change Your Country IP Address & Location with Easy Hide IP Software" < <a href="https://www.youtube.com/watch?v=ulwkf1sOfdA">https://www.youtube.com/watch?v=ulwkf1sOfdA</a> and <a href="https://www.youtube.com/watch?v=iFEMT-b9DTc">https://www.youtube.com/watch?v=iFEMT-b9DTc</a> >

If you wish to add additional non-patent literature document citation information please click the Add button

**EXAMINER SIGNATURE**

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

<sup>1</sup> See Kind Codes of USPTO Patent Documents at [www.USPTO.GOV](http://www.USPTO.GOV) or MPEP 901.04. <sup>2</sup> Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). <sup>3</sup> For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. <sup>4</sup> Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. <sup>5</sup> Applicant is to place a check mark here if English language translation is attached.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> ( Not for submission under 37 CFR 1.99)	Application Number	15957945
	Filing Date	2018-04-20
	First Named Inventor	Derry Shribman
	Art Unit	2459
	Examiner Name	NGUYEN, MINH CHAU
	Attorney Docket Number	HOLA-005-US4

**CERTIFICATION STATEMENT**

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

**OR**

That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

A certification statement is not submitted herewith.

**SIGNATURE**

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Yehuda Binder/	Date (YYYY-MM-DD)	2019-01-18
Name/Print	Yehuda Binder	Registration Number	73,612

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

## Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.



(11) **EP 2 597 869 A1**

(12) **EUROPEAN PATENT APPLICATION**  
published in accordance with Art. 153(4) EPC

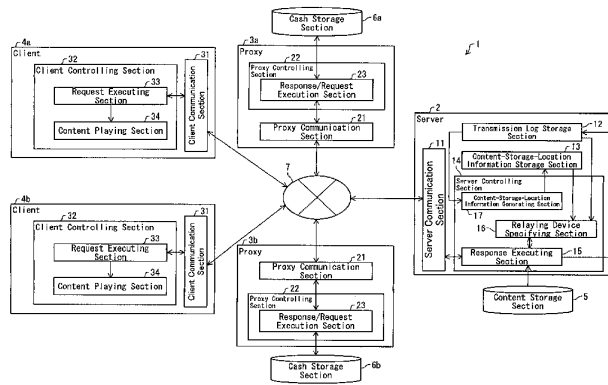
- (43) Date of publication: **29.05.2013 Bulletin 2013/22**
- (44) Application number: **11809620.5**
- (45) Date of filing: **15.07.2011**
- (51) Int Cl.: **H04N 7/173 (2011.01) G06F 13/00 (2006.01)**  
**H04L 12/56 (0000.00)**
- (86) International application number: **PCT/JP2011/066279**
- (87) International publication number: **WO 2012/011450 (26.01.2012 Gazette 2012/04)**

- (84) Designated Contracting States:  
**AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR**
- (30) Priority: **10.03.2011 JP 2011053614**  
**20.07.2010 JP 2010163367**
- (71) Applicant: **Sharp Kabushiki Kaisha**  
**Osaka-shi, Osaka 545-8522 (JP)**
- (72) Inventors:  
• **KANEKO, Takashi**  
**Osaka 545-8522 (JP)**
- **TAKAHASHI, Maki**  
**Osaka 545-8522 (JP)**
- **ITOH, Norio**  
**Osaka 545-8522 (JP)**
- **OGISAWA, Yoshiaki**  
**Osaka 545-8522 (JP)**
- **TOKUMO, Yasuaki**  
**Osaka 545-8522 (JP)**
- **WATANABE, Shuhichi**  
**Osaka 545-8522 (JP)**
- (74) Representative: **Müller - Hoffmann & Partner**  
**Patentanwälte**  
**Innere Wiener Strasse 17**  
**81667 München (DE)**

(54) **CONTENT DISTRIBUTION DEVICE, CONTENT PLAYBACK DEVICE, CONTENT DISTRIBUTION SYSTEM, METHOD FOR CONTROLLING A CONTENT DISTRIBUTION DEVICE, CONTROL PROGRAM, AND RECORDING MEDIUM**

(57) A content distributing device in the present invention receives a request message to request to transmit content. Then, in a case where a device transmitted the request message is a relaying device for receiving the content thus requested and possessing and transferring the content to another device, the content distributing device transmits the content thus requested to the relaying device, or, in a case where a device transmitted the request message is a content playing device for playing the content thus requested, the content distributing device transmits, to the content playing device, an instruction to acquire the content from a relaying device which has transmitted before the content thus requested.

FIG. 1



EP 2 597 869 A1

**Description**

## Technical Field

**[0001]** The present invention relates to a content distribution service in which a server distributes content to the client in response to a request from a client for playing the content.

## Background Art

**[0002]** There has been conventionally and widely used techniques for providing content, such as moving images, via communication networks. Examples of the techniques encompass a video on demand (VOD) service in which a request is transmitted from a client which plays content and, in response to the request, the content is distributed to the client from a server which manages the content. In the content distribution services such as the VOD, content is provided to clients by means of a streaming method, a download method, or a progressive download method.

**[0003]** The following Patent Literature 1 discloses an example where a content distribution service is provided by means of the progressive download method. The Patent Literature 1 discloses a content distribution service system in which a request for content is transmitted to a server from a client with use of HTTP and the client plays the content in response to the request.

## Citation List

## Patent Literature

**[0004]**

Patent Literature 1: Japanese Patent Application Publication, Tokukai, No. 2005-110244 A (Publication Date: April 21, 2005)

## Summary of Invention

## Technical Problem

**[0005]** Whatever method (the streaming method, the download method, or the progressive download method) is used, it is necessary to continuously transfer a great amount of data to a client from a server in a case where a content distribution service is provided, specifically, different kinds of content are distributed to a plurality of clients. When the number of clients increases and a load of a network, which is used to transmit the data from the server, and a load of the server exceed their limits, the data transfer is, for example, delayed. This decreases quality of the content distribution service.

**[0006]** The present invention has been made in view of the aforementioned problem, and an object of the present invention is to achieve (A) a content distributing

device, (B) a content playing device, (C) a content distributing system, (D) a method for controlling the content distributing device, (E) a controlling program, and (F) a recording medium, each of which, reduces, in a content distributing system in which content is distributed to a client from a server, reduces an increase in load of the server and a network and for data transmission from the server.

## 10 Solution to Problem

**[0007]** In order to achieve aforementioned object, a content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request, in accordance with the present invention, includes: determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; content transmitting means for transmitting, in response to the request, the content thus requested to the relaying device in a case where the determining means determines that the source is a relaying device; content-storage-location information generating means for generating content-storage-location information by associating (A) the content transmitted by the content transmitting means with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and content-acquiring-location instructing means for transmitting, in response to the request, an instruction to the content playing device which is the source in a case where the determining means determines that the source is a content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address that the content-storage-location information associates with the content thus requested or (ii) a content playing device indicated by an address that the content-storage-location information associates with the content thus requested.

**[0008]** In order to achieve aforementioned object, a method for controlling content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request, the method in accordance with the present invention includes: a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; a content transmitting step of transmitting, in response to the request, the content thus requested to the relaying device in a case where it is determined that, in the determining step, the source is a relaying device; a content-storage-location information generating step of generating content-storage-location information by associating (A) the content transmit-



ted in the content transmitting step with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and a content-acquiring-location instructing step of transmitting, in response to the request, an instruction to the content playing device which is the source in a case where it is determined that, in the content-storage-location information generating step, the source is the content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (ii) a content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested.

**[0009]** According to the arrangement, upon receipt of the request from the relaying device, the content transmitting means transmits the content thus requested to the relaying device which is the source, and the content-storage-location information generating means generates the content-storage-location information by associating (A) the content transmitted from the content transmitting means with (B) the address of the relaying device, which is the destination of the content, or the address of the content playing device, to which the content is transferred from the relaying device. Further, upon receipt of the request from the content playing device, the content-acquiring-location instructing means transmits, to the content playing device which is the source, the instruction to acquire the content from (I) the relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (II) the content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested. Here, the relaying device and the content playing device possess the content thus acquired, and the content-storage-location information is information indicating which relaying device or content playing device possesses a content.

**[0010]** That is, the content distributing device associates (A) a content which has been transmitted before with (B) a relaying device or a content playing device which possesses the content, and, upon receipt of a request from a certain content playing device, the content distributing device does not directly transmit the content to the content playing device which is the source, but transmits, to the content playing device which is the source, an instruction to acquire the content from a relaying device or a content playing device which possesses the content thus requested. The content playing device, which is the source, acquires the content thus requested from a designated relaying device or a designated content playing device. Therefore, if the designated relaying device or the designated content playing device possesses the content, it is possible to complete transmission and reception of the content with use of only (A) the content playing device which is the source and

(B) the designated relaying device or the designated content playing device. That is, the content playing device, which is the source, can acquire content, without carrying out a process for transmitting the content.

**[0011]** This makes it possible to reduce (A) a load of a network, which is used to transmit data from the content distributing device, and (B) a load of the content distributing device. Among processes carried out by the content distributing device, the relaying device, and the content playing device, a process for transmitting and receiving the content is a process which applies the heaviest load, and the process applies the heaviest load of the network among the content distributing device, the relaying device, and the content playing device. However, even if, for example, the number of content playing devices is increased and the number of requests to the content distributing devices is therefore increased, it is possible to reduce (A) an increase in load of the network which is used to transmit data from the content distributing device and (B) an increase in load of the content distributing device. Therefore, a large number of content playing devices can acquire contents, without increasing throughput of the content distributing device or capacity of the network.

#### Advantageous Effects of Invention

**[0012]** As described above, a content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request, in accordance with the present invention, includes: determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; content transmitting means for transmitting, in response to the request, the content thus requested to the relaying device in a case where the determining means determines that the source is a relaying device; content-storage-location information generating means for generating content-storage-location information by associating (A) the content transmitted by the content transmitting means with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and content-acquiring-location instructing means for transmitting, in response to the request, an instruction to the content playing device which is the source in a case where the determining means determines that the source is a content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address that the content-storage-location information associates with the content thus requested or (ii) a content playing device indicated by an address that the content-storage-location information associates with the content thus requested.

**[0013]** A method for controlling content distributing de-

vice for transmitting, in response to a request, a content to a source which is a sender of the request, the method in accordance with the present invention includes: a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; a content transmitting step of transmitting, in response to the request, the content thus requested to the relaying device in a case where it is determined that, in the determining step, the source is a relaying device; a content-storage-location information generating step of generating content-storage-location information by associating (A) the content transmitted in the content transmitting step with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and a content-acquiring-location instructing step of transmitting, in response to the request, an instruction to the content playing device which is the source in a case where it is determined that, in the content-storage-location information generating step, the source is the content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (ii) a content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested.

**[0014]** This makes it possible to reduce (A) a load of the network, which is used to transmit data from the content distributing device and (B) a load of the content distributing device.

**[0015]** Additional objects, features, and strengths of the present invention will be made clear by the description below. Further, the advantages of the present invention will be evident from the following explanation in reference to the drawings.

#### Brief Description of Drawings

#### **[0016]**

##### Fig. 1

Fig. 1 is a view illustrating Embodiment 1 of the present invention, and illustrates a schematic configuration of a content distributing system in accordance with Embodiment 1 and main configurations of devices constituting the content distributing system.

##### Fig. 2

Fig. 2 is a view showing an exemplary transmission log stored in a transmission log storage section included in the content distributing system.

##### Fig. 3

Fig. 3 is a view illustrating exemplary content-storage-location information stored in a content-storage-location information storage section included in

the content distributing system.

##### Fig. 4

Fig. 4 is a flowchart showing an exemplary process which is carried out by a server constituting the content distributing system.

##### Fig. 5

Fig. 5 is a flowchart showing an exemplary process which is carried out by a proxy constituting the content distributing system.

##### Fig. 6

Fig. 6 is a flowchart showing an exemplary process which is carried out by a client constituting the content distributing system.

##### Fig. 7

Fig. 7 is a view illustrating an exemplary operation sequence of a content distributing system in Example of Embodiment 1.

##### Fig. 8

Fig. 8 is a view illustrating exemplary HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 8 show HTTP messages of respective requests illustrated in Fig. 7, and (b), (e), and (f) of Fig. 8 show HTTP messages of respective responses illustrated in Fig. 7.

##### Fig. 9

Fig. 9 is a view illustrating exemplary HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 9 show HTTP messages of respective requests illustrated in Fig. 7, and (b), (e), and (f) of Fig. 9 show HTTP messages of respective responses illustrated in Fig. 7.

##### Fig. 10

Fig. 10 is a view illustrating exemplary HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 10 show HTTP messages of respective requests illustrated in Fig. 7, and (b), (e), and (f) of Fig. 10 show HTTP messages of respective responses illustrated in Fig. 7.

##### Fig. 11

Fig. 11 is a view illustrating Embodiment 2 of the present invention, and illustrates a schematic configuration of a content distributing system in accordance with Embodiment 2 and main configurations of devices constituting the content distributing system.

##### Fig. 12

Fig. 12 is a view showing an exemplary transmission log stored in a transmission log storage section included in the content distributing system.

##### Fig. 13

Fig. 13 is a view illustrating exemplary content-storage-location information stored in a content-storage-location information storage section included in the content distributing system.

##### Fig. 14

Fig. 14 is a flowchart showing an exemplary process which is carried out by a server constituting the content distributing system.

##### Fig. 15

Fig. 15 is a flowchart showing an exemplary process which is carried out by a client which serves as a content playing device and constitutes the content distributing system.

Fig. 16

Fig. 16 is a flowchart showing an exemplary process which is carried out by a client which serves as a relaying device and constitutes the content distributing system.

Fig. 17

Fig. 17 is a view illustrating an exemplary operation sequence of a content distributing system in Example of Embodiment 2.

Fig. 18

Fig. 18 is a view illustrating example HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 18 show HTTP messages of respective requests illustrated in Fig. 17, and (b), (e), and (f) of Fig. 18 show HTTP messages of respective responses illustrated in Fig. 17.

Fig. 19

Fig. 19 is a view illustrating example HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 19 show HTTP messages of respective requests illustrated in Fig. 17, and (b), (e), and (f) of Fig. 19 show HTTP messages of respective responses illustrated in Fig. 17.

Fig. 20

Fig. 20 is a view illustrating Embodiment 3 of the present invention, and illustrates a schematic configuration of a content distributing system in accordance with Embodiment 3 and main configurations of devices constituting the content distributing system.

Fig. 21

Fig. 21 is a format of content dealt by the content distributing system.

Fig. 22

Fig. 22 is a flowchart showing an exemplary process which is carried out by a server constituting the content distributing system.

Fig. 23

Fig. 23 is a flowchart showing an exemplary process which is carried out by a client which serves as a content playing device and constitutes the content distributing system.

Fig. 24

Fig. 24 is a view illustrating an exemplary operation sequence of a content distributing system in Example of Embodiment 3.

Fig. 25

Fig. 25 is a view showing an exemplary transmission log stored in a transmission log storage section included in the content distributing system.

Fig. 26

Fig. 26 is a view illustrating exemplary content-storage-location information stored in a content-storage-location information storage section included in the content distributing system.

Fig. 27

Fig. 27 is a view illustrating exemplary HTTP messages which are transmitted/received as a request or a response. (a), (c), and (d) of Fig. 27 show HTTP messages of respective requests illustrated in Fig. 24, and (b), (e), and (f) of Fig. 27 show HTTP messages of respective responses illustrated in Fig. 24.

Fig. 28

Fig. 28 is a view illustrating exemplary HTTP messages which are transmitted/received as a request or a response. (a) and (b) of Fig. 28 show HTTP messages of respective requests illustrated in Fig. 24, and (c) and (d) of Fig. 28 show HTTP messages of respective responses illustrated in Fig. 24.

Fig. 29

Fig. 29 is a view illustrating Embodiment 4 of the present invention, and illustrates a schematic configuration of a content distributing system in accordance with Embodiment 4 and main configurations of devices constituting the content distributing system.

Fig. 30

Fig. 30 is a view illustrating exemplary content-storage-location information stored in a content-storage-location information storage section included in the content distributing system.

Fig. 31

Fig. 31 is a flowchart showing an exemplary process which is carried out by a server constituting the content distributing system.

Fig. 32

Fig. 32 is a flowchart showing an exemplary process which is carried out by a client which serves as a content playing device and constitutes the content distributing system.

Fig. 33

Fig. 33 is a view illustrating an exemplary operation sequence of a content distributing system in Example of Embodiment 4.

Fig. 34

Fig. 34 is a view illustrating an exemplary HTTP message which is transmitted and received as a response.

Fig. 35

Fig. 35 is a view illustrating exemplary MPD data which is meta data of content dealt by a content distributing system in accordance with Embodiment 5 of the present invention.

Fig. 36

Fig. 36 is a view illustrating exemplary MPD data which is dealt by the content distributing system.

Fig. 37

Fig. 37 is a view illustrating an exemplary external resource which is dealt by the content distributing system.

Fig. 38

Fig. 38 is a view illustrating exemplary MPD data and an example external resource which are dealt by the content distributing system.

Fig. 39

Fig. 39 is a flowchart showing an exemplary process which is carried out by a server constituting the content distributing system.

Fig. 40

Fig. 40 is a flowchart showing an exemplary process which is carried out by a client which serves as a content playing device and constitutes the content distributing system.

Fig. 41

Fig. 41 is a view illustrating an exemplary operation sequence of a content distributing system in Example of Embodiment 5.

#### Description of Embodiments

<Embodiment 1>

**[0017]** The description will discuss Embodiment 1 of the present invention with reference to Fig. 1 to Fig. 10. First, an outline of a content distributing system 1 of Embodiment 1 will be described with reference to Fig. 1.

[Outline of content distributing system 1]

**[0018]** Fig. 1 illustrates a schematic configuration of the content distributing system 1 in accordance with Embodiment 1 and main configurations of devices constituting the content distributing system 1. As illustrated in Fig. 1, the content distributing system 1 includes a server (content distributing device) 2, proxies (relaying devices, content acquiring devices) 3a and 3b, and clients (content playing devices, content acquiring devices) 4a and 4b. The content distributing system 1 also includes: a content storage section 5 connected to the server 2; and cache storage sections 6a and 6b connected to the respective proxies 3a and 3b.

**[0019]** As illustrated in Fig. 1, the server 2, the proxies 3a and 3b, and the client 4a and 4b are connected to one another via a network 7. Note that the network 7 is not particularly limited, provided that the aforementioned devices can be communicated with one another. Accordingly, the network 7 may be a wired communication network or a wireless communication network.

**[0020]** Note that, in the following description, the proxies 3a and 3b will be called generally as a proxy 3; the cache storage sections 6a and 6b, a cache storage section 6; and the client 4a and 4b, a client 4.

**[0021]** Further, the present invention is not limited to the example illustrated in Fig. 1, in which the content distributing system 1 includes two proxies 3 and two clients 4. The content distributing system 1 may include one proxy or three or more proxies, and may include three or more clients 4. That is, the content distributing system 1 only needs to include at least one proxy 3 and a plurality of clients 4.

**[0022]** In Embodiment 1, a content distributed by the server 2 is assumed to be a video content for a VOD

service, and a format of the content is assumed to be a MP4 file format. Further, a transfer protocol on the network 7 in the content distributing system 1 is assumed to use the HTTP, which is widely used as a general file transferring protocol. Note that, in Embodiment 1, the content distributed by the server 2 and the transfer protocol on the network 7 in the content distributing system 1 are not limited to this configuration.

10 [Server 2]

**[0023]** The server 2 is a content distributing device which receives a request message (request) to request to transmit a content from the proxy 3 and the client 4 (content acquiring device) and transmits a response message (response) in response to the request message thus received. As described above, the server 2 is connected to the content storage section 5 which stores contents such as moving images, and manages the contents stored in the content storage section 5. Note that the content storage section 5 may be provided inside the server 2.

**[0024]** Note that the server 2 may simultaneously distribute the same content to an unspecified number of devices, may distribute a content to a single device, or may simultaneously distribute the same content to a predetermined number of devices.

**[0025]** As illustrated in Fig. 1, the server 2 includes: a server controlling section 14 for collectively controlling operations of the server 2; a server communication section 11 which is provided so that the server 2 can communicate to an external device; a transmission log storage section 12 in which a response transmitted by the server 2 is stored; and a content-storage-location information storage section 13 for storing content-storage-location information (details will be described below). The server controlling section 14 includes: a response executing section (determining means, content transmitting means, content-acquiring-location instructing means, transmission record creating means) 15; a relaying device specifying section (content-acquiring-location instructing means, distance calculating means) 16; and a content-storage-location information generating section (content-storage-location information generating means) 17.

**[0026]** The response executing section 15 receives, via the server communication section 11, a request message to request to transmit a content, and then the response executing section 15 transmits, to a device (proxy 3 or client 4) which is the sender of the request message, a response message in response to the request message thus received.

**[0027]** Specifically, the response executing section 15 is ready to receive the request message to request to transmit the content, and, upon receiving, via the server communication section 11, the request message to request to transmit the content, the response executing section 15 checks a header of the request message in

order to determine whether or not the request message thus received is transmitted via the proxy 3. For example, it may be so configured that in a case of receiving a request message which includes a "Via" header, the response executing section 15 determines that the request message thus received has been transmitted by the proxy 3 in response to a request message of the client 4, whereas, in a case of receiving a request message which does not include the "Via" header, the response executing section 15 may determine that the request message thus received has been transmitted directly from the client 4.

**[0028]** Upon receipt of the request message to request the content directly from the client 4, the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify address information (e.g., URI of proxy 3 etc.) of the proxy 3 which (possibly) possesses the content that the client 4 requests. Then, the response executing section 15 receives, from the relaying device specifying section 16, the address information of the proxy 3 which possesses the content that the client 4 requests. After that, in response to the request message from the client 4, the response executing section 15 transmits, to the client 4, an instruction (i.e., a response message) to acquire such a requested content based on the address information specified by the relaying device specifying section 16. For example, by giving, to "Location field", with use of a "305 USE Proxy" serving as the response message, the address information specified by the relaying device specifying section 16, the response executing section 15 transmits, to the client 4, an instruction to transmit, to the proxy 3 which (possibly) possesses the content that the client 4 requests, the request message of the content.

**[0029]** When receiving, from the proxy 3, the request message to request the content, the response executing section 15 determines whether or not the request message thus received is a conditional request such as "If-Modified-Since". In a case where the request message thus received is not a conditional request, the response executing section 15 reads out a requested content from the content storage section 5 and transmits such a read-out content to the proxy 3 in response to the request message of the proxy 3.

**[0030]** Meanwhile, in a case where the request message thus received is a conditional request, the response executing section 15 then determines whether or not data held by the proxy 3 is latest. To put it another way, the response executing section 15 determines whether or not the data of the content held by the proxy 3 and data of the same content stored in the content storage section 5 are identical to each other.

**[0031]** In a case where the data of the content held by the proxy 3 is not latest, the response executing section 15 reads out a requested content from the content storage section 5, and, in response to the request message transmitted from the proxy 3, transmits the content thus read out to the proxy 3. Meanwhile, in a case where the

data of the content held by the proxy 3 is latest, the response executing section 15 transmits, to the proxy 3, in response to the request message transmitted from the proxy 3, a response message that the data of the content held by the proxy 3 is latest. The response executing section 15 transmits, to the proxy 3, for example, "304 NOT modified" serving as the response message that the data of the content held by the proxy 3 is latest.

**[0032]** Further, the response executing section 15 creates a transmission log (response transmission record) based on the response message thus sent, and causes the transmission log storage section 12 to store the transmission log thus created. The transmission log created by the response executing section 15 will be described later in detail.

**[0033]** As for the content to be transmitted from the response executing section 15 to the proxy 3, for example, "Cache-Control: must-revalidate" or "Cache-Control: proxy-revalidate" may be added to a header of the content. Accordingly, when the proxy 3 which receives the content uses, i.e., transmits the content held by the cache storage section 6 in response to another request, it is possible to confirm the server 2 before the transmission of the content from the proxy 3, as to whether or not the content is the latest version.

**[0034]** In response to an instruction of the response executing section 15, the relaying device specifying section 16 specifies which proxy 3 (possibly) possesses the content that the client 4 requests.

**[0035]** Specifically, the relaying device specifying section 16 reads out content-storage-location information from the content-storage-location information storage section 13 in response to the instruction of the response executing section 15. With reference to the content-storage-location information thus read out, the relaying device specifying section 16 specifies, as address information of a device from which the content is acquired, the address information (e.g., URI) of the proxy 3 associated with the content that the client 4 requests. The relaying device specifying section 16 transmits the address information thus specified to the response executing section 15.

**[0036]** In a case where the content-storage-location information for the content that the client 4 requests is not stored in the content-storage-location information storage section 13, the relaying device specifying section 16 may specify, as address information of a device from which the content is acquired, predetermined (default) address information of the content that the client 4 requests. Alternatively, in a case where the content-storage-location information storage section 13 does not store the content-storage-location information for the content that the client 4 requests, the relaying device specifying section 16 may select, at random, from proxies 3 connected to the server 2 via the network 7, address information of a device from which the content is acquired.

**[0037]** In a case where the content-storage-location in-

formation storage section 13 contains a plurality of pieces of content-storage-location information containing a plurality of pieces of address information of proxies 3 associated with the content requested by the client 4, the relaying device specifying section 16 may specify a plurality of pieces of address information as address information of a device from which the content is acquired, and may make a content-storage-location list containing the plurality of pieces of address information thus specified. Alternatively, in a case where the relaying device specifying section 16 specifies the plurality of pieces of address information as address information of a device from which the content is acquired, the relaying device specifying section 16 may make a content-storage-location list by adding priorities to the plurality of pieces of address information.

**[0038]** In a case where the content-storage-location information storage section 13 contains the plurality of pieces of content-storage-location information containing the respective plurality of pieces of address information of the proxies 3 associated with the content that the client 4 requests, the relaying device specifying section 16 may specify address information of a proxy 3, which address information is contained in the latest content-storage-location information (which means that date and time contained in the content-storage-location information are the latest), as the address information of the device from which the content is acquired. In a case where a plurality of pieces of address information are specified as the address information of the device from which the content is acquired, the relaying device specifying section 16 may make a content-storage-location list by adding priorities to the pieces of address information in such a manner that, for example, later date and time contained in content-storage-location information are given a higher priority.

**[0039]** Further, in a case where the content-storage-location information storage section 13 contains the plurality of pieces of content-storage-location information containing the respective plurality of pieces of address information of proxies 3 associated with the content that the client 4 requests, the relaying device specifying section 16 may specify an address of a proxy 3 as an address of a device from which the content is acquired, which address of the proxy 3 has a shortest distance between the proxy 3 and the client 4 on the basis of a physical or network-structural distance and is contained in the content-storage-location information. Also in a case where the plurality of pieces of address information are specified as the address information of the device from which the content is acquired, the relaying device specifying section 16 may make a content-storage-location list on the basis of, for example, the distance between the proxy 3 and the client 4 on the physical or network structure. That is, the relaying device specifying section 16 may make the content-storage-location list by adding priorities to the plurality of pieces of address information in such a manner that address information with a shorter distance

gets a higher priority. Specifically, the relaying device specifying section 16 may make a content-storage-location list in such a manner that (A) a plurality of physical or network-structural distances between proxies 3 indicated by the plurality of addresses and clients 4 transmitted the request are calculated on the basis of the plurality of addresses contained in respective plurality of pieces of content-storage-location information, (B) the plurality of addresses, which are contained in the plurality of pieces of content-storage-location information, respectively, are arranged so that an address with a shorter distance gets a higher priority, and (C) priorities are assigned to the plurality of addresses.

**[0040]** In a case where, for example, the server 2 already has (A) URIs of the clients 4 transmitted the request message and (B) URIs of the proxies 3 connected to the server 2 via the network 7, the relaying device specifying section 16 may make a content-storage-location list in such a manner that (I) domain names of the clients 4 transmitted the request message are found out on the basis of URIs of the clients 4, (II) proxies 3 are arranged so that, in each level of the domain names thus found out, the shorter physical or network-structural distance between a client 4 and a proxy 3 gets a higher priority, and (III) priorities are assigned to the plurality of pieces of address information.

**[0041]** Specifically, in a case where the URI of the client 4 is, for example, "http://client.co.jp", the relaying device specifying section 16 may make a content-storage-location list in view of a second level domain name. Specifically, the relaying device specifying section 16 may make the content-storage-location list by (i) selecting proxies 3 whose second level domain name is "co.jp" and (ii) arranging the proxies 3 thus selected in such a manner that the shorter physical or network-structural distance between the proxy 3 and the client 4 gets a higher priority. In a case where no proxy 3 has "co.jp" as the second level domain name, the relaying device specifying section 16 may make a content-storage-location list by (A) selecting proxies 3 whose first level domain name is "jp" and (B) arranging the proxies 3 in such a manner that the shorter physical or network-structural distance between the proxy 3 and the client 4 gets a higher priority.

**[0042]** In a case where the URI of the client 4 is not an organizational type JP domain name such as "co.jp", but is a geographical type JP domain name such as "kai-sha.chiyoda.tokyo.jp" or "pref.fukuoka.jp" in which a prefecture label or a city (city, ward, town, village) label is contained, the relaying device specifying section 16 may make a content-storage-location list by selecting (I) labels (prefecture labels or city labels) to which proxies 3 are attached are same as those to which the clients 4 are attached or (II) labels to which proxies 3 are attached are resemble to those to which the clients 4 are attached.

**[0043]** Similarly, in a case where server 2 already contains information on an IP address of the client 4 which has transmitted the request message, an IP address of the proxy 3 connected to the server 2 via the network 7,

and the like, the relaying device specifying section 16 may select, with reference to a network address section on a high-order-bit side of the IP address of the client 4 which has transmitted the request message, a proxy 3 having a short physical or network-structural distance between the proxy 3 and the client 4.

**[0044]** In addition, a connecting path between the server 2 and the client 4 and a network status are checked in advance with use of, for example, a general method such as a routing technology, a network monitoring, or traffic analysis, and, on the basis of the connecting path and the network status specified by the relaying device specifying section 16, a more appropriate proxy may be selected. For example, before the relaying device specifying section 16 transmits, to client 4, an instruction to designate a proxy 3 for acquiring the content, the connecting path between the server 2 and the client 4 is checked with use of a "traceroute" command or a "tracert" command, and, on the basis of a result of such check, the relaying device specifying section 16 may select a proxy 3 which is closer to the client 4.

**[0045]** Further, the relaying device specifying section 16 may specify, in view of load statuses of the proxies 3, a proxy 3 which (possibly) possesses the content that the client 4 requests and has a low load. More specifically, the relaying device specifying section 16 may specify, with reference to the transmission log stored in the transmission log storage section 12, address information of the device from which the content is acquired.

**[0046]** For example, the relaying device specifying section 16 may carry out the following processes: the relaying device specifying section 16 determines, with reference to the transmission log stored in the transmission log storage section 12, whether or not each proxy 3 has transmitted or received the content within a predetermined time period; and the relaying device specifying section 16 specifies, as the address information of the device from which the content is acquired, address information of a proxy 3 which (i) has transmitted or received no content within the predetermined time period and (ii), in the content-storage-location information thus read out, is associated with the content that the client 4 requests. Note that the predetermined time period may be set in accordance with the content associated with the proxy 3.

**[0047]** In a case where the content-storage-location information storage section 13 contains the plurality of pieces of content-storage-location information containing the respective plurality of pieces of address information of the proxies 3 associated with the content that the client 4 requests, the relaying device specifying section 16 may specify, as the address information of the device from which the content is acquired, address information of a proxy 3 which (A) has transmitted or received no content within a predetermined time period and (B) is contained in the latest content-storage-location information.

**[0048]** Further, the relaying device specifying section 16 may specify, in view of the number of accesses to the

server 2 from the proxies 3, a proxy 3 which (possibly) possesses the content that the client 4 requests. For example, it can be considered that, if a proxy 3 has a large number of accesses to the server 2, the proxy 3 caches a large number of contents. Accordingly, the relaying device specifying section 16 may specify a proxy 3 whose number of accesses to the server 2 is a predetermined lower limit or more. Note that the relaying device specifying section 16 may specify the number of accesses to the server 2 with reference to the transmission log stored in the transmission log storage section 12. Specifically, the relaying device specifying section 16 may specify, as the address information of the device from which the content is acquired, an address of a proxy 3 whose number of accesses to the server 2 is a predetermined lower limit or more, which address of the proxy 3 is selected from the proxies 3 associated with the contents, which are requested by the clients 4, of the content-storage-location information.

**[0049]** Further, in a case where the plurality of pieces of address information are associated with the content that the client 4 requests, the relaying device specifying section 16 may specify, as the address information of the device from which the content is acquired, an address information of a proxy 3 having a largest number of accesses to the server 2.

**[0050]** Furthermore, in order not to concentrate accesses to a specific proxy 3, the relaying device specifying section 16 may specify a proxy 3 whose number of accesses to the server 2 is a predetermined upper limit or less.

**[0051]** The content-storage-location information generating section 17 generates, on the basis of the transmission log stored in the transmission log storage section 12, content-storage-location information for specifying address information of a proxy 3 which (possibly) possesses content, and causes the content-storage-location information storage section 13 to store the content-storage-location information.

**[0052]** Specifically, the content-storage-location information generating section 17 generates content-storage-location information by associating, with reference to a transmission log that the server 2 has transmitted the content to a proxy 3, (A) the content which has been transmitted by the server 2, (B) address information of the proxy 3 which is a destination of the content, and (C) date and time when the server 2 has transmitted the content. Alternatively, the content-storage-location information generating section 17 generates content-storage-location information by associating (A) the content which is confirmed, by the server 2, that the content is the latest version with reference to a transmission log indicating that the server 2 has transmitted a response message that the content held by the proxy 3 is the latest version, (B) address information of the proxy 3 which possesses the content, and (C) date and time when the server 2 has transmitted the response message.

**[0053]** Note that the content-storage-location informa-

tion generating section 17 may generate content-storage-location information every time when a predetermined transmission log is added to the transmission log storage section 12, or may generate content-storage-location information by reading out the transmission log per predetermined time period.

**[0054]** The transmission log storage section 12 stores a transmission log in which (A) transmission date and time of a response message that the server 2 has transmitted to the proxy 3 or to the client 4, (B) address information of the device which receives the response message, (C) contents of the response message, and (D) a content requested by a request message in response to the response message are associated with one another. Examples of the transmission log stored in the transmission log storage section 12 encompass data illustrated in Fig. 2. Fig. 2 is a view showing an exemplary transmission log stored in the transmission log storage section 12.

**[0055]** As shown in Fig. 2, the transmission log associates (A) "Date" which is date and time when a response message is transmitted, (B) "destination address" which is address information of a device which receives the response message, (C) "transmitted contents" which indicates the contents of the response message, and (D) "content ID" which indicates the content requested by a request message in response to the response message.

**[0056]** "200 OK", "305 USE Proxy", or "304 Not Modified", which is the contents of the response message, is stored as the "transmitted contents". In a case where the "transmitted contents" is "305 USE Proxy", the "305 USE Proxy" contains, in parentheses, information indicative of a proxy 3 which is designated by the relaying device specifying section 16, i.e., "305 Use Proxy (proxy 1)" as shown in Fig. 2.

**[0057]** "200 OK" is contained in a response message transmitted when the content is transmitted in response to the request for the content. Thus, a transmission log of the response message "200 OK" is referred to as a content transmission log. "305 USE Proxy" is contained in a response message to provide a device with the content from a designated proxy. Thus, a transmission log of the response message "305 USE Proxy" is referred to as an acquiring instruction log. "304 Not Modified" is contained in a response message which notifies a device that the content is the latest version, the transmission log of the response message "304 Not Modified" is referred to as a version notification log.

**[0058]** The response executing section 15 creates a transmission log based on a transmitted response message, so that the content-storage-location information generating section 17 can make, on the basis of the transmission logs, content-storage-location information for specifying address information of a proxy 3 which (possibly) possesses a content (as described above). The relaying device specifying section 16 can refer to the transmission log in order to specify a proxy 3 which has transmitted or received no content within a predeter-

mined time period.

**[0059]** With reference to a transmission log stored in the transmission log storage section 12, for example, the relaying device specifying section 16 specifies, as a proxy 3 which has transmitted or received a content within a predetermined time period, a proxy 3 which is associated with "200 OK" or "304 Not Modified" as the transmitted contents.

**[0060]** The content-storage-location information storage section 13 stores content-storage-location information in which (A) a content, (B) address information of a proxy 3 which (possibly) possesses the content, and (C) date and time when the proxy 3 acquires the content are associated with one another. Examples of the content-storage-location information stored in the content-storage-location information storage section 13 may encompass data shown in Fig. 3. Fig. 3 is a view showing exemplary content-storage-location information stored in the content-storage-location information storage section 13. To put it another way, the content-storage-location information is information in which the content and the address information indicative of a location where the content is stored are associated with each other.

**[0061]** As shown in Fig. 3, the content-storage-location information is information in which (A) "Date" which is date and time when the proxy 3 acquires a content, (B) "content ID" (content identification information) which identifies the content, and (C) "storage-location address" which is address information of the proxy 3 which possesses the content are associated with one another.

**[0062]** The content-storage-location information generating section 17 generates content-storage-location information based on a transmission log. Accordingly, as described above, the relaying device specifying section 16 can specify, with reference to the content-storage-location information, address information of a proxy 3 which (possibly) possesses the content.

**[0063]** Note that, like the content storage section 5, the transmission log storage section 12 and the content-storage-location information storage section 13 may be provided outside the server 2 and connected to the server 2, instead of being provided in the server 2.

[Proxy 3]

**[0064]** The proxy 3 is a device for transmitting a requested content and also for requesting and acquiring a content. As described above, the proxy 3 is connected to the cache storage section 6 which stores, for example, a content received from the server 2. In a case where the cache storage section 6 includes a requested content, the proxy 3 reads out the requested content from the cache storage section 6 and then transmits the requested content to the client 4. Meanwhile, in a case where the cache storage section 6 does not include the requested content, the proxy 3 requests the content to the server 2. Note that the cache storage section 6 may be provided in the proxy 3.



**[0065]** As illustrated in Fig. 1, the proxy 3 includes: a proxy controlling section 22 for collectively controlling operations of the proxy 3; and a proxy communication section 21 with which the proxy 3 communicates to an external device. The proxy controlling section 22 includes a response/request execution section 23.

**[0066]** The response/request execution section 23 is provided for transmitting, to the client 4, a content designated by a request message which has been received from the client 4 to request the content.

**[0067]** Specifically, the response/request execution section 23 is ready to receive, from the client 4, a request message to request to transmit a content, and, upon receipt of the request message from the client 4 via the proxy communication section 21, the response/request execution section 23 confirms whether or not a requested content is stored in the cache storage section 6.

**[0068]** In a case where the requested content is not stored in the cache storage section 6, the response/request execution section 23 confirms a "Host" field of the request message transmitted from the client 4, and transmits, to the server 2 indicated by a URI written in the "Host" field, the request message of the content that the client 4 requests. Then, the response/request execution section 23 acquires the content from the server 2, and causes the cache storage section 6 to store the content thus acquired. After that, the response/request execution section 23 transmits the requested content to the client 4.

**[0069]** Further, in a case where the requested content is stored in the cache storage section 6, the response/request execution section 23 acts such that, in order to confirm whether or not data of the content stored in the cache storage section 6 is the latest, the response/request execution section 23 makes a request message provided with a conditional request, such as "If-Modified-Since" in which time information indicative of time when the proxy 3 acquires the content is written, and transmits, to the server 2, the request message provided with the conditional request thus made.

**[0070]** In a case where the response/request execution section 23 receives, from the server 2, a response message "304 NOT Modified" in response to the request message provided with the conditional request, the response/request execution section 23 determines that the content stored in the cache storage section 6 is the latest data. Then the response/request execution section 23 reads out the content from the cache storage section 6, and transmits, to the client 4, the content thus read out.

**[0071]** Meanwhile, in a case where the content stored in the cache storage section 6 is not the latest data, the server 2 transmits the latest content in response to the request message provided with the conditional request, and the response/request execution section 23 acquires the content transmitted from the server 2. The response/request execution section 23 causes the cache storage section 6 to store the content thus acquired, and then transmits, to the client 4, the content thus required.

**[0072]** Note that the response/request execution sec-

tion 23 may be configured to transmit the content to the client 4 while the content contains, in its header, "Cache-Control: must-revalidate" or "Cache-Control" which has been added by the server 2. Accordingly, in a case where the content, which has been acquired by the client 4 serving as a destination of the content, is transmitted in response to another request, it is possible to cause the client 4 to always transmit, to the server 2, an inquiry as to whether or not the content is the latest version, before the client 4 transmits the content.

**[0073]** That is, in Embodiment 1, the proxy 3 is a device for transmitting a request of a content and acquiring the content, and is also a relaying device for possessing such a requested content and relaying the requested content to another device (e.g., client 4) to which the requested content is transmitted from the relaying device.

[Client 4]

**[0074]** The client 4 is a device for requesting and acquiring a content, and playing the content thus acquired. Although not illustrated in Fig. 1, the client 4 includes an input section for accepting a user's operation, and requests a content on the basis of an input operation accepted by the input section.

**[0075]** As illustrated in Fig. 1, the client 4 includes: a client control section 32 for collectively controlling operations of the client 4; and a client communication section 31 with which the client 4 communicates to an external device. The client control section 32 includes: a request executing section 33 and a content playing section 34.

**[0076]** The request executing section 33 generates a request message to request to transmit a content, transmits the request message to the server 2 via the client communication section 31, and receives a response message in response to the request message. In the present invention, as described above, the request executing section 33 receives, from the server 2, the response message which is (A) information for specifying a proxy and (B) an instruction for transmitting, to the proxy, a request message to request the content. Upon receipt of the response message from the server 2, the request executing section 33 transmits, to such a designated proxy 3, the request message to request to transmit the content in response to the instruction from the server 2. Then, the request executing section 33 acquires the content from the proxy 3 designated by the server 2.

**[0077]** The content playing section 34 plays the content acquired by the request executing section 33. In a case where, for example, the acquired content is a moving image, the content playing section 34 decodes the acquired content. Then, the content playing section 34 causes an external display device (not shown) to output a moving image and audio thud obtained by decoding.

**[0078]** That is, in Embodiment 1, the client 4 is a device for transmitting a request for a content and acquiring the content, and also serves as a content playing device for acquiring and playing the requested content.

[Processes carried out by each device]

**[0079]** The following description will discuss, with reference to Figs. 4 to 6, processes carried out by the server 2, the proxy 3, and the client 4. First, a process carried out by the server 2 will be described below with reference to Fig. 4.

[Process carried out by server 2]

**[0080]** Fig. 4 is a flowchart showing an exemplary process which is carried out by the server 2. The response executing section 15 is ready to receive a request message to request to transmit a content. When the response executing section 15 receives, via the server communication section 11, the request message to request to transmit the content (S401), the response executing section 15 checks a header of the request message thus received, so as to determine whether or not the request message thus received is transmitted via a proxy 3 (S402).

**[0081]** In a case where the received request message is one transmitted from the client 4 (NO in S402), the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify a proxy 3 from which the client 4 acquires the content. In response to the instruction from the response executing section 15, the relaying device specifying section 16 reads out content-storage-location information from the content-storage-location information storage section 13 (S403). With reference to the content-storage-location information thus read out, the relaying device specifying section 16 specifies, as address information of a device from which the content should be acquired, a URI of the proxy 3 associated with the content that the client 4 requests (S404). The relaying device specifying section 16 transmits such specified address information to the response executing section 15. The response executing section 15 transmits, to the client 4, from the proxy 3 indicated by an address contained in the address information which has been specified by the relaying device specifying section 16, a response message to instruct the client 4 to acquire a requested content (S405). The response executing section 15 creates a transmission log based on the response message which has been transmitted to the client 4, and adds the transmission log thus created to the transmission log storage section 12 (S410).

**[0082]** Meanwhile, in a case where the request message thus received is one transmitted by the proxy 3 (YES in S402), the response executing section 15 then determines whether or not the received request message is a conditional request such as "If-Modified-Since" (S406). In a case where the received request message is not a conditional request (NO in S406), the response executing section 15 reads out, from the content storage section 5, the content requested by the proxy 3, and, in response to the request message from the proxy 3, transmits the

content thus read out to the proxy 3 (S407). After that, the response executing section 15 creates a transmission log based on the response message which has been transmitted to the proxy 3, and adds the transmission log thus created to the transmission log storage section 12 (S410).

**[0083]** In a case where the received request message is provided with a conditional request (YES in S406), the response executing section 15 determines whether or not data of the content held by the proxy 3 is the latest (S408). In a case where the data of the content kept by the proxy 3 is not the latest (NO in S408), the response executing section 15 reads out, from the content storage section 5, the content requested by the proxy 3, and, in response to the request message from the proxy 3, transmits to the proxy 3 the content thus read out (S407). Meanwhile, in a case where the data of the content held by the proxy 3 is the latest (YES in S408), the response executing section 15 transmits, to the proxy 3, a response message that the data of the content held by the proxy 3 is the latest, in response to the request message from the proxy 3 (S409). Then, the response executing section 15 adds, to the transmission log storage section 12, the response message which has been transmitted to the proxy 3 (S410).

**[0084]** Note that, in a case where the response executing section 15 transmits the content to the proxy 3 in S407, "Cache-Control: must-revalidate" or "Cache-Control: proxy-revalidate", for example, is added to the header of the content. Accordingly, in a case where the proxy 3 serving as a destination of a content transmits the content held by the cache storage section 6 in response to another request, it is possible to always cause the proxy 3 to transmit, to the server, an inquiry as to whether or not the content is the latest, before the proxy 3 transmits the content.

[Process carried out by proxy 3]

**[0085]** The following description will discuss a process carried out by the proxy 3. Fig. 5 is a flowchart showing an exemplary process which is carried out by the proxy 3.

**[0086]** As shown in Fig. 5, the response/request execution section 23 is ready to receive a request message to request to transmit a content from the client 4. Upon receipt of the request message from the client 4 via the proxy communication section 21 (S421), the response/request execution section 23 confirms whether or not such a requested content is stored in the cache storage section 6 (S422).

**[0087]** In a case where the requested content is not stored in the cache storage section 6 (NO in S422), the response/request execution section 23 refers to a "Host" field of the request message which has been transmitted from the client 4, and transmits, to the server 2 indicated by a URI written in the "Host" field, the request message to request the content transmitted by the client 4 (S423). Then, the response/request execution section 23 ac-

quires the content from the server 2 (S424), and causes the cache storage section 6 to store the content thus acquired (S425). After that, the response/request execution section 23 transmits, to the client 4, the content thus requested (S426).

**[0088]** In a case where the requested content is stored in the cache storage section 6 (YES in S422), the response/request execution section 23 creates a request message provided with a conditional request, such as "If-Modified-Since" in which time information indicative of time when the content has been acquired is written, and transmits, to the server 2, the request message provided with the conditional request thus made (S427).

**[0089]** In a case where the response/request execution section 23 receives a response message "304 NOT Modified" from the server 2 in response to the request message provided with the conditional request (YES in S428), the response/request execution section 23 reads out the content thus stored from the cache storage section 6, and transmits to the client 4 the content thus read out (S426).

**[0090]** Meanwhile, in a case where the content stored in the cache storage section 6 is not the latest data, the server 2 transmits the latest content in response to the request message provided with the conditional request, and the response/request execution section 23 acquires the latest content transmitted from the server 2 (S424). The response/request execution section 23 causes the cache storage section 6 to store the content thus acquired (S425), and then transmits the requested content to the client 4 (S426).

**[0091]** Note that, in S426, the response/request execution section 23 transmits the content to the client 4 while the content contains, in its header, "Cache-Control: must-revalidate" or "Cache-Control" which has been added by the server 2. Accordingly, in a case where the content, which has been acquired by the client 4 serving as a destination of a content, is transmitted in response to another request, it is possible to keep such a state that the client 4 should always transmits, to the server 2, an inquiry as to whether or not the content is the latest version, before the client 4 transmits the content.

[Process carried out by client 4]

**[0092]** The following description will discuss, with reference to Fig. 6, a process carried out by the client 4. Fig. 6 is a flowchart showing an exemplary process, which is carried out by the client 4.

**[0093]** As shown in Fig. 6, the request executing section 33 transmits, to the server 2, a request message to request to transmit a content (S441). In response to the request message, the request executing section 33 receives a response message to instruct the request executing section 33 to acquire a content from a designated proxy 3 (S442). In response to such an instruction from the server 2, the request executing section 33 transmits the request message to the proxy 3 designated by the

server 2 (S443). Then, in response to the request message, the request executing section 33 acquires the content from the proxy 3 which has received the request message (S444).

5

[Example 1]

**[0094]** Fig. 7 illustrates Example 1, and Embodiment 1 will be described in detail below. Example 1 shows an operation example of the content distributing system 1 that instructs the client 4 about from which proxy 3 the client 4 is to acquire the content. Fig. 7 is a view illustrating an example operation sequence of the content distributing system 1 in Example 1.

**[0095]** Note that Example 1 is based on the following presumption. A content 1 and a content 2 are stored in the content storage section 5, and both the cache storage sections 6a and 6b cache no content. Further, the server 2 is set so that the content 1 is acquired from the proxy 3a as its default and the content 2 is acquired from the proxy 3b as its default. Furthermore, when the process shown in Fig. 7 is started in Example 1, a transmission log and content-storage-location information are not stored in the transmission log storage section 12 and the content-storage-location information storage section 13, respectively. Still further, the response executing section 15 creates the transmission log of Fig. 2 every time when a response message is transmitted, and the content-storage-location information generating section 17 generates the content-storage-location information of Fig. 3 every time when a content transmission log or a version notification log (transmission log whose "transmitted contents" is "200 OK" or "304 Not Modified") is added to the transmission log storage section 12.

**[0096]** Further, one session is defined as a sequence starting from a time at which the client 4 transmits a request message and ending a time at which the client 4 receives a response message in response to the request message is regarded as one session.

**[0097]** As shown in Fig. 7, in a session 110, the client 4a transmits, to the server 2, a request message to request to transmit the content 1 (request 111). In the server 2 which has received the request 111, the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify a proxy 3 to acquire the content 1. In response to the instruction, the relaying device specifying section 16 confirms whether or not the content-storage-location information is stored in the content-storage-location information storage section 13 (process 112). Here, the content-storage-location information storage section 13 does not contain the content-storage-location information indicative of the proxy 3 to acquire the content 1, the relaying device specifying section 16 specifies the default proxy 3a as the proxy 3 to acquire the content 1. The response executing section 15 transmits, to the client 4a, the response message to instruct the client 4a to acquire a requested content from the proxy 3 indicated by address information specified

by the relaying device specifying section 16 (response 113). Then, the response executing section 15 creates a transmission log based on the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 114).

**[0098]** Then, in the session 120, the client 4a, which has received the response 113, transmits, to proxy 3a, the request message to request to transmit the content 1 (request 121). The proxy 3a, which received the request 121, confirms whether or not the content 1 is stored in the cache storage section 6a (process 122). Here, the content 1 is not stored in the cache storage section 6a, so that the proxy 3a transmits, to the server 2, the request message based on the request 121 (request 123). In the server 2 which has received the request 123 from the proxy 3a, the response executing section 15 transmits the content 1 to the proxy 3a (response 124). The proxy 3a received the response 124 stores the acquired content 1 in the cache storage section 6a and caches the content 1 (process 125). After that, the proxy 3a transmits, to the client 4a, the content 1 as a response to the request 121 (response 126). Note that, after transmitting the response 124, the response executing section 15 creates a transmission log based on the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 127). Further, the content-storage-location information generating section 17 generates content-storage-location information in which the content 1 and address information of the proxy 3a are associated with each other, and causes the content-storage-location information storage section 13 to store the content-storage-location information (process 128).

**[0099]** Next, in the session 130 and the session 140, the client 4b acquires the content 2. Operations of the client 4b, the proxy 3b, and the server 2 in the session 130 and the session 140 are similar to those of the client 4a, the proxy 3a, and the server 2 in the session 110 and the session 120, except that data acquired in the sessions 110 and 120 is different from that acquired in the sessions 130 and 140. Therefore description thereof will be omitted.

**[0100]** When the session 140 is completed, the content 1 is in a state of being cached in the cache storage section 6a of the proxy 3a and the content 2 is in a state of being cached in the cache storage section 6b of the proxy 3b. That is, the content-storage-location information storage section 13 stores (A) the content-storage-location information in which the content 1 and the proxy 3a are associated with each other and (B) content-storage-location information in which the content 2 and the proxy 3b are associated with each other.

**[0101]** In the session 150, the client 4a transmits, to the server 2, a request message to request to transmit the content 2 (request 151). In the server 2 received the request 151, the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to designate a proxy 3 to acquire the content 2. In response to the instruction, the relaying device spec-

ifying section 16 confirms whether or not the content-storage-location information is stored in the content-storage-location information storage section 13 (process 152). Here, the content-storage-location information storage section 13 stores the content-storage-location information in which the content 2 and the proxy 3b associated with each other, the relaying device specifying section 16 specifies the proxy 3b to acquire the content 2. The response executing section 15 transmits, to the client 4a, a response message to instruct the client 4a to acquire a requested content from the proxy 3 specified by the relaying device specifying section 16 (response 153). Then, the response executing section 15 creates a transmission log based on the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 154).

**[0102]** Next, in the session 160, in response to the instruction of the server 2, the client 4a received the response 153 transmits, to the proxy 3b, the request message to request to transmit the content 2 (request 161). The proxy 3b received the request 161 confirms whether or not the content 2 is stored in the cache storage section 6b (process 162). The content 2 is already stored in the cache storage section 6b, so that the proxy 3b transmits, to the server 2, the request message, based on the request 121, provided with a conditional request "If-Modified-Since", in order to confirm whether or not the content 2 stored in the cache storage section 6b is the latest data (request 163). Because the proxy 3b is used to transmit the request 163 and the request 163 is a request message contained in the conditional request, the response executing section 15 in the server 2 received the request 163 confirms whether or not the content 2 held by the proxy 3b is the latest data (process 164). The response executing section 15 determines that the content 2 held by the proxy 3b is the latest data, and transmits, to the proxy 3b, a response message "304 NOT Modified" (response 165). The proxy 3b received the response 165 reads out the content 2 stored in the cache storage section 6b, and, in response to the request 161, transmits the read out content 2 to the client 4a (response 166). Note that, after transmitting response 165, the response executing section 15 creates a transmission log based on the response message, and adds the transmission log to the transmission log storage section 12 (process 167). Further, the content-storage-location information generating section 17 generates the content-storage-location information in which the content 2 and address information of the proxy 3b are associated with each other, and causes the content-storage-location information storage section 13 to store the content-storage-location information (process 168).

**[0103]** As described above, in Embodiment 1, in a case where the client 4 requests the content from the server 2 and the server 2 has transmitted the same content to the proxy before, the server 2 determines that the proxy 3 caches the content, and transmits, to the client 4, an instruction to acquire the content from the proxy 3 which

has been determined to possess the requested content.

**[0104]** Embodiment 1 is assumed to be applied to a VOD service in which a moving image content is used with an MP4 file format. Accordingly, in terms of loads to the server 2, the proxy 3, and the network 7, "transmission and reception of content" applies the heaviest loads.

**[0105]** Accordingly, the number of request and response in the sessions 110 and 120 and the sessions 130 and 140 in Example 1 is larger in the present invention than in a conventional method. "Transmission and reception of the content" in the present invention, however, is almost similar to that in the conventional method, except that the content is transmitted and received via the proxy 3 in the present invention. However, because of the sessions 110 and 120 and the sessions 130 and 140, it is possible to carry out "transmission and reception of the content" in the session 160 between the proxy 3b and the client 4a. This particularly decreases (A) a processing load of the server 2 and (B) an amount of transmission information (network load) of a network between the server 2 and the proxy 3.

**[0106]** For the sake of easy explanation, the following is the simplest example, specifically, an example where two proxies 3, two clients 4, and two kinds of contents managed by the server 2 are used. In a case of an actual content distribution service such as VOD, however, various and many contents are transmitted and received in an extremely huge system. Therefore, the present invention is greatly effective in reducing a load of the server 2 and a load of the network between the server 2 and the proxy 3. To put it another way, in the conventional methods, when the number of clients 4 becomes larger, processing loads especially of the server 2 and to a network between the server 2 and the client 4 become greatly higher, depending on the number of the clients 4. However, by using the present invention, the server 2 can efficiently utilize a throughput of the proxy 3 and a cash function. As a result, the load of the server 2 and the load of the network between the server 2 and the proxy 3 can be shared by the proxy 3 or by the network between the proxy 3 and the client 4.

[HTTP Message in Embodiment 1]

**[0107]** The requests and responses involved in the operation sequence illustrated in Fig. 7 will be described in detail with reference to Figs. 8 through 10. Figs. 8 through 10 are views each illustrating example HTTP messages transmitted/responded as requests or responses. Fig. 8 shows example HTTP messages transmitted/responded in the sessions 110 and 120. Fig. 9 shows example HTTP messages transmitted/responded in the sessions 130 and 140. Fig. 10 shows example HTTP messages transmitted/responded in the sessions 150 and 160.

**[0108]** (a), (b), (c), (d), (e), and (f) of Fig. 8 show HTTP messages of the request 111, the response 113, the request 121, the request 123, the response 124, and the response 126 of Fig. 7, respectively.

**[0109]** (a), (b), (c), (d), (e), and (f) of Fig. 9 show HTTP messages of the request 131, the response 133, the request 141, the request 143, the response 144, and the response 146 of Fig. 7, respectively.

**[0110]** (a), (b), (c), (d), (e), and (f) of Fig. 10 show HTTP messages of the request 151, the response 153, the request 161, the request 163, the response 165, and the response 166 of Fig. 7, respectively.

[HTTP messages in sessions 110 and 120]

**[0111]** The HTTP messages in the sessions 110 and 120 will be described below with reference to Fig. 8. Note that (a) through (f) of Fig. 9 correspond to (a) through (f) of Fig. 8, respectively, and that Fig. 9 is different from Fig. 8 only in that a content 1, a proxy 3a, and a client 4a in Fig. 8 are changed to a content 2, a proxy 3b, and a client 4b in Fig. 9, respectively. Therefore, description of the HTTP messages in the sessions 130 and 140 with reference to Fig. 9 will be omitted. Note also that, of all the components of the HTTP messages, (i) components specific to the present invention will be primarily described below and (ii) description of well-known components of the HTTP messages will be appropriately omitted.

(Request 111 to request content)

**[0112]** As illustrated in (a) of Fig. 8, the HTTP message, which serves as the request 111 (i) transmitted from the client 4a to the server 2 and (ii) requesting the content 1, contains a request line and a header which notifies additional information.

**[0113]** The request line illustrated in (a) of Fig. 8 contains "GET" followed by additional information, which "GET" indicates a method for acquiring a content and which information specifies what content to be acquired. Specifically, the information is described in the form of "/content name". This means that the HTTP message illustrated in (a) of Fig. 8 serves as a request for transmission of the content 1 described by "content 1" in the request line.

**[0114]** Headers illustrated in (a) of Fig. 8 include a "Host" header for specifying a server to acquire the content, and the "Host" header shows an address, "example.com", indicative of the address of the server 2.

**[0115]** The headers illustrated in (a) of Fig. 8 also include an "Accept" header indicative of a data format that can be processed by the client 4a, and the "Accept" header shows, "video/mp4", indicative of video data in MP4 format. This allows the client 4a (the sender of the request) to inform the server 2 (the recipient of the request) that the client 4a is capable of receiving video data in MP4 format.

(Response 113 specifying device from which content is to be acquired)

**[0116]** As illustrated in (b) of Fig. 8, the HTTP message, which serves as the response 113 (i) transmitted from the server 2 to the client 4a and (ii) specifying a device from which the content is to be acquired, contains a response line and a header.

**[0117]** The response line illustrated in (b) of Fig. 8 contains information instructing to use a proxy 3 specified by a "Location" header. Specifically, the instruction is described in the form of "status number (space) message". This means that the HTTP message illustrated in (b) of Fig. 8 serves as a response for instructing the client 4a to request the content 1 from the proxy 3 designated by the "Location" header described below.

**[0118]** The header illustrated in (b) of Fig. 8 includes the "Location" header for specifying a proxy to be used, and the "Location" header contains address information, "http://example-proxy1.com", indicative of an address of the proxy 3 to be used. From this, the client 4a (the recipient of the response) obtains the address information of the proxy 3 (the device that the client 4a requests the content 1).

(Request 121 requesting content from specified proxy 3)

**[0119]** As illustrated in (c) of Fig. 8, the HTTP, which serves as the request 121 (i) transmitted from the client 4a to the proxy 3a and (ii) requesting the content 1, contains a request line and headers.

**[0120]** The request line illustrated in (c) of Fig. 8 contains "GET" followed by a URL, which "GET" indicates a method for acquiring the content and which URL is of the content to be requested. Specifically, the URL is described in the form of "http://name of a server storing a content/content name." This means that the HTTP message illustrated in (c) of Fig. 8 serves as a request for transmission of a content 1 stored in the server 2.

**[0121]** As in the case of the request 111, the headers illustrated in (c) of Fig. 8 include a "Host" header and an "Accept" header.

(Request 123 from proxy 3a to server 2 for content)

**[0122]** As illustrated in (d) of Fig. 8, the HTTP message, which serves as the request 123 (i) transmitted from the proxy 3 to the server 2 and (ii) requesting the content 1, contains a request line and headers.

**[0123]** The request line illustrated in (d) of Fig. 8 contains "GET" followed by a URL, which "GET" indicates a method for acquiring a content and which URL is of the content to be requested. Specifically, the URL is described in the form of "/" content name." This means that the HTTP message illustrated in (a) of Fig. 8 serves as a request for transmission of the content 1 described by "content 1" in the request line.

**[0124]** As in the case of the request 111, the headers

illustrated in (d) of Fig. 8 include a "Host" header and an "Accept" header. The headers also include a "Via" header indicative of a transmission path of the message, which "Via" header contains an address, "example-proxy1.com", indicative of an address of a device via which the message is transferred. From this, the server 2 which received the request finds out via which device (which is the proxy 3a in the present case) the request was transmitted.

(Response 124 to transmit content from server 2 to proxy 3a)

**[0125]** As illustrated in (e) of Fig. 8, the HTTP message, which serves as the response 124 to transmit the content 1 from the server 2 to the proxy 3a, contains (i) a response line, (ii) headers, and (iii) a body containing the "content 1."

**[0126]** The response line illustrated in (e) of Fig. 8 contains information indicating that the request has been successfully received, that is, information indicating that the content thus requested is to be transmitted. Specifically, the response line is described in the form of "status number (space) response message".

**[0127]** The headers contain information regarding a content to be transmitted. In the example illustrated in (e) of Fig. 8, the headers include (i) a "Date" header indicative of date and time when the content was transmitted, (ii) a "Cache-Control" header issuing an instruction regarding a cache of the content, and (iii) a "Content-Type" header indicative of a type of the content to be transmitted.

**[0128]** In the example, the "Cache-Control" header contains "must-revalidate" causing the proxy 3a to confirm, before the cache of the content is transmitted to other devices, whether or not the content to be transmitted is the latest data. The "Content-Type" header contains "video/mp4" indicating that the content is video data in MP4 format.

**[0129]** The body illustrated in (e) of Fig. 8 contains "[binary-data: content1]" indicative of the data of the content 1.

(Response 126 to transmit content from proxy 3a to client 4a)

**[0130]** As illustrated in (f) of Fig. 8, the HTTP message, which serves as the response 126 to transmit the content 1 from the proxy 3a to the client 4a, contains a response line, headers, and a body.

**[0131]** The response line illustrated in (f) of Fig. 8 contains information indicating that the request has been successfully received, that is, information indicating that the content thus requested is being transmitted. Specifically, the response line is described in the form of "status number (space) response message".

**[0132]** The headers contain information regarding a content to be transmitted. In the example illustrated in (f)

of Fig. 8, the headers include (i) a "Cache-Control" header issuing an instruction regarding the cache of the content to be transmitted, (ii) a "Content-Type" header indicative of a type of the content, and (iii) a "Via" header indicative of a transmission path of the message.

[0133] As in the case of the response 124, the "Cache-Control" header and the "Content-Type" header illustrated in (f) of Fig. 8 contain "must-revalidate" and "video/mp4", respectively. A "Via" header (i) indicates that a request containing the "Via" header has been transmitted via a certain device and (ii) contains address information indicative of an address of the device via which the request has been transmitted. The response 124 contains address information, "example-proxy1.com", indicative of an address of the proxy 3a which is a device via which the response 124 was transmitted.

[0134] The body illustrated in (f) of Fig. 8 contains actual data (binary data) of the content 1. The "[binary-data: content1]" in (f) of Fig. 8 indicates the data of the content 1.

[HTTP message in sessions 150 and 160]

[0135] The following description will discuss, with reference to Fig. 10, the HTTP messages in the sessions 150 and 160. Note that (a) through (c) and (f) of Fig. 10 correspond to (a) through (c) and (f) of Fig. 8, respectively, and that Fig. 10 is different from Fig. 8 only in that the content 1 and the proxy 3a in Fig. 8 are changed to a content 2 and a proxy 3b in Fig. 10, respectively. Therefore, description of the HTTP messages serving as the request 151, the response 153, the request 161, and the response 166 will be omitted here.

(Request 163 requesting content from proxy 3b to server 2 with condition)

[0136] As illustrated in (d) of Fig. 10, the HTTP message, which serves as the request 163 (i) transmitted from the proxy 3b to the server 2 and (ii) requesting the content 2 with conditions, contains a request line and headers.

[0137] The request line illustrated in (d) of Fig. 10 contains "GET" followed by a URL, which GET indicates a method for acquiring a content and which URL is of the content to be requested. Specifically, the URL is described in the form of "/content name."

[0138] The headers illustrated in (d) of Fig. 10 include (i) an "If-Modified-Since" header which requests a latest version of the content if the content is updated after date and time recorded in the "If-Modified-Since" header, (ii) an "Accept" header, (iii) a "Host" header, and (iv) a "Via" header. The "If-Modified-Since" header contains date and time "Sun, 31 May 2013 15:03:08 GMT" when the proxy 3b cached the content 2 requested by the request 163. This allows the server 2, which is a device that has received the request 163, to determine, based on the date and time when the proxy 3b cached the content 2, whether or not the content 2 stored in the proxy 3b is the

latest data.

(Response 165 transmitted from server 2 to proxy 3b)

5 [0139] As illustrated in (e) of Fig. 10, the HTTP message, which serves as the response 165 (i) transmitted from the server 2 to the proxy 3b and (ii) indicating that the content 2 stored in the proxy 3b is the latest data, contains a response line and a header.

10 [0140] The response line illustrated in (e) of Fig. 10 contains (i) the response message that the proxy 3b has not updated the content since the date and time when the proxy 3b cached the content and (ii) a status number of the response message. Specifically, the response line is described in the form of "status number (space) response message."

15 [0141] The header illustrated in (e) of Fig. 10 is a "Date" header indicative of date and time when the response 165 was transmitted.

20

<Embodiment 2>

[0142] Embodiment 2 of the present invention illustrates an example where a client 4 has a function as and acts as a proxy so that it is possible to widely distribute (i) a processing load of a server 2 and (ii) a network load which is used to transmit data from the server 2.

25 [0143] More specifically, in Embodiment 2, the client 4, which includes a storage section, (i) caches an acquired content in the storage section, (ii) specifies, based on response messages previously transmitted from the server 2, a device (proxy 3 or client 4) that (possibly) possesses the content, and then (iii) transmits, to a device (client 4) which has requested the content, an instruction to acquire the content from the device thus specified.

30 [0144] That is, in Embodiment 2, the client 4 (i) is a device that requests a content and then acquires the content and (ii) acts as (a) a relaying device (proxy) that stores the content thus requested and then transfers the content to another device or (b) a playing device that acquires the content thus requested and then plays the content.

35 [0145] Note that (i) the proxy 3 and a client 4 acting as a proxy are hereinafter each referred to as a relaying device and (ii) a client 4 that acquires a requested content and then plays the content is hereinafter referred to as a content playing device.

40 [0146] The following description will discuss Embodiment 2 with reference to Figs. 11 through 19. Embodiment 2 is (i) different from Embodiment 1 only in that the client 4 in Embodiment 2 has a function also as a proxy and (ii) similar to Embodiment 1 in regard to the rest of the points. Therefore, the following description will mainly discuss the point in which Embodiment 2 is different from Embodiment 1.

45

50

55

[Outline of Content Distribution System 1a]

**[0147]** First, an outline of a content distribution system 1a of Embodiment 2 will be described with reference to Fig. 11. Fig. 11 is a view illustrating a schematic configuration of the content distribution system 1a, and illustrates main configuration of devices constituting the content distribution system 1a. As illustrated in Fig. 11, the content distribution system 1a includes the server 2, the proxy 3, a client 4c, and a client 4d. The content distribution system 1a further includes (i) a content storage section 5 connected to the server 2, (ii) a cache storage section 6 connected to the proxy 3, and (iii) client storage sections 8c and 8d connected to the client 4c and the client 4d, respectively.

**[0148]** Hereinafter, the client storage sections 8c and 8d are generally referred to as a client storage section 8.

**[0149]** Since the proxy 3 of Embodiment 2 is similar to the proxy 3 of Embodiment 1 in terms of a configuration and an operation process, the details of the proxy 3 of Embodiment 3 will be omitted here.

[Server 2]

**[0150]** The server 2 of Embodiment 2 has a configuration identical to that of the server 2 of Embodiment 1. However, since the client 4 of Embodiment 2 acts as a proxy in some cases, part of operations of a response executing section 15, a relaying device specifying section 16, and a content-storage-location information generating section 17 of Embodiment 2 are different from the operations of the corresponding members of Embodiment 1.

**[0151]** The response executing section 15 (i) receives, via a server communication section 11, a request message requesting transmission of a content, which request message has been transmitted from a relaying device or a content playing device and then (ii) transmits, to the relaying device or the content playing device depending on which one of the devices transmitted the request message, a response message in response to the request message thus received.

**[0152]** Specifically, the response executing section 15, which is ready to receive a request message to request to transmit a content, (i) receives the request message via the server communication section 11 and then (ii) refers to a header of the request message thus received, so as to determine whether or not the request message was transmitted via a relaying device. For example, the response executing section 15 can be configured to receive a request message and then to (i) determine, in a case where the request message contains a "Via" header (transmission path information), that the request message has been transferred from a relaying device that had received the request message from a content playing device or (ii) determine, in a case where the request message does not contain the "Via" header, that the request message was directly transmitted from the content play-

ing device.

**[0153]** In a case where a request message is not one transmitted from a relaying device (i.e. the response executing section 15 receives the request message directly from a content playing device which is the sender of the request message), the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify address information (e.g. a URI of the proxy 3 or the client 4 etc.) of a relaying device that (possibly) possesses a content requested by the content playing device. Then, the response executing section 15 receives, from the relaying device specifying section 16, the address information of the relaying device that possesses the content requested by the content playing device, and then transmits, in response to the request message, a response message to instruct the content playing device to acquire the content from the relaying device. For example, by supplying the address information to the "Location" field with use of "305 USE Proxy", the response executing section 15 transmits, to the content playing device, an instruction (as a response message) to resend the request message to the relaying device that (possibly) possesses the content requested by the content playing device.

**[0154]** In a case where the request message thus received is one transmitted via the relaying device (i.e. the response executing section 15 receives the request message from the relaying device), the response executing section 15 proceeds to determine whether or not the request message is a conditional request such as "If-Modified-Since". If the request message is not provided with a conditional request, the response executing section 15 reads out, from the content storage section 5, a content requested by the request message, and then transmits, to the relaying device, the content in response to the request message.

**[0155]** Meanwhile, in a case where the request message is a conditional request, the response executing section 15 proceeds to determine whether or not the data of the content the relaying device possesses is the latest. To put it another way, the response executing section 15 determines whether or not the data of the content the relaying device possesses is identical to that stored in the content storage section 5.

**[0156]** In a case where the data of the content possessed by the relaying device from which the request message was transmitted, is not the latest data of the content, the response executing section 15 reads out the requested content from the content storage section 5, and then transmits the content thus read out, to the relaying device in response to the request message. In a case where the data of content possessed by the relaying device from which the request message was transmitted, is the latest data of the content, the response executing section 15 transmits, to the relaying device, a response message (in response to the request message) indicating that the relaying device possesses the latest data. Examples of such a response message encompass "304



NOT modified".

**[0157]** As in the case of Embodiment 1, the response executing section 15 also creates a transmission log based on the response message thus transmitted, and then stores the transmission log in a transmission log storage section 12.

**[0158]** Note that, as in the case of Embodiment 1, in a case where the response executing section 15 transmits the content to a relaying device from which a request message has been transmitted, "Cache-Control: must-revalidate" or "Cache-Control: proxy-revalidate", for example, may be added to the header of the content. Accordingly, in a case where the relaying device, which has acquired the content, transmits the content in response to another request, it is possible to always cause the relaying device to transmit, to the server 2, an inquiry as to whether or not the content is the latest, before the relaying device transmits the content.

**[0159]** The relaying device specifying section 16 is for determining, in accordance with an instruction from the response executing section 15, a relaying device that (possibly) possesses a content requested by a content playing device.

**[0160]** Specifically, the relaying device specifying section 16 receives an instruction from the response executing section 15, and accordingly reads out, from a content-storage-location information storage section 13, content-storage-location information. By referring to the content-storage-location information, the relaying device specifying section 16 specifies address information of a relaying device associated with the content, as address information of the relaying device from which the content requested by a content playing device is to be acquired. Thereafter, the relaying device specifying section 16 transmits the address information thus specified to the response executing section 15.

**[0161]** In a case where content-storage-location information for a content requested by a content playing device is not stored in the content-storage-location information storage section 13, the relaying device specifying section 16 can (i) specify predetermined (default) address information as address information of a relaying device from which the content is to be acquired, which predetermined address information is specific to each content requested by a content playing device or (ii) randomly select, out of relaying devices connected to the server 2 via a network 7, a relaying device from which the content is to be acquired.

**[0162]** In a case where the content-storage-location information storage section 13 stores a plurality of pieces of content-storage-location information containing address information of relaying devices which are each associated with a content requested by a content playing device, the relaying device specifying section 16 can (i) specify the pieces of address information as address information from which the content is to be acquired and then (ii) create a list of the relaying devices from which the content is to be acquired, which relaying devices are

indicated by the respective pieces of address information thus identified. In a case where the relaying device specifying section 16 specifies the plurality of pieces of address information as address information each indicative of a location from which the content is to be acquired, the relaying device specifying section 16 can create a list of content storage location by assigning priorities to the plurality of pieces of address information.

**[0163]** As in the case of Embodiment 1, the relaying device specifying section 16 may (i) select, based on date and time contained in content-storage-location information, a relaying device from which a content is to be acquired or (ii) select, based on physical or network-structural distances between a content playing device and relaying devices, a relaying device from which a content is to be acquired. The relaying device specifying section 16 may also select, in view of the load status of each relaying device, a relaying device which (i) (possibly) possesses a content requested by a content playing device and (ii) has a low load. To be more specific, the relaying device specifying section 16 can select, by referring to a transmission log stored in the transmission log storage section 12, address information of a relaying device from which the content is acquired. In addition, the relaying device specifying section 16 may select, in view of the number of accesses to the server 2 from the relaying devices access, a relaying device that (possibly) possesses a content requested by a content playing device. Note that a process in Embodiment 2 carried out by the relaying device specifying section 16 in order to determine which relaying device (possibly) possesses a content requested by a content playing device is identical to a process in Embodiment 1 carried out by the relaying device specifying section 16 in order to determine which proxy 3 (possibly) possesses a content requested by the client 4. Hence, description of the process in Embodiment 2 will be omitted here.

**[0164]** The content-storage-location information generating section 17 is for (i) generating, from a transmission log stored in the transmission log storage section 12, content-storage-location information for determining address information of a relaying device that (possibly) possesses a content and then (ii) storing the information in the content-storage-location information storage section 13. Unlike the case of Embodiment 1, the content-storage-location information generating section 17 in accordance with Embodiment 2 includes, as a device that (possibly) possesses a content, not only the proxy 3 but also the client 4.

**[0165]** Specifically, the content-storage-location information generating section 17 generates content-storage-location information by associating (A) a content transmitted from the server 2, (B) address information to which the content was transmitted, and (C) the date and time when the content was transmitted from the server 2, wherein the content, the address information, and the date and time are specified by referring to a transmission log (content transmission log), which indicates that the

server 2 has transmitted the content to a relaying device. This is because the relaying device, to which the content has been transmitted, is considered to store the content.

**[0166]** The content-storage-location information generating section 17 generates content-storage-location information by associating (A) a content confirmed as a latest version, (B) address information of a device (relaying device) which possesses the content, and (C) the date and time when the server 2 transmitted a response message that indicates that the content possessed by the relaying device is the latest version, wherein the content, the address information, and the date and time are specified by referring to a transmission log (version notification log), which indicates that the server 2 has transmitted the response message. This is because the relaying device, to which the response message has been transmitted, stores the latest version of the content.

**[0167]** The content-storage-location information generating section 17 can generate content-storage-location information by associating (A) a content, (B) address information to which a content acquisition instruction for the content was transmitted, and (C) date and time when the response message was transmitted, wherein the content, the address information, and the date and time are specified by referring to a transmission log (acquisition instructing log), which indicates that the server 2 transmitted to a content playing device the content acquisition instruction to specify a device from which the content is to be acquired. This is because the content playing device, to which the content acquisition instruction has been transmitted, is highly likely to store the content.

**[0168]** There is a possibility that a playing device, to which a content acquisition instruction for a content has been transmitted, fails to acquire the content. Therefore, it is possible to generate content-storage-location information by utilizing both a content transmission log and an acquisition instructing log. That is, in a case where an acquisition instructing log for a content and a content transmission log indicative of transmission of the content from the server 2 to a relaying device are both available, it is possible to generate content-storage-location information by associating together (i) the content, (ii) an address to which the content acquisition instruction has been transmitted, and (iii) date and time contained in the content transmission log. This is because (a) a playing device is considered to request the content from the server 2 in a case where a relaying device that a content acquisition instruction designates as a device from which the content is to be acquired does not store the content and (b), in a case where the server 2 responds to the request for the content, the content is transmitted to the playing device via a relaying device.

**[0169]** The content-storage-location information generating section 17 can generate content-storage-location information with use of an acquisition instructing log and a version notification log. That is, in a case where there exist (i) a content transmission log which indicates transmission of a content acquisition instruction instructing a

playing device to acquire a content from a certain relaying device and (ii) a version notification log which indicates that the server 2 notified the relaying device that the content possessed by the relaying device is the latest version of the content, the content-storage-location information generating section 17 can generate content-storage-location information by associating together (i) the content, (ii) an address of the playing device to which the content acquisition instruction was transmitted, and (iii) date and time contained in the version notification log.

**[0170]** The content-storage-location information generating section 17 may generate content-storage-location information (i) whenever a certain transmission log is added to the transmission log storage section 12 or (ii) by reading out a transmission log at the regular intervals.

**[0171]** Examples of a specific operation of the content-storage-location information generating section 17 in accordance with Embodiment 2 will be described below with reference to Figs. 12 and 13. Fig. 12 is a table illustrating an example of a transmission log stored in the transmission log storage section 12. Fig. 13 is a table illustrating an example of content-storage-location information stored in the content-storage-location information storage section 13. An example of the operation of the content-storage-location information generating section 17 for generating the content-storage-location information illustrated in Fig. 13 will be described below with reference to the transmission log illustrated in Fig. 12.

**[0172]** First, the content-storage-location information generating section 17 generates content-storage-location information 45 of Fig. 13 by associating together, with use of a transmission log (content transmission log) 42 in which "200 OK" is contained, (i) "content 1", (ii) http://example-proxy1.com, and (iii) "Sun, 31 May 2013 13:53:38 GMT", each of which is contained in the transmission log 42.

**[0173]** Then, the content-storage-location information generating section 17 generates content-storage-location information 46 from (i) a transmission log (acquisition instructing log) 41 in which "305 Use Proxy (proxy 1)" is contained and (ii) the transmission log 42 (a) in which "200 OK" is contained, (b) which has a content ID identical to that of the transmission log 41, and (c) which indicates that "proxy 1" contained in the transmission log 41 has been responded. In other words, the content-storage-location information 46 illustrated in Fig. 13 is generated by associating together "content 1", "http://example-client1.com" which is an address (contained in the transmission log 41) of a designation and "Sun, 31 May 2013 13:53:38 GMT" which is a date and time (contained in the transmission log 42) when the request was transmitted.

**[0174]** Note that, since "http://example-proxy1.com" is also considered to store "content 1", "http://example-proxy1.com" can be added to address information contained in the content-storage-location information 46.

**[0175]** Thereafter, the content-storage-location information generating section 17 generates content-storage-

age-location information 47 illustrated in Fig. 13 by associating together, with use of a transmission log (version notification log) 44 in which "304 Not Modified" is contained, (i) "content 1", (ii) "http://example-client1.com", and (iii) "Mon, 01 Jun 2013 08:05:30 GMT", each of which is contained the transmission log 44.

**[0176]** Finally, the content-storage-location information generating section 17 generates content-storage-location information 46 from (i) a transmission log (acquisition instructing log) 43 in which "305 Use Proxy (client 1)" is contained and (ii) the transmission log 44 (a) in which "304 Not Modified" is contained, (b) which has a content ID identical to that of the transmission log 43, and (c) which indicates that "client 1" contained in the transmission log 43 has been responded. To put it another way, the content-storage-location information 47 is generated by associating together "content 1", "http://example-client2.com" which is an address (contained in the transmission log 43) of a recipient of a request for a content, and "Mon, 01 Jun 2013 08:05:30 GMT" which is a date and time (contained in the transmission log 44) when the request was transmitted. Note that, since "http://example-client1.com" also is expected to store "content 1", "http://example-client1.com" can be added to address information contained in the content-storage-location information 47.

[Client 4]

**[0177]** The client 4 in accordance with Embodiment 2 functions also as a proxy. Therefore, unlike the case of Embodiment 1, a client control section 32 in accordance with Embodiment 2 includes a response/request executing section 35 instead of a request executing section 33. A client storage section 8 for caching contents is connected to the client 4. Alternatively, the client storage section 8 can be provided inside the client 4.

**[0178]** In a case where the client 4 acts as a content playing device, the response/request executing section 35 executes an operation similar to that of the request executing section 33.

**[0179]** Specifically, the response/request executing section 35 generates a request message requesting transmission of a content, transmits the request message to the server 2 via a client communication section 31, and then receives a response message as a response to the request message. That is, the response/request executing section 35 receives, as the response message from the server 2, (i) information that specifies a relaying device and (ii) an instruction to transmit, to the relaying device, a request message requesting the content. The response/request executing section 35 receives the response message, and then transmits, to the relaying device thus specified by the server 2, the request message requesting transmission of the content. Thereafter, the response/request executing section 35 receives the content from the relaying device, and then stores in the content in the client storage section 8.

**[0180]** In a case where the client 4 acts as a relaying device, the response/request executing section 35 executes an operation similar to that of a request executing section 23 of the proxy 3.

**[0181]** Specifically, the response/request executing section 35 acts as a device for transmitting, to a content playing device, a content specified by a request message requesting the content, which request message was transmitted from the content playing device.

**[0182]** More specifically, the response/request executing section 35, which is ready to receive from a content playing device a request message to request transmission of a content, (i) receives a request message from the content playing device via the client communication section 31 and then (ii) determine whether or not the content thus requested is stored in the client storage section 8.

**[0183]** In a case where the content is not stored in the client storage section 8, the response/request executing section 35 (i) examines a Host field of the request message and then (ii) transmits a request message to a server 2 whose URL is shown in the Host field, which request message requests the content requested by the content playing device. Thereafter, the response/request executing section 35 acquires the content from the server 2, stores the content thus acquired in the client storage section 8, and then transmits the content to the content playing device.

**[0184]** In a case where the content is stored in the client storage section 8, the response/request executing section 35, in order to transmit, to the server 2, an inquiry as to whether or not the content stored in the client storage section 8 is the latest data, (i) creates a request message with a conditional request by adding, to a regular request message, "If-Modified-Since" containing information about time at which the relaying device (client 4) has acquired the content and then (ii) transmits, to the server 2, the request message thus created.

**[0185]** Upon receipt of a "304 NOT Modified" response message from the server 2 in response to the request message, the response/request executing section 35 (i) determines that the content stored in the client storage section 8 is the latest data, (ii) reads out the content from the client storage section 8, and then (iii) transmits the content to the content playing device.

**[0186]** On the other hand, in a case where the content stored in the client storage section 8 is not the latest data, the server 2 transmits, to the response/request executing section 35, the latest version of the content in response to the request message, and then the response/request executing section 35 receives the content thus transmitted. Thereafter, the response/request executing section 35 stores the content thus received in the client storage section 8, and then transmits the content to the content playing device.

**[0187]** Note that the response/request execution section 23 may transmit the content to the content playing device while the content contains, in its header,

"Cache-Control: must-revalidate" or "Cache-Control" which has been added by the server 2. Accordingly, in a case where the content, which has been acquired by the content playing device serving as a destination of a content, is transmitted in response to another request, it is possible to cause the content playing device to always transmits, to the server 2, an inquiry as to whether or not the content is the latest version before the content playing device transmits the content.

[Processes carried out by each device]

**[0188]** The following description will discuss, with reference to Figs. 14 through 16, processes carried out by the server 2 and the client 4. Since a process carried out by the proxy 3 in Embodiment 2 is identical to that in Embodiment 1, description of the process will be omitted here.

[Process carried out by server 2]

**[0189]** The operation carried out by the server 2 will be described first with reference to Fig. 14. Fig. 14 is a flow-chart showing an exemplary process which is carried out by the server 2.

**[0190]** The response executing section 15 is ready to receive a request message to request to transmit a content. When the response executing section 15 receives, via the server communication section 11, the request message to request to transmit the content (S501), the response executing section 15 refers to a header of the request message thus received, so as to determine whether or not the request message thus received is transmitted via a relaying device (S502).

**[0191]** In a case where the content playing device has transmitted the received request message (NO in S502), the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify a relaying device from which the content playing device acquires the content. In response to the instruction from the response executing section 15, the relaying device specifying section 16 reads out content-storage-location information from the content-storage-location information storage section 13 (S503). With reference to the content-storage-location information thus read out, the relaying device specifying section 16 specifies, as address information of a device from which the content should be acquired, a URI of the relaying device associated with the content that the content playing device requests (S504). The relaying device specifying section 16 transmits such specified address information to the response executing section 15. The response executing section 15 transmits, to the content playing device, from the relaying device indicated by an address contained in the address information which has been specified by the relaying device specifying section 16, a response message to instruct the content playing device to acquire a requested content (S505). The response executing sec-

tion 15 creates a transmission log based on the response message which has been transmitted to the content playing device, and adds the transmission log thus created to the transmission log storage section 12 (S510).

**[0192]** Meanwhile, in a case where the relaying device transmits the request message thus received (YES in S502), the response executing section 15 then determines whether or not the received request message is a conditional request such as "If-Modified-Since" (S506).

In a case where the received request message is not a conditional request (NO in S506), the response executing section 15 reads out, from the content storage section 5, the content requested by the relaying device, and, in response to the request message from the relaying device, transmits the content thus read out to the relaying device (S507). After that, the response executing section 15 creates a transmission log based on the response message which has been transmitted to the relaying device, and adds the transmission log thus created to the transmission log storage section 12 (S510).

**[0193]** In a case where the received request message is provided with a conditional request (YES in S506), the response executing section 15 determines whether or not data of the content held by the relaying device is the latest (S508). In a case where the data of the content kept by the relaying device is not the latest (NO in S508), the response executing section 15 reads out, from the content storage section 5, the content requested by the relaying device, and, in response to the request message from the relaying device, transmits to the relaying device the content thus read out (S507). Meanwhile, in a case where the data of the content held by the relaying device is the latest (YES in S508), the response executing section 15 transmits, to the relaying device, a response message that the data of the content held by the relaying device is the latest, in response to the request message from the relaying device (S509). Then, the response executing section 15 adds, to the transmission log storage section 12, the response message which has been transmitted to the relaying device (S510).

**[0194]** Note that, in a case where the response executing section 15 transmits the content to the relaying device in S507, "Cache-Control: must-revalidate" or "Cache-Control: proxy-revalidate", for example, is added to the header of the content. Accordingly, in a case where the relaying device serving as a destination of a content transmits the content held by the cache storage section 6 in response to another request, it is possible to always cause the relaying device to transmit, to the server, an inquiry as to whether or not the content is the latest, before the relaying device transmits the.

[Process carried out by client 4]

**[0195]** A process carried out by the client 4 will be described next with reference to Figs. 15 and 16. As described earlier, the client 4 acts as a relaying device or as a content playing device. An operation of the client 4

in a case where the client 4 acts as a content playing device will be described first with reference to Fig. 15.

(Process carried out by client 4 as content playing device)

**[0196]** Fig. 15 is a flowchart showing an exemplary process which is carried out by the client 4 serving as a content playing device. As shown in Fig. 15, the response/request executing section 35 transmits, to the server 2, a request message to request to transmit a content (S521). In response to the request message, the response/request executing section 35 receives a response message to instruct the response/request executing section 35 to acquire a content from a designated relaying device (S522). In response to such an instruction from the server 2, the response/request executing section 35 transmits the request message to the relaying device designated by the server 2 (S523). Then, in response to the request message, the response/request executing section 35 acquires the content from the relaying device which has received the request message (S524).

(Process carried out by client 4 as content playing device)

**[0197]** The description will discuss, with reference to Fig. 16, a process of the content 4 in a case where the client 4 acts as a content playing device. Fig. 16 is a flow chart showing an exemplary process which is carried out by a client 4 acting as a content playing device.

**[0198]** As shown in Fig. 16, the response/request executing section 35 is ready to receive, from a content playing device (the client 4 different from the one described in the previous example), a request message requesting transmission of a content. The response/request executing section 35 receives the request message from the content playing device via the client communication section 31 (S541), and then determines whether or not the content requested by the request message is stored in the client storage section 8 (S 542).

**[0199]** In a case where the content is not stored in the client storage section 8 (NO in S542), the response/request executing section 35 examines a Host field of the request message, and then transmits a request message to a server 2 whose URL is shown in the Host field, which request message requests the content requested by the content playing device (S543). The response/request executing section 35 acquires the content from the server 2 (S544), stores the content thus acquired in the client storage section 8 (S545), and then transmits the content to the content playing device (S546).

**[0200]** In a case where the content requested by the content playing device is stored in the content storage section 8 (YES in S542), the response/request executing section 35 (i) creates a request message provided with a conditional request by adding, to a regular request message, "If-Modified-Since" containing information about time at which the client device 4 has acquired the content

and then (ii) transmits the request message thus created to the server 2 (S547).

**[0201]** Upon receipt of a "304 NOT Modified" response message from the server 2 (YES in S548), the response/request executing section 35 reads out the content stored in the client storage section 8, and then transmits the content to the content playing device (S426).

**[0202]** On the contrary, in a case where the content stored in the client storage section 8 is not the latest data, the server 2 transmits, to the response/request executing section 35, the latest version of the content as a response to the request message (carrying the conditional request), and then the response/request executing section 35 receives the content thus transmitted (S544). The response/request executing section 35 stores the content thus received in the client storage section 8 (S545), and then transmits the content to the content playing device (S546).

**[0203]** Note that, in S546, the response/request execution section 23 transmits the content to the content playing device while the content contains, in its header, "Cache-Control: must-revalidate" or "Cache-Control" which has been added by the server 2. Accordingly, in a case where the content, which has been acquired by the client 4 serving as a destination of a content, is transmitted in response to another request, it is possible to keep such a state that the content playing device should always transmit, to the server 2, as to whether or not the content is the latest version, before the content playing device transmits the content.

[Example 2]

**[0204]** The following description will further discuss Embodiment 2 in more detail with reference to Fig. 17 illustrating Example 2. Example 2 will illustrate an exemplary operation of a content distributing system 1a which instructs, to a content playing device (which is a client 4d), from which relaying device (client 4c) the content playing device should acquire a content. Fig. 17 is a view illustrating an example operation sequence of the content distribution system 1a in Example 2.

**[0205]** Note that Example 2 is carried out on the following conditions. A content 1 is stored in the content storage section 5, and both the client storage sections 8c and 8d and the cache storage section 6 caches no content. Further, the server 2 is set so that the content 1 is acquired from the proxy 3 as its default. Furthermore, when the process shown in Fig. 17 is started in Example 2, a transmission log and content-storage-location information are not stored in the transmission log storage section 12 and the content-storage-location information storage section 13, respectively. Still further, the response executing section 15 creates the transmission log of Fig. 12 every time when a response message is transmitted, and the content-storage-location information generating section 17 generates the content-storage-location information of Fig. 13 every time when a content transmission

log or a version notification log (transmission log whose "transmitted contents" is "200 OK" or "304 Not Modified") is added to the transmission log storage section 12.

**[0206]** Further, Example 2 is carried out on condition that a sequence starting from a time at which the content playing device transmits a request message and ending at a time at which the content playing device receives a response message in response to the request message is considered as a single session.

**[0207]** As shown in Fig. 17, in a session 210, the client 4c transmits, to the server 2, a request message to request to transmit the content 1 (request 211). In the server 2 which has received the request 211, the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify a relaying device to acquire the content 1. In response to the instruction, the relaying device specifying section 16 confirms whether or not the content-storage-location information is stored in the content-storage-location information storage section 13 (process 212). Here, the content-storage-location information storage section 13 does not contain the content-storage-location information indicative of the proxy 3 to acquire the content 1, the relaying device specifying section 16 specifies the default proxy 3a as the proxy 3 to acquire the content 1. The response executing section 15 transmits, to the client 4a, the response message to instruct the client 4c to acquire a requested content from the proxy 3 indicated by address information specified by the relaying device specifying section 16 (response 213). Then, the response executing section 15 creates a transmission log based on the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 214).

**[0208]** Then, in the session 220, the client 4c, which has received the response 213, transmits, to proxy 3, the request message to request to transmit the content 1 (request 221). The proxy 3 received the request 221 confirms whether or not the content 1 is stored in the cache storage section 6 (process 222). Here, the content 1 is not stored in the cache storage section 6, so that the proxy 3 transmits, to the server 2, the request message based on the request 221 (request 223). In the server 2 which has received the request 223 from the proxy 3, the response executing section 15 transmits the content 1 to the proxy 3 (response 124). The proxy 3a received the response 124 stores the acquired content 1 in the cache storage section 6 and caches the content 1 (process 225). After that, the proxy 3a transmits, to the client 4a, the content 1 as a response to the request 221 (response 226). When acquiring the content 1 from the proxy 3, the client 4c stores the content 1 thus acquired in the client storage section 8c and caches the content 1 (process 227). Note that, after transmitting the response 224, the response executing section 15 creates a transmission log based on the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 228). Further, the content-storage-location information generating section 17 gen-

erates content-storage-location information by associating the content 1 and address information of the proxy 3 with each other, and causes the content-storage-location information to be stored in the content-storage-location information storage section 13. Furthermore, the content-storage-location information generating section 17 generates content-storage-location information by associating the content 1 and address information of the client 4c with each other, and causes the content-storage-location information to be stored in the content-storage-location information storage section 13 (process 229).

**[0209]** When the session 220 is completed, the content 1 is in a state of being cached in the cache storage section 6 of the proxy 3 and in the cache storage sections 8c of the client 4c. That is, the content-storage-location information storage section 13 stores (A) the content-storage-location information in which the content 1 and the proxy 3a are associated with each other and (B) content-storage-location information in which the content 1 and the client 4c are associated with each other. That is, in this state, not only the proxy 3 but also the client 4c potentially serves as a relaying device from which the content 1 is acquired.

**[0210]** Next, in a session 230, the client 4d transmits, to the server 2, a request message to request to transmit the content 1 (request 231). In the server 2 which has received the request 231, the response executing section 15 transmits, to the relaying device specifying section 16, an instruction to specify a relaying device from which the content 1 is to be acquired. The relaying device specifying section 16 receives the instruction, and then checks the content-storage-location information in the content-storage-location information storage section 13 (process 232). At this point, (i) the information stored in the content-storage-location information storage section 13 identifies the proxy 3b and the client 4c both as locations where the content 1 is stored and (ii) the content-storage-location information pointing to the client 4c is later than the content-storage-location information pointing to the proxy 3. Therefore, the relaying device specifying section 16 specifies the client 4c as a relaying device from which the content 1 is to be acquired. The response executing section 15 transmits, to the client 4d, a response message with an instruction to acquire the content 1 from the client 4c thus specified by the relaying device specifying section 16 (response 233). The response executing section 15 creates a transmission log based on the response message, and then adds the transmission log to the transmission log storage section 12 (process 234).

**[0211]** In a session 240, the client 4d (which has received the response 233) transmits, to the client 4c, a request message (request 241) in accordance with the instruction of the server 2, which request message request to transmit the content 1. The client 4c receives the request 241, and then checks whether or not the content 1 is stored in the client storage section 8c (process 242). Since the content 1 is already stored in the client storage section 8c, the client 4c transmits a conditional

request message (request 243) to the server 2 in order to examine whether or not the content 1 stored in the client storage section 8c is the latest data, which conditional request message is created by adding "If-Modified-Since" to a request message based on the request 241. The request 243 contains a "Via" header, and is a conditional request. Hence, the response executing section 15 in the server 2 that has received the request 243 verifies whether or not the content 1 stored in the client 4c is the latest data (process 244). The response executing section 15 determines that the content 1 stored in the client 4c is the latest data, and then transmits, to the client 4c, a response message in which "304 NOT Modified" is contained (response 245). Upon receipt of the response 245, the client 4c reads out the content 1 stored in the client storage section 8c, and then transmits, to the client 4d, the content 1 in response to the request 241 (response 246). Subsequent to the transmission of the response 245, the response executing section 15 creates a transmission log based on the response message thus transmitted, and then adds the transmission log to the transmission log storage section 12 (process 247). The content-storage-location information generating section 17 generates content-storage-location information by associating the content 1 with the address information of the client 4c, and then stores the content-storage-location information in the content-storage-location information storage section 13. The content-storage-location information generating section 17 also creates content-storage-location information associating the content 1 with address information of the client 4d, and then stores the content-storage-location information in the content-storage-location information storage section 13 (process 248).

**[0212]** In Embodiment 2, the client 4 acts as a content playing device or as a relaying device. This gives a greater number of candidates that the server 2 can designate as a relaying device. Therefore, a processing load of the server 2 and a network load which is used to transmit data from the server 2 can be diluted by more widely distributing the loads over the network so that the loads are also shared by the clients 4 and networks between a client 4 and another client 4.

[HTTP Message in Example 2]

**[0213]** The details of the requests and responses illustrated in the operation sequence diagram of Fig. 17 will be illustrated in Figs. 18 and 19. Figs. 18 and 19 are views each illustrating example HTTP messages transmitted/received as requests and responses, Fig. 18 particularly illustrating HTTP messages in the session 210 and 220, and Fig. 19 particularly illustrating HTTP messages in the sessions 230 and 240.

**[0214]** (a), (b), (c), (d), (e), and (f) of Fig. 18 illustrate HTTP messages of the request 211, the response 213, the request 221, the request 223, the response 224, and the response 226 of Fig. 17, respectively.

**[0215]** (a), (b), (c), (d), (e), and (f) of Fig. 19 illustrate HTTP messages of the request 231, the response 233, the request 241, the request 243, the response 245, and the response 246 of Fig. 17, respectively.

**[0216]** (a) through (f) of Fig. 18 correspond to (a) through (f) of Fig. 8, respectively, and Fig. 18 is different from Fig. 8 only in that the client 4a and the proxy 3a in Fig. 8 are changed to the client 4c and the proxy 3 in Fig. 18, respectively. Also, (a) through (f) of Fig. 19 correspond to (a) through (f) of Fig. 10, respectively, and Fig. 19 is different from Fig. 10 only in that the content 2, the client 4a, and the proxy 3b in Fig. 10 are changed to the content 1, the client 4d, and the client 4c in Fig. 19, respectively. Description other than the above difference has been already made.

<Embodiment 3>

**[0217]** Embodiment 3 of the present invention will discuss an example where (i) a server 2 specifies a plurality of relaying devices from which a content is acquired and (ii) a content playing device acquires a content by selecting a relaying device from the plurality of relaying devices thus specified, in order to more widely distribute a load of a network between the content playing device and the relaying device.

**[0218]** Specifically, in Embodiment 3, a client 4, which is the content playing device, (i) selects one relaying device from the plurality of relaying devices specified by the server 2 and (ii) acquires a content from the relaying device thus selected. In a case where a delay occurs in regard to the acquisition of the content while the content is being acquired, the client 4 reselects another relaying device from the plurality of relaying devices specified by the server 2, so as to change the relaying device from which the content is acquired.

**[0219]** Embodiment 3 will be described below with reference to Figs. 20 through 28. Embodiment 3 differs from Embodiment 2 only in a configuration of the client 4 and a data format of a content which is managed by the server 2, and Embodiment 3 is identical to Embodiment 2 in other points. As such, the difference of Embodiment 3 from Embodiment 2 will be mainly described below.

[Outline of content distribution system 1b]

**[0220]** First, an outline of a content distribution system 1b of Embodiment 3 will be described with reference to Fig. 20. Fig. 20 is a view illustrating an outline of the content distribution system 1b in accordance with Embodiment 3 and a main configuration of devices constituting the content distribution system 1b. As illustrated in Fig. 20, the content distribution system 1b includes the server 2, proxies 3a, 3b and 3c, and clients 4e and 4f. Further, the content distribution system 1b includes: a content storage section 5 which is connected to the server 2; cache storage sections 6a, 6b, and 6c which are connected to the proxies 3a, 3b, and 3c, respectively;

and client storage sections 8e and 8f which are connected to the client 4e and 4f, respectively.

**[0221]** The proxy 3 of Embodiment 3 is identical to the proxy 3 of Embodiment 1 in a configuration and an operation process. As such, details of the proxy 3 will not be discussed here.

[Server 2]

**[0222]** The server 2 of Embodiment 3 (i) has the same configuration with that of the server 2 of Embodiment 2 and (ii) executes the same operation as that of the server 2 of Embodiment 2. However, in order to allow the content playing device to select a relaying device from which a content is acquired, the server 2 of Embodiment 3 (i) specifies a plurality of relaying devices which (possibly) possess a content that the content playing device requests, (ii) presents, to the content playing device, a content storage location list containing address information of the plurality of relaying devices thus specified, and (iii) instructs the content playing device to acquire a content from one of the relaying devices which exist at an address indicated by the address information contained in the content storage location list thus presented.

**[0223]** Specifically, upon receipt of a request message to request a content directly from the content playing device, a response executing section 15 transmits, to a relaying device specifying section 16, an instruction to specify address information of a relaying device (e.g., URI of the proxy 3 or the client 4, etc.) which (possibly) possesses the content that the content playing device requests. Then, the response executing section 15 receives a content storage location list, from the relaying device specifying section 16, the content storage location list containing a plurality of pieces of address information of the relaying device which possesses the content that the content playing device requests. Then, the response executing section 15 transmits a response message to the content playing device in response to the request message from the content playing device, the response message instructing to acquire the content that the content playing device requests, from one of the relaying devices of the address indicated by the address information contained in the content storage location list which is created by the relaying device specifying section 16.

**[0224]** Note that, since other processes of the response executing section 15 are identical to those of Embodiment 2, those processes will not be discussed here.

**[0225]** In accordance with the instruction from the response executing section 15, the relaying device specifying section 16 (i) specifies a plurality of relaying devices which (possibly) possess a content that the content playing device requests and (ii) creates a content storage location list containing address information of the plurality of relaying devices thus specified.

**[0226]** Specifically, the relaying device specifying section 16 reads out content-storage-location information from the content-storage-location information storage

section 13 in accordance with the instruction of the response executing section 15. With reference to the content-storage-location information thus read out, the relaying device specifying section 16 (i) specifies a plurality of pieces of address information of a relaying device associated with the content that the content playing device requests and (ii) creates a content storage location list containing the plurality of pieces of address information thus specified. The relaying device specifying section 16 transmits the content storage location list thus created to the response executing section 15.

**[0227]** In a case where the content-storage-location information storage section 13 stores (i) no content-storage-location information containing the content that the content playing device requests or (ii) only one piece of content-storage-location information containing the content that the content playing device requests, the relaying device specifying section 16 may (a) specify predetermined (default) address information as address information of the device from which the content that the content playing device requests is available, and (b) create a content storage location list containing a plurality of pieces of address information. Further, in a case where the content-storage-location information storage section 13 stores (i) no content-storage-location information containing the content that the content playing device requests or (ii) only one piece of storage location information containing the content that the content playing device requests, the relaying device specifying section 16 may (a) specify, at random, from relaying devices connected to the server 2 via a network 7, address information of the device from which the content is acquired and (b) create a content storage location list.

**[0228]** Moreover, the relaying device specifying section 16 may create a content storage location list by adding priorities to the pieces of address information thus specified. In this case, as with Embodiment 2, a priority may be determined on a basis of date and time contained in the content-storage-location information, a physical or a network-structural distance between the content playing device and the relaying device, a load status of the relaying device, a transmission log which is stored in a transmission log storage section 12, or the like.

[Client 4]

**[0229]** The client 4 of Embodiment 3, unlike that of Embodiment 2, when functioning as a content playing device, (i) receives a content storage location list which is transmitted from the server 2 and (ii) acquires a content that the client 4 requests, from one of relaying devices which exist at an address indicated by the address information contained in the content storage location list thus received.

**[0230]** In the client 4 illustrated in Fig. 20, a client control section 32 includes, in addition to a response/request executing section 35, a client status determining section 36 and a relaying device selecting section (relaying de-



vice changing means) 37, both of which are not included in Embodiment 2.

**[0231]** The response/request executing section 35 receives, from the server 2, as a response message, (i) a content storage location list and (ii) an instruction to transmit a request message for requesting a content to one of the relaying devices which exist at the address indicated by the address information contained in the content storage location list.

**[0232]** When the response/request executing section 35 receives the content storage location list and the instruction, the relaying device selecting section 37 selects one of pieces of the address information contained in the content storage location list that the response/request executing section 35 receives. The relaying device selecting section 37 transmits, to the response/request executing section 35, an instruction to acquire a content from a relaying device of an address indicated by the address information thus selected.

**[0233]** In a case where priority is not added to the address information contained in the content storage location list (in a case where the server 2 does not instruct an order of selecting a relaying device from which a content is acquired), the relaying device selecting section 37 may select (i) a relaying device, at random, on a basis of the address information contained in the content storage location list, (ii) a relaying device on a basis of a predetermined rule (default), or (iii) a relaying device having a shortest physical or network-structural distance to the client 4.

**[0234]** Meanwhile, in a case where priority is added to the address information contained in the content storage location list, the relaying device selecting section 37 selects address information having a highest priority.

**[0235]** When the relaying device selecting section 37 receives, from the client status determining section 36, delay information indicating that an acquisition speed at which the response/request executing section 35 acquires a content (receiving speed required to acquire a content) is slower than a predetermined receiving speed, the relaying device selecting section 37 (i) changes the relaying device from which the content is acquired, from the relaying device that the relaying device selecting section 37 selects, to another relaying device which exists at the address indicated by the address information contained in the content storage location list and (ii) transmits, to the response/request executing section 35, an instruction to acquire a content from the another relaying device thus changed to.

**[0236]** Here, in a case where priority is not added to the address information contained in the content storage location list, the relaying device selecting section 37 may select (i) a relaying device, at random, on a basis of the address information contained in the content storage location list, (ii) a relaying device on a basis of a predetermined rule (default), or (iii) a relaying device having a second shortest physical or a network-structural distance to the client 4.

**[0237]** Meanwhile, in a case where priority is added to the address information contained in the content storage location list, the relaying device selecting section 37 selects address information having a second highest priority.

**[0238]** The client status determining section 36 detects an occurrence of a predetermined event. Specifically, the client status determining section 36 detects an event that a content is received with delay when the content playing device acquires, from the relaying device, the content that the content playing device requires. The event indicates (i) a network communication status between the content playing device and the relaying device and/or (ii) a size of a load of the relaying device. When detecting the event that the content is received with delay, the client status determining section 36 transmits, to the relaying device selecting section 37, delay information indicating the event thus detected.

[Format of content]

**[0239]** Next, in Embodiment 3, a format of a content which is stored in the content storage section 5 will be described with reference to (a) of Fig. 21. (a) of Fig. 21 is a view illustrating an example format of the content.

**[0240]** As illustrated in (a) of Fig. 21, a media file, which is data indicating the content, is fragmented by a predetermined unit. The unit is not particularly limited but the media file may be fragmented (i) by a time unit such as one minute or (ii) by a unit of GOP (group of picture) in an image coding.

**[0241]** In the following description, the fragment is referred to as a movie fragment, and an MP4 file is used as a specific example of a media file which is constituted by the movie fragment.

**[0242]** In a case where the MP4 file is used as a media file, a fragment which is constituted by "moof" storing header information which manages an image and a sound in the fragment and "mdat" storing data such as an image and a sound which are played by a client corresponds to the movie fragment.

**[0243]** Here, as to the MP4 file, in addition to "moof" and "mdat," information (e.g., image resolution, profile information, etc.) related to an entire media file, that is, information (play information) required for a formatting of a content playing section 34 in the client 4, is stored in "moov," which is different from the "moof" or the "mdat."

**[0244]** Accordingly, it is necessary to notify, before a play starts, the client 4 of the play information stored in "moov". The play information stored in "moov" may be notified in a procedure different from that for the movie fragment, and it is not always necessary to include "moov" in the movie fragment. However, the following description will discuss an example where a first movie fragment in each of media files includes "moov." That is, "information required for the formatting of the playing device" illustrated in (a) of Fig. 21 is "moov."

**[0245]** As illustrated in Fig. 21, consecutive reference

numerals "movie fragment 1", "movie fragment 2" ... are assigned to a movie fragment in each of the media files in the order from the first movie fragment. Note that each of the movie fragments includes image data for one minute.

[Format for transmission of content]

**[0246]** Next, in the content distribution system 1b of the present invention, a transmission unit of the content will be described. In the content distribution system 1b, among the server 2, the proxy 3, and the client 4, the content is (i) divided by a unit which is referred to as media segment and (ii) transmitted with use of HTTP.

**[0247]** (b) of Fig. 21 is a view illustrating a concept of a media segment which is treated by the content distribution system 1b as a transmission unit of the content and showing an exemplary transmission unit of the content in the content distribution system 1b.

**[0248]** The media segment is constituted to include at least one movie fragment. In other words, a content is constituted by one or more media segments and each of the media segments is constituted by one or more movie fragments. Generally, each of the media segments, in a predetermined content, is configured to include two or more movie fragments each of which has consecutive playing time. However, the media segment may be constituted by (i) one movie fragment or (ii) two or more movie fragments each of which has no consecutive playing time.

**[0249]** Specifically, (b) of Fig. 21 shows an example where a plurality of movie fragments are combined so as to constitute one media segment. This makes it possible to reduce the number of messages which transmit a content as compared with a case where each of the plurality of movie fragments is transmitted separately. As such, it is possible to send a content efficiently.

**[0250]** In an example of (b) of Fig. 21, one media segment "media segment 1" is constituted by combining "movie fragments 1 through 60", and another media segment "media segment 2" is constituted by combining "movie fragments 61 through 120". Note that a media segment should include two or more movie fragments each of which has consecutive playing time in the predetermined content, and the number of movie fragments included in one media segment is not particularly limited. In Embodiment 3, as illustrated in (b) of Fig. 21, one media segment includes 60 movie fragments.

[Process carried out by each device]

**[0251]** Next, a process which is carried out by the server 2 and the client 4 functioning as a content playing device will be described with reference to Figs. 22 and 23. A process which is carried out by the proxy 3 of Embodiment 3 is identical to that of Embodiment 1, and a process which is carried out by the client 4 functioning as a relaying device of Embodiment 3 is identical to that

of Embodiment 2. As such, those processes will not be discussed here.

[Process carried out by server 2]

5

**[0252]** First, a process which is carried out by the server 2 will be described with reference to Fig. 22. Fig. 22 is a flowchart showing an exemplary process which is carried out by the server 2.

10

**[0253]** The response executing section 15 is ready to receive a request message for requesting to transmit a content, and upon receipt of the request message for requesting to transmit the content, via the server communication section 11 (S601), the response executing section 15 determines whether or not the request message thus received is transmitted from a relaying device by referring to a header of the request message thus received (S602).

15

20

**[0254]** In a case where a subject which transmits the request message thus received is a content playing device (in a case -where an answer for S602 is No), the content playing device instructs the relaying device specifying section 16 to specify a plurality of relaying devices from which the content playing device acquires the content.

25

30

The relaying device specifying section 16 reads out content-storage-location information from the content-storage-location information storage section 13 in response to the instruction of the response executing section 15 (S603). With reference to the content-storage-location information thus read out, the relaying device specifying section 16 (i) specifies a plurality of URIs of a relaying device associated with the content that the content playing device requests and (ii) creates a content storage location list containing the plurality of pieces of address information thus specified (S604).

35

40

The relaying device specifying section 16 transmits the content storage location list thus created to the response executing section 15. The response executing section 15 transmits, to the content playing device, a response message instructing to acquire the content that the content playing device requests from one of the relaying devices which exist at an address indicated by the address information contained in the content storage location list which is created by the relaying device specifying section 16 (S605).

45

50

The response executing section 15 (i) creates a transmission log on a basis of the response message which is sent to the content playing device and (ii) adds the transmission log thus created to the transmission log storage section 12 (S610).

55

60

**[0255]** Processes (S606 through S609) for a case where the sender of the received request message is the relaying device is identical to those (S506 through S509 in Fig. 14) for the server 2 of Embodiment 2. As such, those processes will not be discussed here.

65

70

75

80

85

90

95

100

105

110

115

120

125

130

135

140

145

150

155

160

165

170

[Process carried out by client 4 functioning as content playing device]

**[0256]** Next, a process which is carried out by the client 4 functioning as a content playing device will be described with reference to Fig. 23. Fig. 23 is a flowchart showing an exemplary process which is carried out by the client 4 functioning as a content playing device.

**[0257]** The response/request executing section 35 transmits a request message for requesting to transmit a content to the server 2 (S621). In response to the request message, the response/request executing section 35 receives a response message containing (i) a content storage location list and (ii) an instruction to transmit, to one of relaying devices which is of an address indicated by address information contained in the content storage location list, a request message for requesting a content (S622).

**[0258]** When the response/request executing section 35 receives the response message, the relaying device selecting section 37 selects one of pieces of the address information contained in the content storage location list which the response/request executing section 35 receives (S623). The relaying device selecting section 37 instructs the response/request executing section 35 to acquire a content from the relaying device is of the address indicated by the address information thus selected.

**[0259]** The response/request executing section 35 which is instructed from the relaying device selecting section 37 transmits a request message to the relaying device that the relaying device selecting section 37 selects (S624). First, upon receipt of the request message, the relaying device sends, as a response message, a header in response to the request message thus received. As such, the response/request executing section 35 receives the header (S625) and notifies the client status determining section 36 of the receipt of the header.

**[0260]** Upon receipt of the notification, the client status determining section 36 initializes a timer and starts counting in order to evaluate receiving time for a movie fragment. Moreover, the client status determining section 36 initializes a variable (counter) ( $N_1 = N_2 = 0$ ) which is used for the evaluation (S626). For example, an initial value for the timer may be a value calculated by subtracting a predetermined threshold  $T_{th}$  from a value of a time stamp of a movie fragment which is most recently sent.

**[0261]** The relaying device that received the request message transmits, after the header, as a response message in response to the request message received in S624, a body for which a plurality of movie fragments are multiparted. The response/request executing section 35 receives a movie fragment (S627) and notifies the client status determining section 36 of the receipt of the movie fragment.

**[0262]** Furthermore, the response/request executing section 35 (i) determines, on a basis of a value of "Content-Type" header contained in the header thus received, that the movie fragment is received in a MIME multipart

format, (ii) notify the content playing section 34 of the receipt of the movie fragment, and also (iii) transmits the movie fragment thus received to the content playing section 34. Then, the content playing section 34 (a) specifies a time stamp of the movie fragment in reference to an "X-Timestamp" header of the movie fragment thus received and (ii) plays the movie fragment with reference to the time stamp.

**[0263]** Here, the client status determining section 36 evaluates receiving time (S628). Specifically, the client status determining section 36 compares time  $t$  indicated by the timer which starts counting in S626 and a time stamp  $T_{fr}$  (value of X-Timestamp) of the movie fragment which is received in S627.

**[0264]** In a case where the comparison shows that  $t < T_{fr} - T_{th}$ , the client status determining section 36 determines that it is sufficiently earlier than predetermined receiving time (an event indicating a good communication status is detected), and a process proceeds to S629. In contrast, in a case where  $T_{fr} + T_{th} > t$ , the client status determining section 36 determines that a delay occurs (an event indicating a poor communication status is detected), and the process proceeds to S630. Meanwhile, in a case where neither of the above cases applies ( $|t - T_{fr}| \leq T_{th}$ ), the process proceeds to S631.

**[0265]** That is, after starting a receipt of a first movie fragment contained in the response message, the client status determining section 36 counts time  $t$  until starting a receipt of a next fragment. When (i)  $T_{fr}$  indicates a difference between a value of a time stamp associated with the first movie fragment and a value of a time stamp associated with the next movie fragment and (ii)  $T_{th}$  indicates zero or more predetermined threshold, (a) in a case where  $T_{fr} + T_{th} > t$ , the client status determining section 36 determines that the event indicating a poor communication status is detected, and (b) in a case where  $t < T_{fr} - T_{th}$ , the client status determining section 36 determines that the event indicating a good communication status is detected.

**[0266]** In S629, the client status determining section 36 increments  $N_1$  which is a counter for the number of times that the movie fragment is received sufficiently earlier than the predetermined receiving time. Then, the process proceeds to S631.

**[0267]** In S630, the client status determining section 36 increments  $N_2$  which is a counter for the number of times that the movie fragment is received with delay. Then, the process proceeds to S631.

**[0268]** In S631, the response/request executing section 35 confirms whether or not all of the movie fragments contained in the media segment specified by the request which is transmitted in S624, and in a case where an unreceived movie fragment is found (in a case where an answer for S631 is No), the process returns to S627.

**[0269]** Meanwhile, in a case where the response/request executing section 35 confirms that all of the movie fragments are already received (in a case where an answer for S631 is Yes), the response/request executing

section 35 confirms whether or not all of the media segments of the content which is subject to be requested are received (S632), and in a case where it is confirmed that all of the media segments has been received (in a case where an answer for S632 is Yes), the process is completed. Meanwhile, in a case where an unreceived media segment is found (in a case where an answer for S632 is No), the client status determining section 36 determines, whether or not  $N_2 - N_1 > 0$  with use of  $N_1$  and  $N_2$  which are calculated in S629 and S630 (S633).

**[0270]** In a case where  $N_2 - N_1 > 0$  is not true, that is, in a case where  $N_2 - N_1 \leq 0$ , the relaying device is not changed, and a request message in which a media segment number is incremented is transmitted to the relaying device which is selected in S623 (S624). As a response message in response to the request message, the response/request executing section 35 receives a header (S625) and notifies the client status determining section 36 of the receipt of the header. Then, in order to evaluate receiving time for a movie fragment contained in a next media segment, the client status determining section 36 reinitializes the timer, so as to start counting of the time. Moreover, the client status determining section 36 also reinitializes the variable (counter) ( $N_1 = N_2 = 0$ ) which is used for the evaluation. Then, the response/request executing section 35 receives the movie fragment contained in the next media segment (S627).

**[0271]** Meanwhile, in a case where  $N_2 - N_1 > 0$ , the client status determining section 36 transmits, to the relaying device selecting section 37, delay information indicating that an acquisition of the content is delayed. Upon a receipt of the delay information from the client status determining section 36, the relaying device selecting section 37 selects other address information which is (i) contained in the content storage location list and (ii) different from the address information which is currently selected (S634). The relaying device selecting section 37 instructs the response/request executing section 35 to acquire a content from a relaying device which exists at an address indicated by the address information thus selected.

**[0272]** Upon receipt of the instruction from the relaying device selecting section 37, the response/request executing section 35 retransmits, to the relaying device selected by the relaying device selecting section 37, a request message for which the media segment number is incremented (S624).

**[0273]** The description above describes an example where the relaying device from which the content is acquired is reselected per media segment (S634). Note, however, that the relaying device may be reselected per movie fragment. In that case, for example, in accordance with a result of the evaluation of the receiving time in S628, it is possible (i) to change the relaying device, (ii) to transmit a new request message to the relaying device thus changed, and (iii) to cancel a subsequent movie fragment which is transmitted on a basis of a request message which is sent first.

**[0274]** For example, the client status determining sec-

tion 36 may count, in S629 and S630, (i) the number  $N_1$  of movie fragments which are received sufficiently earlier than the predetermined time and (ii) the number  $N_2$  of movie fragments which are transmitted with delay. After a transmission of one media segment is completed, in a case where  $N_2 - N_1 > 0$ , the client status determining section 36 may determine that there is an overall delay (an event indicating a poor communication status is detected) and notify the relaying device selecting section 37 of the determination.

**[0275]** Moreover, in the description above, in S633, the client status determining section 36 determines whether or not  $N_2 - N_1 > 0$ , so as to determine whether or not a delay occurs when the content is acquired. However, the process is not limited to this. For example, it is also possible (i) to predetermine an upper limit for  $N_2$ , which is a counter for the number of times that the movie fragment is received with delay, and (ii) to cancel a subsequent movie fragment when a value of  $N_2$  exceeds the value thus predetermined, so as to switch the relaying device to another relaying device.

[Example 3]

**[0276]** The present embodiment is further described below with reference to Example 3 illustrated in Fig. 24. Example 3 shows an operation example of a content distribution system 1b that transmits, to a content playing device serving as a client 4e, an instruction to acquire a content from any one of address information contained in a content-storage-location list. Fig. 24 is a view illustrating an example operation sequence of the content distribution system 1b in Example 3.

**[0277]** Note that Example 3 is carried out on the following conditions. A content 1 in a format illustrated in Fig. 21 is already stored in the content storage section 5, and the client storage sections 6a, 6c, and 8f. Further, the server 2 generates a content-storage-location list by adding priorities to the plurality of pieces of address information. Furthermore, when the process shown in Fig. 25 is started in Example 3, the transmission logs 51 to 53 stores the transmission log storage section 12 and the content-storage-location information storage section 13, respectively. Still further, the response executing section 15 creates the transmission logs 54 to 56 of Fig. 25 every time when a response message is transmitted, and pieces of the content-storage-location information 64 to 67 generating section 17 generates the content-storage-location information of Fig. 26 every time when a content transmission log or a version notification log (transmission log whose "transmitted contents" is "200 OK" or "304 Not Modified") is added to the transmission log storage section 12.

**[0278]** Further, Example 3 is carried out on condition that a sequence starting from a time at which the content playing device transmits a request message and ending at a time at which the content playing device receives a response message in response to the request message

is considered as a single session.

**[0279]** As shown in Fig. 24, in a session 310, the client 4e transmits, to the server 2, a request message to request to transmit the content 1 (request 311). In the server 2 which has received the request 311, the response executing section 15 transmits, to a plurality of relaying devices specifying section 16, an instruction to specify a relaying device to acquire the content 1. In response to the instruction, the relaying device specifying section 16 confirms whether or not the content-storage-location information is stored in the content-storage-location information storage section 13 (process 312). At this point, the content-storage-location information storage section 13 contains a proxy 3a, a proxy 3c, and a client 4f serving as a storage location associated with the content 1. Therefore, the relaying device specifying section 16 creates the content storage location list by (i) adding priorities 1 through 3 to the proxy 3a, the client 4f, and the proxy 3c, respectively, on a basis of date and time contained in the content-storage-location information so that the content storage location list added with priority contains the address information of the proxy 3, the client 4f, and the proxy 3c. The response executing section 15 transmits, to the client 4e, the response message to instruct to (i) select, in order of high priority, the address information contained in the content storage location list created by the relaying device specifying section 16 and (ii) acquire the content from a relaying device existing at an address indicated by the address information thus selected (response 313). In other words, the response executing section 15 transmits to the client 4e an instruction to acquire the content first from (i) the proxy 3a and, in a case where the content cannot be acquired from the proxy 3a or a speed to acquire the content is slow, secondarily from (ii) the client 4f, and lastly from (iii) the proxy 3c. Sequentially, the response executing section 15 creates the transmission log on the basis of the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 314).

**[0280]** In the client 4e which received the response 313, a relaying device selecting section 37 selects the proxy 3a having a highest priority, on the basis of the instruction from the server 2, as a relaying device in the address information of a device from which the content is acquired (process 320). Then, the relaying device selecting section 37 transmits to a response/request executing section 35 an instruction to acquire the content 1 from the proxy 3a.

**[0281]** Next, in a session 330, the response/request executing section 35 thus instructed by the relaying device selecting section 37, transmits to the proxy 3a the request message for requesting to transmit the content 1 (a request 331). In response to the request 331, the proxy 3a checks whether the content 1 is stored in the cash storage section 6a or not (process 332). Because the content 1 is already stored in the cash storage section 6a, in order to check with the server 2 if the contents 1

stored in the cash storage section 6a is the latest data, the proxy 3a transmits to the server 2 a conditional request that is the request message being associated with the request 331 added with "If\_Modified-Since" according to the request 331. (a request 333). In the server 2 which received the request 333, because the request 333 is the conditional message containing a "Via" header, the response executing section 15 determines whether or not the content 1 kept by the proxy 3 is the latest data (process 334). The response executing section 15 determines that the content 1 kept by the proxy 3a is the latest data, and transmits the response message "304 NOT Modified" to the proxy 3a (a response 335). In response to the response 335, the proxy 3a retrieves the content 1 stored in the cash storage section 6a and, as a response to the request 331, transmits the content 1 thus retrieved to the client 4e (a response 336). The client 4e acquires "movie fragment1 to 60" one by one, in response to the response 336. After transmitting the response 335, the response executing section 15 creates the transmission log on the basis of the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 337). The content-storage-location information generation section 17 generates the content-storage-location information by associating the content 1 with the address information of the proxy 3a, and stores the content-storage-location information in the content-storage-location information storage section 13. In addition, the content-storage-location information generation section 17 generates the content-storage-location information by associating the content 1 and the address information of the client 4e, and stores the content-storage-location information in the content-storage-location information storage section 13 (process 338).

**[0282]** A client status determining section 36 evaluates receiving time at every receipt of movie fragments. If the receiving time is evaluated as  $N_2 - N_1 > 0$  at the time of receipt of one media segment, at this point, the client status determining section 36 determines that a receiving speed required to acquire the content from the proxy 3a is slower than a predetermined receiving speed. Then the client status determining section 36 gives information of such delay to the relaying device selecting section 37 (process 340).

**[0283]** The relaying device selecting section 37, in response to the information of the delay from the client status determining section 36, changes the proxy 3a which is currently selected as the relaying device, from which the content is required, to the client 4f having a second priority to be selected as the relaying device (process 350). In other words, the relaying device selecting section 37 selects the client 4f as the device from which the content is acquired. Continuously, the relaying device selecting section 37 transmits to the response/request executing section 35 an instruction to acquire the content 1 from the client 4f.

**[0284]** Next, in a session 360, the "movie fragments 1

to 60" has been received by the response/request executing section 35 according to the instruction from the relaying device selecting section 37. Thus, the response/request executing section 35 transmits to the client 4f the request message to request to transmit the content 1 from "movie fragment61" onward (request 361). In response to the request 361, the client 4f determines whether or not the content 1 is stored in the client storage section 8f (process 362). The content 1 is already stored in the client 8f. Thus, in order to check with the server 2 whether or not the content 1 stored in the client storage section 8f is the latest data, the client 4f transmits the conditional request, which is the request message associated with the request 361, added with "If\_Modified-Since" (request 363). In the server 2 which received the request 363, because the request 363 is the conditional request message containing the "Via" header, the response executing section 15 determines that the content 1 kept by the client 4f is the latest data, and transmits the response message "304 NOT Modified" to the client 4f (response 365). Sequentially, the client 4f, in response to the response 365, retrieves data of the content 1 "movie fragment61" or later stored in the client storage section 8f. Then the client 4f transmits the retrieved content 1 "movie fragment61" or later to the client 4e one by one (response 366). In response to the response 366, the client 4e acquires "movie fragment61" or later one by one. Further, after transmitting the response 365, the response executing section 15 creates the transmission log on the basis of the response message thus transmitted, and adds the transmission log to the transmission log storage section 12 (process 367). In addition, the content-storage-location information generating section 17 generates the content-storage-location information by associating the content 1 with the address information of the client 4e so as to store the content-storage-location information in the content-storage-location information storage section 13 (process 368).

**[0285]** As mentioned above, in the present embodiment, a content playing device receives from a server 2 a content storage location list containing address information of a plurality of relaying devices, and then acquires a content from any one of the relaying devices indicated by the address information contained in the content-storage-location list. In a case where a receiving speed required to acquire the content is slow, the relaying device, from which the content is acquired, is changed to another relaying device which is indicated by the address information contained in the content storage location list. This makes it possible to distribute a load of network (particularly, a network between the content playing device and the relaying device) efficiently time-wise, thus realizing a control in more detail in the content distribution system 1b. Accordingly, it is possible to keep higher quality of service for a larger number of content playing devices.

[HTTP Messages in Example 3]

**[0286]** Next, the detail of the requests and the responses used in the operation sequence illustrated in Fig. 24 is illustrated in Fig. 27 and Fig. 28. Fig. 27 and Fig. 28 show examples of HTTP messages which are transmitted/received as a request or a response. Fig. 27 shows example HTTP messages in the session 310 and the session 330. Fig. 28 shows examples of HTTP messages in the session 360.

**[0287]** (a), (b), (c), (d), (e), and (f) of Fig. 27 show the HTTP messages of the request 311, the response 313, the request 331, the request 333, the response 335, and the response 336 of Fig. 24, respectively.

**[0288]** (a), (b), (c), and (d) of Fig. 28 show HTTP messages of the request 361, the response 363, the request 365, the request 363, the request 365, and the request 336 of Fig. 24, respectively.

[HTTP messages in sessions 310 and 330]

**[0289]** HTTP messages in the session 310 and the session 330 will be described with reference to Fig. 27. Since (c), (d), and (e) of Fig. 27 correspond to (c), (d), and (e) of Fig. 10, respectively. A content 2, a client 4a, and a proxy 3b of FIG. 10 are merely changed to a content 1 in MIME multipart format, a client 4e, and a proxy 3a of Fig. 27, respectively. Thus, the description of the HTTP messages of the request 331, the request 333, and the response 336 in the session 330 is not repeated here. In addition, (a), (b), (c), and (d) of Fig. 28 correspond to (c), (d), (e), and (f) of Fig. 27, respectively. Since "media segment1" of the content 1 and a proxy 3a of Fig. 27 are merely changed to a "media segment2" and a proxy 4f in Fig. 28, respectively. Therefore, the description of the HTTP message in the session 360 is not repeated here. Note also that, an element, which is more peculiar to the present invention, contained in an HTTP message is mainly described here. Thus, the description of well-known elements will be appropriately omitted.

(Request 311 Requesting for Content)

**[0290]** As shown in (a) of Fig. 27, a request line and a header are contained in an HTTP message which corresponds to the request 311 for requesting a "media segment1" of a content 1 from a server 2 by a client 4e.

**[0291]** In the request line in (a) of Fig. 27, information for specifying the content to be acquired is described after a "GET" indicating a method for acquiring the content. Specifically, the information is described in a form of "/content name/media segment number". In other words, the HTTP message (a) of Fig. 27 is a request for the "0(zero)"th "media segment 1" (first portion) of the "content 1".

**[0292]** Further, a header of (a) of Fig. 27 contains an "Accept" header indicative of a processible data format for the client 4e. The "Accept" header has information

"video/mp4" indicative of a moving image data of an MP4 format, and information "multipart/media-segment" indicative of a MIME multipart format. This enables the client 4e, which is a source that has transmitted the request, to inform the server 2, which is a receiver of the request, that the moving image data of the MP4 format is receivable in the MIME multipart format.

**[0293]** Furthermore, the header of (a) of Fig. 27 contains a "Host" header for specifying a server to which a request is transmitted. The "Host" header has a description "example.com" indicative of the address of the server 2.

(Response 313 for instructing on device from which content is acquired)

**[0294]** As shown in (b) of Fig. 27, a response line and a header are contained in an HTTP message corresponding to the response 313, in which the server 2 transmits to the client 4e an instruction on address information of a device from which the content 1 is acquired.

**[0295]** In the response line of (b) of Fig. 27, information for instructing to use the proxy is described in a form of "status number message".

**[0296]** Further, the header of (b) of Fig. 27 contains a "Location" header for specifying the relaying device to be used. That is, the HTTP message of (b) of Fig. 27 is the response instructing to request for the content 1 with use of the relaying device designated by the "Location" header mentioned below. In the example of (b) of Fig. 27, the "Location" header has the address information "http://example-proxy1.com" indicative of the address of the proxy 3a. This enables the client 4e to be informed of the address information of the relaying device (proxy 3a) from which the content 1 is requested.

**[0297]** Moreover, the header of (b) of Fig. 27 contains an "X-Alternative-Proxy-List" header indicative of the address information of other relaying devices which possesses the content 1. This "X-Alternative-Proxy-List" header describes address information "http://example-client2.com, http://example-proxy3.com" indicative of addresses of the other relaying devices (referring to the client 4f and the proxy 3c here). Thus, to the client 4e, which is a device received this response, the relaying devices (possibly) possessing the content 1 are presented, in addition to the relaying device specified by the "Location" header. Accordingly, the client 4e can select the relaying device, from which the content 1 is acquired, out of the relaying devices of the address information contained in the "Location" header or the "X-Alternative-Proxy-List" header. Note that an "X" in the description of the header indicates that the header was newly defined in the present embodiment.

(Response 336 for transmitting content from proxy 3a to client 4e)

**[0298]** As shown in (f) of Fig. 27, the HTTP message

contains a response line, a header, and a body are contained in the HTTP message corresponding to the response 336 for transmitting the "media segment 1" of the content 1 to the client 4e from the proxy 3a.

**[0299]** The response line of (f) of Fig. 27 describes information that the request has been received, which means the content thus requested is to be transmitted. Specifically, the information is described in the form of "status number response message".

**[0300]** The header has information about a content to be transmitted. In the example shown in the figure, a "Content-Type" header indicative of a type of the content to be transmitted, a "Content-Location" header indicative of a storage location of the content to be transmitted (e.g. URI), a "Cache-Control" indicative of an instruction regarding a cache of the content to be transmitted, a "Via" header indicative of a transmission path via which the message is transferred, and an "X-Media-Segment-Index" header indicative of a location of an entire content of the media segment to be transmitted are contained.

**[0301]** In the example shown in the figure, the "Content-Type" header describes "multipart/ media-segment" indicative of the MIME multipart format. Thus, a device which received this header (the client 4e) can recognize the media segment which was transmitted in MIME multipart format is the next content to receive. In addition, the header contains information "boundary=THIS#STRING#SEPARATES" indicating that a break point of the multipart format is "THIS#STRING#SEPARATES".

**[0302]** The "Content-Location" header describes the URI "http://www.example.com/content1/0". As mentioned earlier, the final number "0" of this URI stands for the "media segment1" which is the initial portion of the content, followed by a URI of the "media segment2" which is "http://www.example.com/content1//1". Thus, a media segment indicated by a serial number enables a device received the response (client 4e) to determine that a URI having an incremented number is the URI of a next media segment to request for.

**[0303]** The "Cache-Control" header describes "must-revalidate", and the "Via" header describes "example-proxy 1.com".

**[0304]** Further, the example in the figure contains the "X-Media-Segment-Index" header. The "X-Media-Segment-Index" header indicates a playing location of the media segment for the entire content. In (f) of Fig. 27, the header contains "1/60". This "1/60" means a first media segment out of 60 media segments in the entire content. According to this information, a full length of the content and a current playing location can be tracked of. It is also possible to access arbitrarily to any media segment in the content with reference to this information.

**[0305]** In the body, a plurality of movie fragments consisting media segments described in the MIME multipart format. Here, one media segment contains 60 movie fragments from 1 through 60.

**[0306]** Moreover, each of parts (each of the movie frag-

ments) can have a header. In the example shown in the figure, a "Content-Type" header indicating a type of the content of a movie fragment and an "X-Timestamp" header indicating a time stamp of the movie fragment are described. It is possible to specify a playing time (timing of starting a play) of the movie fragment without analyzing the movie fragment by referring to the time stamp indicated by the "X-Timestamp" header. Each of the parts contains a data entity (binary data) of the movie fragment of the respective parts.

[Case where client cannot process MIME multipart format data]

**[0307]** The Embodiment 3 has discussed the case where data in the MIME multipart format could have been processed by the client 4 serving as a content playing device, that is, the case where the "Accept" header of the request message has contained "multipart/mediasegment". However, it is also conceivable that the content playing device cannot process the data in the MIME multipart format (that is, the "multipart/mediasegment" is not contained in the "Accept" header of the request message).

**[0308]** In this case, it is advisable that a relaying device and a server 2 respond to a request not by the MIME multipart format but by one body which is a combination of all movie fragments in the media segment. This enables a content playing device which is not capable of processing data in the MIME multipart to play a content received.

<Embodiment 4>

**[0309]** Embodiment 4 of the present invention will discuss an example where (i) a single content is managed by a plurality of servers 2, (ii) a server 2 receives a request for the content, and designates, as a server from which the content thus requested is acquired, at least two servers 2 among the plurality of servers 2, and (iii) a content playing device selects one of the at least two servers 2 thus designated, so as to acquire the content. With the arrangement, it is possible to distribute, more widely, (a) a load of a network between the content playing device and a corresponding one of the plurality of servers 2, and (b) a process load of the corresponding one of the plurality of servers 2.

**[0310]** More specifically, in Embodiment 4, a client 4, which is the content playing device, (i) selects one of a plurality of servers 2 designated by a server 2 to which a request for a content has been transmitted, and (ii) acquires the content from the one of the plurality of servers 2 thus selected. In a case where acquisition of the content is delayed during a time period in which the client 4 acquires the content from the one of the plurality of servers 2 thus selected, the client 4 selects another one of the plurality of servers 2 designated by the server 2 to which the request for the content has been transmitted,

and switches, to the another one of the plurality of servers 2, the server 2 from which the content is acquired.

**[0311]** The following description deals with Embodiment 4 with reference to Figs. 29 through 34. Embodiment 4 is identical to Embodiment 3 except that (i) a content distribution system of Embodiment 4 includes a plurality of servers 2, (ii) an arrangement of each of the plurality of servers 2 of Embodiment 4 is different from that of a server 2 of Embodiment 3, and (iii) an arrangement of the client 4 is different from that of a client 4 of Embodiment 3. For this reason, the following description mainly deals with such differences between Embodiment 4 and Embodiment 3.

Outline of content distribution system 1]

**[0312]** First, the following description explains an outline of a content distribution system 1c of Embodiment 4 with reference to Fig. 29. Fig. 29 is a view illustrating the outline of the content distribution system 1c of Embodiment 4 and an arrangement of a main part of each of devices constituting the content distribution system 1c.

**[0313]** The content distribution system 1c includes servers 2a, 2b, and 2c, proxies 3a, 3b, and 3c, and clients 4g and 4h (see Fig. 29). Further, the content distribution system 1c includes (i) content storage sections 5a, 5b, and 5c, which are connected to the servers 2a, 2b, and 2c, respectively, (ii) cache storage sections 6a, 6b, and 6c, which are connected to the proxies 3a, 3b, and 3c, respectively, and (iii) client storage sections 8g and 8h, which are connected to the clients 4g and 4h, respectively.

**[0314]** The proxies 3a, 3b, and 3c of Embodiment 4 are identical to proxies 3a, 3b, and 3c of Embodiment 3 in arrangement and operational process. For this reason, details of the proxies 3a, 3b, and 3c are omitted here for the sake of simple explanation.

As to server 2]

**[0315]** Each of the servers 2a, 2b, and 2c of Embodiment 4 is different from the server 2 of Embodiment 3 in that each of the servers 2a, 2b, and 2c includes an acquisition location specifying section 18 in place of a relaying device specifying section 16. Further, a content-storage-location information storage section 13 of each of the servers 2a, 2b, and 2c stores not only content-storage-location information including address information of a replying device which possesses a content but also content-storage-location information including address information of a server 2 which possesses a content. Other than these points described above, each of the servers 2a, 2b, and 2c of Embodiment 4 has the same arrangement as that of the server 2 of Embodiment 3.

**[0316]** In order to cause the content playing device to select a sever 2 from which the content playing device acquires a content, each of the server 2a, 2b, and 2c of Embodiment 4 (i) identifies a plurality of servers 2, each



of which (possibly) possesses the content requested by the content playing device, (ii) presents, to the content playing device, a content storage server list including address information of each of the plurality of servers 2 thus identified, and (iii) instructs the content playing device to acquire the content from one of the plurality of servers 2 each of which is located at an address indicated by a corresponding piece of the address information included in the content storage server list thus presented.

**[0317]** Specifically, in a case where a response executing section 15 directly receives, from the content playing device, a request message which requests a content, the response executing section 15 instructs an acquisition location specifying section 18 to specify address information (e.g., an URI of a server 2) of servers 2, each of which (possibly) possesses the content requested by the content playing device. Then, in a case where the response executing section 15 receives, from the acquisition location specifying section 18, a content storage server list including the address information of the servers 2, each of which possesses the content requested by the content playing device, the response executing section 15 transmits, as a response to the request message received from the content playing device, to the content playing device, a response message which is an instruction to acquire such a requested content from one of the servers 2 each of which is located at an address indicated by a corresponding piece of the address information included in the content storage server list created by the acquisition location specifying section 18.

**[0318]** Further, in a case where the response executing section 15 receives, from a response executing section 15 of another server 2, an inquiry as to whether the server 2 possesses a certain content, the response executing section 15 checks whether or not the content is stored in a corresponding content storage section 5 connected to the server 2. Then, the response executing section 15 transmits, to the another server 2, a response indicating whether the server 2 possesses the certain content. For example, in a case where a response executing section 15 of the server 2a receives, from a response executing section 15 of the server 2b, an inquiry as to whether the server 2 possesses a certain content, the response executing section 15 of the server 2a (i) checks contents stored in the content storage section 5a, and (ii) transmits, to the server 2b, a response indicating whether the server 2 possesses the certain content.

**[0319]** Processes of the response executing section 15, other than the aforementioned process, are identical to those of a response executing section 15 of Embodiment 3, and therefore explanations of such processes are omitted here for the sake of simple explanation.

**[0320]** In accordance with the instruction received from the response executing section 15, the acquisition location specifying section 18 (i) identifies a plurality of servers 2, each of which (possibly) possesses the content requested by the content playing device, and (ii) creates a content storage server list including address informa-

tion of the plurality of servers 2 thus identified.

**[0321]** Specifically, on receipt of the instruction from the response executing section 15, the acquisition location specifying section 18 reads out content-storage-location information from a content-storage-location information storage section 13. The acquisition location specifying section 18 (i) refers to the content-storage-location information thus read out, (ii) identifies a plurality of pieces of address information, which are associated with the content requested by the content playing device, and (iii) creates the content storage server list including the plurality of pieces of address information thus identified. The acquisition location specifying section 18 transmits the content storage server list thus created to the response executing section 15.

**[0322]** There is a case where the acquisition location specifying section 18 checks the content-storage-location information storage section 13 for the content-storage-location information including the address information of the servers 2 but such content-storage-location information is not stored in the content-storage-location information storage section 13, for example. Further, there is also a case where the content-storage-location information stored in the content-storage-location information storage section 13 is old information (a time and date included in the content-storage-location information has been obtained before a predetermined time). In such cases, the acquisition location specifying section 18 updates the content-storage-location information which includes the address information of the servers 2, and is stored in the content-storage-location information storage section 13.

**[0323]** In a case where the acquisition location specifying section 18 determines that it is necessary to update the content-storage-location information, the acquisition location specifying section 18 transmits, to each of other servers 2 via the response executing section 15, an inquiry as to whether or not each of other servers 2 has the content thus requested. On the basis of a response received from each of other servers 2, the acquisition location specifying section 18 (i) identifies address information of a server 2 which made such a response that the server 2 has the content thus requested, (ii) creates a content storage server list including the address information thus identified, and (iii) notifies the response executing section 15 of the content storage server list. Further, on the basis of a response received from each of other servers 2, the acquisition location specifying section 18 creates such content-storage-location information that (i) the content thus requested, (ii) address information of a server 2 which made such a response that the server 2 has the content, and (iii) a time and date at which the response is received, are associated with each other. Then, the acquisition location specifying section 18 stores the content-storage-location information thus created in the content-storage-location information storage section 13.

**[0324]** Note that the acquisition location specifying

section 18 can update the content-storage-location information at predetermined intervals.

**[0325]** Furthermore, it is possible that the acquisition location specifying section 18 updates the content-storage-location information in such a manner that the acquisition location specifying section 18 (i) transmits, to each of other servers 2, an inquiry as to whether or not each of other servers 2 has the content, (ii) measures a time period from a time that the inquiry is transmitted to a time that the acquisition location specifying section 18 receives a response, and (iii) ranks other servers 2 on the basis of such time periods. That is, it is possible that the acquisition location specifying section 18 (i) sets a low rank to a server 2 with which the aforementioned time period is long (the server 2 which took a long time to make a response), and (ii) sets a high rank to a server 2 with which the aforementioned time period is short. The acquisition location specifying section 18 can cause address information of a server 2 included in the content storage server list thus created and a rank thus set to be associated with each other.

**[0326]** Moreover, in the same manner as Embodiment 3, it is possible to set a priority of a server 2 on the basis of a physical distance between the content playing device and the server 2, a network-structural distance between the content playing device and the server 2, a load status of the server 2, or the like.

**[0327]** Further, the acquisition location specifying section 18 can create not only the content storage server list which includes the address information of the servers 2 each having the content but also a content storage location list which includes address information of a relaying device which (possibly) possesses the content, in the same manner as Embodiment 3.

**[0328]** The content-storage-location information storage section 13 stores, in addition to the content-storage-location information including the address information of the relaying device, such content-storage-location information that (i) a content, (ii) address information of a server 2 possessing the content, and (iii) a time and date at which a response indicating that updating is executed is received from the server 2, are associated with each other. The content-storage-location information stored in the content-storage-location information storage section 13 can be data shown in Fig. 30, for example. Fig. 30 is a view showing an example of the content-storage-location information stored in the content-storage-location information storage section 13.

**[0329]** As shown in Fig. 30, the content-storage-location information is such that the following (i) through (iii) are associated with each other: (i) "Date" which indicates a time and date at which a content is acquired by a proxy 3 or a client 4, (ii) "Content ID" which indicates the content, and (iii) "Storage Location Address" which indicates address information of the proxy 3 possessing the content, address information of the client 4 possessing the content, or address information of the server 2 possessing the content.

**[0330]** Specifically, Fig. 30 shows, as an example, content-storage-location whether the server 2 possesses information 75 which indicates that a server 2, whose address is "http://srv2.exmample.com", possesses a content "content 1".

**[0331]** As described above, in short, according to the server 2 of Embodiment 4, the response executing section 15 determines whether or not the source of the request is (i) a relaying device which possesses the content thus requested, and transfers the content thus requested to a content playing device, or (ii) a content playing device which plays the content thus requested. Next, in a case where the response executing section 15 determines that the source of the request is the content playing device, the acquisition location specifying section (content-storage-location information acquisition means) 18 acquires, in response to the request, an address of a server 2 having the content thus request, among predetermined other servers 2. Then, the response executing section (content acquiring location designating means) 15 instructs the content playing device, which is the source of the request, to acquire the content from the server 2 indicated by the address acquired by the acquisition location specifying section.

**[0332]** Here, the predetermined other servers 2 described above are servers 2 which are connected to, via a network 7, a server 2 which has received the request, and are in a range determined in advance in accordance with a predetermined rule. For example, in a case where the server 2 which has received the request is the server 2a, the predetermined other servers 2 can be (i) a server 2b and a server 2c, (ii) a server 2b only, or (iii), in addition to the server 2b and the server 2c, all servers 2 with which the server 2a can communicate via a network.

**[0333]** Further, the acquisition location specifying section 18 (i) transmits, to each of the predetermined other servers 2 described above, an inquiry as to whether or not each of the predetermined other servers 2 has the content thus requested, and then (ii) acquires an address(es) of a server(s) 2 each of which makes such a response that the server 2 has the content thus requested, among the predetermined other servers 2.

**[0334]** Furthermore, the acquisition location specifying section 18 (i) creates such content-storage-location information that such acquired address(es) of the server (s) 2 each having the content, and content identification information indicating the content are associated with each other, and (ii) stores the content-storage-location information in the content-storage-location information storage section (storage section) 13.

**[0335]** Moreover, the acquisition location specifying section 18 reads out the content-storage-location information from the content-storage-location information storage section 13. In a case where the content-storage-location information thus read out includes the content identification information indicating the content thus requested, the acquisition location specifying section 18 acquires, from the content-storage-location information,

an address(es) associated with the content identification information. On the other hand, in a case where the content-storage-location information thus read out does not include the content identification information indicating the content thus requested, the acquisition location specifying section 18 transmits the inquiry described above, and acquires the address(es) of the server(s) 2 each having the content thus requested, among the predetermined other servers 2.

**[0336]** Further, the acquisition location specifying section 18 (i) acquires an address of each of a plurality of servers 2 each having the content thus requested, and (ii) creates a content storage server list including the address of each of the plurality of servers 2, and the content identification information indicating the content. Then, the response executing section 15 instructs the content playing device, which is the source of the request, to acquire the content from a server 2 indicated by an address included in the content storage server list created by the acquisition location specifying section 18.

**[0337]** Furthermore, the client 4 serving as a content playing device of Embodiment 4 (i) transmits a request for a content to a server 2, (ii) receives a content storage server list as a response to the request, and (iii) acquires the content thus requested from another server 2 indicated by an address included in the content storage server list thus received.

**[0338]** Specifically, the client 4 includes an acquisition location selecting section (acquisition location changing means) 38. The acquisition location selection section 38 switches the another server 2 (for example, the server 2b) from which the content is acquired to further another server 2 (for example, the server 2c) indicated by an address which is (i) included in the content storage server list and (ii) is different from the address of the another server 2, in a case where a receiving speed at which the content is acquired is slower than a predetermined receiving speed.

[Client 4]

**[0339]** In a case where the client 4 of Embodiment 4 serves as the content playing device, the client 4 receives a content storage server list from a server 2, and acquires a content thus requested from another server 2 which is located at an address indicated by address information included in the content storage server list thus received.

**[0340]** Each of the clients 4g and 4h, illustrated in Fig. 29, includes an acquisition location selecting section 38 in place of a relaying device selecting section 37 included in the client 4 of Embodiment 3. The acquisition location selecting section 38 has a function of the relaying device selecting section 37 in addition to a function of selecting a server 2 (described later in the present embodiment).

**[0341]** A response/request executing section 35 receives, from the server 2, as a response message, (i) the content storage server list and (ii) an instruction to transmit, to a server 2 located at an address indicated by ad-

dress information included in the content storage server list, a request message requesting the content.

**[0342]** In a case where the response/request executing section 35 receives the content storage server list and the instruction, the acquisition location selecting section 38 selects one of pieces of the address information included in the content storage server list received by the response/request executing section 35. The acquisition location selecting section 38 instructs the response/request executing section 35 to acquire the content from the server 2 located at the address indicated by the one of pieces of the address information thus selected.

**[0343]** Here, in a case where no priority is added to the pieces of the address information included in the content storage server list (in a case where there the acquisition location selecting section 38 has not received, from the server 2 to which the request for the content has been transmitted, an instruction as to an order in which the server 2 from which the content is acquired is selected from among the servers 2), the acquisition location selecting section 38 can (i) select a server 2 randomly from among the servers 2 indicated by the respective pieces of the address information included in the content storage server list, (ii) select a server 2 in accordance with a predetermined rule (default) from among the servers 2 indicated by the respective pieces of the address information, or (iii) select, from among the servers 2 indicated by the respective pieces of the address information, a server 2 which is closest to the client 4g or 4h in physical distance or network-structural distance.

**[0344]** Meanwhile, in a case where priorities are added to the pieces of the address information included in the content storage server list, the acquisition location selecting section 38 selects one of the pieces of the address information, which one of the pieces of the address information has the highest priority.

**[0345]** Further, in a case where the acquisition location selecting section 38 receives, from the a client status determining section 36, delay information indicating that a speed (receiving speed of the content) at which the response/request executing section 35 acquires the content from the server 2 selected by the acquisition location selecting section 38 is slower than a predetermined receiving speed, the acquisition location selecting section 38 (i) switches the server 2 from which the content is acquired to another server 2 located at an address indicated by another one of the pieces of the address information included in the content storage server list, and (ii) instructs the response/request executing section 35 to acquire the content from the another server 2.

**[0346]** Here, in a case where no priority is added to the address information included in the content storage server list (in a case where the acquisition location selecting section 38 has not received, from the server 2 to which the request for the content has been transmitted, an instruction as to an order in which the server 2 from which the content is acquired is selected from among the servers 2), the acquisition location selecting section 38 can

(i) select the another server 2 randomly from among the servers 2 indicated by the respective pieces of the address information included in the content storage server list, (ii) select the another server 2 in accordance with a predetermined rule (default) from among the servers 2 indicated by the respective pieces of the address information, or (iii) select, from among the servers 2 indicated by the respective pieces of the address information, the another server 2 which is second-closest to the client 4g or 4h in physical distance or in network structural distance.

**[0347]** Meanwhile, in a case where priorities are added to the respective pieces of the address information included in the content storage server list, the acquisition location selecting section 38 selects one of the pieces of the address information, which one of the pieces of the address information has the second highest priority.

**[0348]** The client status determining section 36 detects an occurrence of an event described below, in addition to operations described in Embodiment 3. Specifically, in a case where acquisition of the content from a server 2, executed by the content playing device, is delayed, the client status determining section 36 detects such an event that the content has been received behind schedule. This event is regarded as an event indicating (i) how good (or bad) a communication condition of a network between the content playing device and the server 2 is, and/or (ii) how large a load of the server 2 is. In a case where the client status determining section 36 detects an event that the content has been received behind schedule, the client status determining section 36 transmits, to the acquisition location selecting section 38, delay information indicating the event thus detected.

**[0349]** The example described above deals with the case where a response message includes a content storage section sever list for selecting another server 2 having the content thus requested. However, the present invention is not limited to this. It is possible that a response message received from the server 2 includes, in addition to a content storage server list, a content storage location list for selecting a relaying device. In a case where the response message received from the server 2 includes both the content storage server list and the content storage location list, the acquisition location selecting section 38 (i) selects whether the content is acquired via a relaying device or the content is acquired from another server 2, and (ii) selects one of pieces of address information, included in one of the content storage server list and the content storage location list, thus selected.

**[0350]** Here, in a case where no priority is added to the pieces of address information included in one of the content storage location list and the content storage server list, thus selected (in a case where the acquisition location selecting section 38 has not received, from the server 2, an instruction as to an order in which the server 2 from which the content is acquired is selected from among the servers 2), the acquisition location selecting section 38

can (i) select a server 2 randomly from among the servers 2 indicated by the respective pieces of the address information included in the one of the content storage location list and the content storage server list, (ii) select a server 2 in accordance with a predetermined rule (default) from among the servers 2 indicated by the respective pieces of the address information, or (iii) select, from among the servers indicated by the respective pieces of the address information, a server 2 which is closest to the client in physical distance or in network-structural distance.

**[0351]** Meanwhile, in a case where priorities are added to the respective pieces of the address information, the acquisition location selecting section 38 selects one of the pieces of the address information, which one of the pieces of the address information has the highest priority.

**[0352]** Further, in a case where the acquisition location selecting section 38 receives, from the client status determining section 36, delay information indicating that a receiving speed (content receiving speed) at which the response/request executing section 35 acquires the content from the device (a relaying device or a server 2) selected by the acquisition location selecting section 38 is slower than a predetermined receiving speed, the acquisition location selecting section 38 (i) switches the device from which the content is acquired to another relaying device or another server 2, located at an address indicated by one of the pieces of the address information included in the content storage location list or in the content storage server list, and (ii) instructs the response/request executing section 35 to acquire the content from the another relaying device or the another server 2.

Process carried out by each device]

**[0353]** Next, the following description deals with a process carried out by the server 2 and a process carried out by the client 4 serving as the content playing device, with reference to Figs. 31 and 32. Since a process carried out by the proxy 3 of Embodiment 4 is identical to a process of Embodiment 3, and a process carried out by the client 4 serving as a relaying device of Embodiment 4 is identical to a process of Embodiment 4, explanations of these are omitted here for the sake of simple explanation. Further, Embodiment 4 deals with an example in which one of the server 2 and the relaying device is selected, and a content is acquired from the one of the server 2 and the relaying device.

Process carried out by server 2]

**[0354]** First, the following description deals with the process carried out by the server 2 with reference to Fig. 31. Fig. 31 is a flowchart showing an example of the process carried out by the server 2. Note that a process identical to a process of Embodiment 3 has the same number as that of the process of Embodiment 3, and details of an explanation of the process are omitted here.

**[0355]** The response executing section 15 is ready to

receive a request message which requests transmission of a content. In a case where the request message which requests transmission of the content is received via a server communication section 11 (S601), the response executing section 15 checks a header of the request message thus received, so as to determine whether or not the request message thus received has been transmitted from a relaying device (S602).

**[0356]** In a case where the request message thus received has been transmitted from a content playing device (NO in S602), the response executing section 15 instructs the acquisition location specifying section 18 to specify a plurality of devices as a device from which the content thus requested is to be acquired by the content playing device.

**[0357]** On receipt of the instruction from the response executing section 15, the acquisition location specifying section 18 reads out content-storage-location information from the content-storage-location information storage section 13 (S603). The acquisition location specifying section 18 creates a content storage server list on the basis of the content-storage-location information (S701).

**[0358]** The acquisition location specifying section 18 determines whether or not the content storage server list has been created (S702). In a case where (i) the content-storage-location information could not be read out or the content-storage-location information stored in the content-storage-location information storage section 13 has been determined as being old information, and, as a result, (ii) the content storage server list has not been created (NO in S702), the acquisition location specifying section 18 instructs the response executing section 15 to update the content-storage-location information. The response executing section 15 thus instructed transmits, to each of other servers 2 connected to a network, an inquiry as to whether or not each of the other servers 2 has the content thus requested (S703).

**[0359]** The response executing section 15 notifies the acquisition location specifying section 18 of a result of a response received from each of the other servers 2. On the basis of the result of the response, the acquisition location specifying section 18 requests the content-storage-location information storage section 13 to update the content-storage-location information, so as to update the content-storage-location information (S704). Then, the acquisition location specifying section 13 creates a content storage server list again (S705).

**[0360]** The acquisition location specifying section 18 (i) refers to the content-storage-location information thus read out, (ii) identifies a plurality of URIs of devices (each being a relaying device or a server (a relaying device), from which the content is acquired) being associated with the content requested by the content playing device, and then, (iii) creates a content storage location list including a plurality of pieces of address information thus identified (S604).

**[0361]** The acquisition location specifying section 18

transmits the content storage location list thus created to the response executing section 15. The response executing section 15 transmits, to the content playing device, a response message which instructs the content playing device to acquire the content thus requested from a relaying device or a server 2, located at an address of one of pieces of address information included in the content storage location list and/or in the content storage server list, created by the acquisition location specifying section 18 (S706).

**[0362]** The response executing section 15 creates a transmission log on the basis of the response message transmitted to the content playing device, and adds the transmission log thus created to a transmission log storage section 12 (S610).

**[0363]** Since a process (S606 through S609) carried out under a condition that the request message thus received is transmitted from a relaying device is identical to a process (S506 through S509 in Fig. 14) of a server 2 of Embodiment 2, explanations of the process are omitted here for the sake of simple explanation.

Process carried out by client 4 serving as content playing device]

**[0364]** Next, the following description deals with a process carried out by the client 4 serving as the content playing device, with reference to Fig. 32. Fig. 32 is a flowchart showing an example of the process carried out by the client 4 serving as the content playing device. Note that a process identical to a process of Embodiment 3 has the same number as that of the process of Embodiment 3, and details of an explanation of the process are omitted here for the sake of simple explanation. Further, a process for receiving a content is identical to a process (S625 through S631 in Fig. 23) of Embodiment 3, and therefore is shown as "S724" in Fig. 32.

**[0365]** The response/request executing section 35 transmits, to a server 2, a request message which requests transmission of a content (S621). The response/request executing section 35 receives, as a response to the request message, a response message which includes (i) a content storage location list and/or a content storage server list, and (ii) an instruction to transmit the request message, which requests the transmission of the content, to a device (a relaying device or a server 2) located at an address indicated by one of pieces of address information included in the content storage location list and/or the content storage server list (S3721).

**[0366]** In a case where response/request executing section 35 receives the response message, the acquisition location selecting section 38 selects one of the pieces of the address information included in the content storage location list and/or the content storage server list, received by the response/request executing section 35 (S722). The acquisition location selecting section 38 instructs the response/request executing section 35 to ac-

quire the content from the device located at an address indicated by the one of the pieces of the address information thus selected.

**[0367]** On receipt of the instruction from the acquisition location selecting section 38, the response/request executing section 35 transmits the request message to the device selected by the acquisition location selecting section 38 (S723).

**[0368]** The device receives the request message and carries out a process of acquiring the content, which process is explained in Embodiment 3. Then, the response/request executing section 35 and the client status determining section 36 carry out a process of acquiring/playing media segments (S724).

**[0369]** Here, in a case where the response/request executing section 35 receives all movie fragments, the response/request executing section 35 checks whether or not all media segments of the content thus requested are received (S632). In a case where the response/request executing section 35 determines that all the media segments of the content are received (YES in S632), the response/request executing section 35 finishes the process. On the other hand, in a case where the response/request executing section 35 determines that there is any media segment which has not been received (NO in S632), the client status determining section 36 determines whether or not the device from which the content is acquired should be switched to another device, in the same manner as Embodiment 3 (S725).

**[0370]** In a case where it is determined that switching of the device to another device is not to be executed (NO in S725), the device from which the content is acquired is not switched to another device, and the response/request executing section 35 transmits, to the device selected in S722, a request message to which media segment numbers are incremented (S723).

**[0371]** In a case where it is determined that the switching of the device to another device is to be executed (YES in S725), the client status determining section 36 transmits, to the acquisition location selecting section 38, delay information indicating that acquisition of the content is delayed. On receipt of the delay information from the client status determining section 36, the acquisition location selecting section 38 selects another one (which is different from the one of the pieces of the address information selected above) of the pieces of the address information included in the content storage location list or the content storage server list (S726). The acquisition location selecting section 38 instructs the response/request executing section 35 to acquire the content from another device located at an address indicated by the another one of the pieces of the address information thus selected.

**[0372]** On receipt of the instruction from the acquisition location selecting section 38, the response/request executing section 35 transmits again, to the another device selected by the acquisition location selecting section 38, the request message to which the media segment num-

bers are incremented (S723).

**[0373]** Note that, it is possible to execute, in Embodiment 4, switching with use of not a media segment unit but a movie fragment unit, in the same manner as Embodiment 3. In addition, in this case, it is possible to carry out a switching process in the same manner as Embodiment 3.

Example 4]

**[0374]** The following description further deals details of Embodiment 4 more specifically with use of Example 4 shown in Fig. 33. Example 4 shows an example of an operation of the content distribution system 1c which instructs a content playing device serving, which is the client 4g, to acquire a content from one of pieces of address information included in a content storage location list or a content storage server list. Fig. 33 is a view showing an example of an operation sequence of the content distribution system 1c of Example 4. Note that a process identical to a process of Embodiment 3 has the same number as that of the process shown in Fig. 24.

**[0375]** Example 4 is made on a premise that a content 1 having a format shown in Fig. 21 is stored in each of content storage sections 5a, 5b, and 5c, and the content storage sections 5a, 5b, and 5c are identical to each other in how to divide the content 1 into media segments. Further, the content 1 has been already cached in (i) a cache storage section 6a, (ii) a cache storage section 6c, and (iii) a client storage section 8h.

**[0376]** Further, a server 2a creates a content storage location list in such a manner that priorities are added to a plurality of pieces of address information, on the basis of times and dates, included in content-storage-location information. Furthermore, in Example 4, when a process shown in Fig. 32 is started, (i) transmission logs 51 through 53 shown in Fig. 25 have been already stored in a transmission log storage section 12, and (ii) pieces (61 through 63) of content-storage-location information shown in Fig. 26 have been already stored in a content-storage-location information storage section 13.

**[0377]** Moreover, one session is defined as a sequence starting from a time at which the content playing device transmits a request message and ending a time at which the content playing device receives a response message in response to the request message thus transmitted.

**[0378]** As shown in Fig. 33, in session 810, the client 4g transmits, to the server 2a, a request message which requests transmission of the content 1 (request 811). The server 2a receives the request 811, and the response executing section 15 instructs the acquisition location specifying section 18 to specify a plurality of devices as a device from which the content 1 is acquired. On receipt of the instruction, the acquisition location specifying section 18 checks content-storage-location information stored in the content-storage-location information storage section 13 (process 812). The acquisition location

specifying section 18 checks whether or not there is a content storage server list on the basis of the content-storage-location information. In a case where there is no content storage server list, the acquisition location specifying section 18 checks (i) whether or not a server 2b has the content 1 and (ii) whether or not a server 2c has the content 1 (process 813). Here, the acquisition location specifying section 18 can execute such checking by (i) transmitting a request for a header of the content 1 to each of the servers 2, and (ii) checking a response received from each of the servers 2, for example (a request 814 transmitted to the server 2c and a response 816 received from the server 2c, in Fig. 33 and a request 815 transmitted to the server 2c and a response 817 received from the server 2c in Fig. 33).

**[0379]** Here, the content-storage-location information storage section 13 indicates that a proxy 3a, a proxy 3c, and a client 4f are associated with the content 1, i.e., the content 1 is stored in the proxy 3a, the proxy 3c, and the client 4f. Accordingly, on the basis of the times and dates, included in the content-storage-location information, the acquisition location specifying section 18 adds a first priority, a second priority, and a third priority to the proxy 3a, the client 4f, and the proxy 3c, respectively, so as to create a content storage location list which includes address information of the proxy 3a, the client 4f, and the proxy 3c, and to which the priorities are added.

**[0380]** In addition, on the basis of responses in process 813 (the responses 815 and 817), the acquisition location specifying section 18 creates a content storage server list. According to the present example, a response from the server 2b has been received earlier than a response from the server 2c, so that the server 2b has a first priority and the server 2c has a second priority. The acquisition location specifying section 18 creates the content storage server list which includes address information of the servers 2b and 2c, and also address information of the server 2a itself, and to which the priorities are added (process 818).

**[0381]** Here, how to set a priority of the server 2a can be determined arbitrarily. For example, it is possible to cause the server 2a to have the highest priority for all cases (a higher priority than those of the other servers 2). Further, it is possible to have such a setting that (i) in a case where a response speed of the server 2b or a response speed of the server 2c is faster than a certain threshold, the server 2a has a lower priority than that of the server 2b or 2c, and (ii) the response speed of the server 2b or the response speed of the server 2c is slower than the certain threshold, the server 2a has a higher priority than that of the server 2b or 2c. Furthermore, it is possible to have such a setting that, in a case where a process load of the server 2a is larger than a predetermined threshold, the server 2a has a lower priority than those of the other servers 2.

**[0382]** The response executing section 15 selects one (having the highest priority) of the pieces of the address information included in the content storage location list

or the content storage server list, created by the acquisition location specifying section 18, in accordance with the priorities thus set. Then, the response executing section 15 transmits, to the client 4g, a response message for acquiring the content thus requested from a relaying device or a server 2 located at an address indicated by the one of the pieces of the address information thus selected (response 819).

**[0383]** Here, the response executing section 15 determines which one of the content storage location list and the content storage server list is preferentially used to selecting one of the pieces of the address information. That is, the response executing section 15 determines whether the client 4g acquires the content from a relaying device or from a server 2.

**[0384]** How to select preferentially one of the content storage location list and the content storage server list can be arbitrarily determined. For example, the response executing section 15 can select the content storage location list (or the content storage server list) preferentially on the basis of a setting of a default. Then, in a case where the content storage location list (or the content storage server list) thus selected cannot be used, the response executing section 15 selects the content storage server list (or the content storage location list), for example.

**[0385]** Further, the response executing section 15 can execute the selection in such a manner that (i) in a case where a time and date of the content-storage-location information including one (having the highest priority) of the pieces of the address information, included in the content storage location list, is a recent time and date (within a predetermined time period), the content storage location list is selected preferentially, and (ii) in a case where the time and date is an old time and date (before the predetermined time period), the content storage server list is selected preferentially. Furthermore, the response executing section 15 can execute the selection in such a manner that, in a case where, under a condition that there is no content-storage-location information, one of the pieces of the address information included in the content storage location list is determined (i) in accordance with a default or (ii) randomly, the content storage server list is selected preferentially.

**[0386]** In the present example, in a case where there is the content storage location list, the response executing section 15 selects the content storage location list preferentially.

**[0387]** Note that, in the present example, acquisition of the content from the server 2a is basically the same as acquisition of the content with use of a relaying device (that is, if the content is acquired with use of the relaying device the content, an address of the device from which the content is acquired is identical to an address of the server 2a). In other words, substantially the same process (the same process as a process of Embodiment 3) is carried out for both (i) a case where the response executing section 15 selects the content storage location

list, and (ii) a case where the response executing section 15 selects the content storage server list, and then selects the server 2a from the content storage server list. Moreover, in a case where (i) the response executing section 15 selects the content storage server list, and then selects the server 2b or the server c from the content storage server list, and (ii) the server b or the server c thus selected receives the request for the content from the client 4g, the server b or the server c identifies a predetermined relaying device and instructs the client 4g to acquire the content from the relaying device thus identified, in the same manner as Embodiment 3.

**[0388]** That is, in a case where the relaying device is used ((i) in a case where the content storage location list is selected, and (ii) in a case where the content storage server list is selected and then the server 2a is selected from the content storage server list), the response executing section 15 first instructs the client 4g to acquire the content from the proxy 3a. If it is impossible to acquire the content from the proxy 3a, or a speed at which the content is acquired from the proxy 3a speed is slow, then, the response executing section 15 instructs the client 4g to execute acquisition of the content from the client 4f (if impossible, then acquisition of the content from the proxy 3c). On the other hand, in a case where another server 2 is used, the response executing section 15 instructs the client 4g to execute the acquisition of the content from the server 2b (if impossible, then acquisition of the content from the server 2c).

**[0389]** Then, the response executing section 15 creates a transmission log on the basis of the response message thus transmitted, and adds the transmission log thus created to the transmission log storage section 12 (process 820).

**[0390]** On receipt of the response 819, the acquisition location selecting section 38 of the client 4g selects the proxy 3a having the highest priority, as a candidate for the device from which the content is acquired, on the basis of the instruction received from the server 2a (process 830). Then, the acquisition location selecting section 38 instructs the response/request executing section 35 to acquire the content 1 from the proxy 3a.

**[0391]** In a case where the proxy 3a or another client (client 4f) is selected as the device from which the content is acquired, a process (session 840, process 850, process 860, and session 870) is the same as a process (session 330, process 340, process 350, and session 360 shown in Fig. 24) of Embodiment 3, and therefore explanations of these are omitted here for the sake of simple explanation.

**[0392]** Note that, in a case where a server 2 is selected, the content is acquired from the server 2 in the same manner as a general acquisition process with use of HTTP.

**[0393]** Further, in Example 3, the client status determining section 36 notifies the relaying device specifying section 16 of the delay information per movie fragment, whereas, in Example 4, the client status determining sec-

tion 36 notifies the acquisition location specifying section 18 of the delay information. On receipt of such a notification, the acquisition location specifying section 18 executes again selection of the device from which the content is acquired, in the same manner as the relaying device specifying section 16.

**[0394]** As described above, in Embodiment 4, a content playing device receives, from a server 2, (i) a content storage location list including address information of a plurality of relaying devices and (ii) a content storage server list including address information of a plurality of servers. Then, the content playing device acquires a content from (i) a relaying device located at an address indicated by one of pieces of the address information included in the content storage location list or (ii) a server 2 located at an address indicated by one of pieces of the address information included in the content storage server list. Then, in a case where there is a delay in acquisition of the content, the content playing device switches the device from which the content is acquired to (i) another relaying device located at an address indicated by another one of the pieces of the address information included in the content storage location list or (ii) another server 2 located at an address indicated by another one of the pieces of the address information included in the content storage server list. Accordingly, it becomes possible to distribute a load of a network (particularly, a network between the content playing device and a relaying device, and a network between the content playing device and a server 2) efficiently even in terms of time. It becomes therefore possible for the content distribution system 1c to execute control more finely. As a result, it becomes possible to maintain higher service quality for a larger number of content playing devices.

**[0395]** Further, Embodiment 4 shows the example in which the content playing device receives, from the server 2a, (i) the content storage location list including the address information of the plurality of relaying devices and (ii) the content storage server list including the address information of the plurality of servers. Note, however, that the content playing device can receive, from the server 2a, only the content storage server list including the address information of the plurality of servers. Similarly, the server 2a can notify the content playing device of only the content storage server list including address information of other servers 2 each having the content thus requested.

HTTP message in Example 4]

**[0396]** Next, the following description deals with details of the response 819 used in the operation sequence shown in Fig. 33. Note that request 811 and other requests, and responses are the same as those in Example 3, and therefore explanations of these are omitted here for the sake of simple explanation.



HTTP message in response 819]

**[0397]** The following description deals with an HTTP message in response 819 with reference to Fig. 34.

(Response R3210 instructing which one of devices content is acquired from)

**[0398]** (a) of Fig. 34 shows an HTTP message corresponding to the response 819, which (i) is transmitted from the server 2a to the client 4g and (ii) instructs the client 4g which one of the devices the client 4g acquires the content from. As shown in (a) of Fig. 34, the HTTP message includes a response line and a header.

**[0399]** The response line shown in (a) of Fig. 34 is such that information instructing the use of a relaying device is described in a format of "a status number and a message".

**[0400]** Further, the header shown in (a) of Fig. 34 includes a "Location" header which designates the relaying device to be used. In other words, the HTTP message shown in Fig. 34 is a response which instructs to request the content 1 with use of the relaying device designated by the following "Location" header. In the example shown in Fig. 34, address information "http://example-proxy1.com", indicating an address of the proxy 3a, is described in the "Location" header. With the arrangement, the client 4g, which has received the response, can obtain the address information of the relaying device (proxy 3a) to which the client 4g transmits the request for the content 1.

**[0401]** Further, the header shown in (a) of Fig. 34 includes an "X-Alternative-Proxy-List" header indicating address information of other relaying devices, each of which possesses the content 1. In the "X-Alternative-Proxy-List" header, the address information "http://example-client2.com, http://example-proxy3.com", indicating addresses of other relaying devices (here, a client 4h and a proxy 3c), are described. With the arrangement, the relaying devices each (possibly) possessing the content 1, other than the relaying device designated by the "Location" header, are presented to the client 4g. As a result, the client 4g can select, as the relaying device from which the content 1 is acquired, one of the relaying devices located at the addresses indicated by (i) the address information included in the "Location" header and (ii) the address information included in the "X-Alternative-Proxy-List" header. Note that, "X" in a title of the header indicates that the header is newly defined in Embodiment 4.

**[0402]** Further, the header shown in (a) of Fig. 34 includes an "X-Alternative-Server-List" header indicating address information of other servers, each of which possesses the content 1. In the "X-Alternative-Server-List" header, address information "http://svr2.example.com, http://svr3.example.com" indicating addresses of other servers (here, the server 2b and the server 2c), each of which (possibly) possesses the content 1, is described. With the arrangement, the client 4g, which has received

the response, can not only execute such selection that the content is acquired via the relaying device included in the "Location" header or in the "X-Alternative-Proxy-List" header but also execute such selection that the content 1 is acquired from the server 2 described in the "X-Alternative-Server-List".

**[0403]** Note that, "X" in a title of the header shows that the header is newly defined in Embodiment 4.

**[0404]** The HTTP message shown in (a) of Fig. 34 is such an HTTP message that the server 2a requests the client 4g to access the content with use of the relaying device. Meanwhile, (b) of Fig. 34 shows an example of a message instructing the client 4g to access another server 2 without using any relaying device.

**[0405]** In (b) of Fig. 34, information which instructs the client 4g to make an access with use of another URI is described in a format of "status number (space) message".

**[0406]** Further, the header includes a "Location" header indicating another URI. The message shown in (b) of Fig. 34 instructs the client 4g to request the content 1 with use of the URI.

**[0407]** Furthermore, in the same manner as (a) of Fig. 34, the header includes an "X-Alternative-Server-List" header, in which other servers 2 which can be used are described.

<Embodiment 5>

**[0408]** Embodiment 4 deals with the example in which the content playing device is notified of, with use of the HTTP message, information on each of the servers 2, from which the content can be acquired.

**[0409]** Embodiment 5 of the present invention deals with an example in which information on servers 2, each of which can supply a content, is notified with use of meta data related to the content.

**[0410]** In Embodiment 5, the meta data of the content is described with use of a markup language MPD (Media Presentation Description) proposed in DASH (Dynamic Adaptive Streaming over HTTP) with which standardization has been currently executed. The MPD is meta data related to a moving image content, and is such that information, such as an address of a media segment and a video bit rate of a media segment, is defined for each of predetermined time periods. In Embodiment 5, meta data of a content is referred to as "MPD data".

**[0411]** An arrangement of Embodiment 5 is identical to that of Embodiment 4 illustrated in Fig. 29, and therefore is explained below with reference to Fig. 29. More specifically, a server 2a illustrated in Fig. 29 prepares MPD data which is meta data of the content. In the MPD data, not only information related to a moving image content, such as an encoding method and a bit rate, but also address information of the servers 2, each of which can supply the content, and address information used to acquire a media segment are described. Before playing the content, a client 4g acquires and analyzes the MPD data,

so as to select one of a plurality of servers 2 described in the MPD data.

**[0412]** In addition, the MPD data, which is the meta data of the content, employs a format with which an external resource can be referred to. By taking advantage of the format, even if a condition of a network or a condition of a server changes during a time period from a time that the MPD data is created to a time that the content is actually played with use of the MPD data, it is possible to (i) reflect such a change and therefore (ii) distribute a load. Further, by setting timing at which the external resource is referred to so that the external resource is referred to at short intervals, it becomes possible to (i) reflect changes in condition more finely and therefore to (ii) execute control more finely.

Outline of content distribution system 1c]

**[0413]** The arrangement of Embodiment 5 is identical to the arrangement of Embodiment 4 of the subject application, illustrated in Fig. 29. Functionally, (i) the server 2a of Embodiment 5 is different from the server 2a of Embodiment 4 in that the server 2a of Embodiment 5 prepares MPD data which is meta data of the content supplied from the server 2a, and (ii) the client 4g of Embodiment 5 is different from the client 4g of Embodiment 4 in that the client 4g of Embodiment 5 acquires, from the MPD data, information of a server to which the client 4g make an access, to play the content.

**[0414]** Specifically, a response executing section (managing means) 15 manages (i) a content and (ii) meta data (MPD data which is meta data of the content) including (a) content-storage-location information in which content identification information for specifying the content and addresses of other content distributing devices, each having the content, are associated with each other, or (b) a storage location address (external resource) indicating a location of the content-storage-location information.

**[0415]** In a case where the content is stored in a content storage section 5, the response executing section 15 basically creates MPD data of the content, and, if necessary, updates the MPD data thus created. Further, the response executing section 15 creates an external resource on receipt of a request for creation of the external resource.

**[0416]** Moreover, the response executing section (request determining means) 15 determines whether the request described above is a request for the content or a request for the meta data.

**[0417]** Further, in a case where the response executing section 15 determines that the request is the request for the content, the response executing section 15 transmits the content thus requested to a device which is a source of the request. On the other hand, in a case where the response executing section 15 determines that the request is the request for the meta data, the response executing section 15 transmits the meta data thus request-

ed to the device which is the source of the request.

**[0418]** The response executing section 15 can determine which one of the request for the content, the request for the meta data, and a request for the content-storage-location information with use of a storage location address, the request described above is.

**[0419]** In a case where the response executing section 15 determines that the request is the request for the content-storage-location information with use of the storage location address (external resource), the response executing section 15 transmits, to the device which is the source of the request, the content-storage-location information whose location is indicated by the storage location address.

**[0420]** Further, the acquisition location specifying section 18 (i) transmits, to each of the predetermined other content distributing devices, an inquiry as to whether or not each of the predetermined other content distributing devices has a predetermined content, and (ii) acquires addresses of other content distributing devices, each making, in response to the inquiry, a response that the content distributing device has the predetermined content, among the predetermined other content distributing devices.

**[0421]** Furthermore, the acquisition location specifying section 18 (i) creates content-storage-location information by causing the addresses of other content distributing devices each having the predetermined content and content identification information for specifying the predetermined content to be associated with each other, and (ii) stores the content-storage-location information thus created in a content-storage-location information storage section (storage section) 13.

**[0422]** Moreover, the acquisition location specifying section (update determining means) 18 determines whether to update the content-storage-location information stored in the content-storage-location information storage section 13.

**[0423]** Further, in a case where (i) the response executing section 15 determines that the request is the request for the content-storage-location information with use of the storage location address, and (ii) the acquisition location specifying section 18 determines that it is necessary to update the content-storage-location information whose location is indicated by the storage location address, the acquisition location specifying section 18 (i) makes the inquiry described above, (ii) acquires the addresses described above, and (iii) creates the content-storage-location information on the basis of the addresses thus acquired. Then, the response executing section 15 transmits the content-storage-location information to the device which is the source of the request.

**[0424]** Furthermore, in a case where (i) the response executing section 15 determines that the request is the request for the content-storage-location information with use of the storage location address, and (ii) the acquisition location specifying section 18 determines that it is unnecessary to update the content-storage-location in-

formation whose location is indicated by the storage location address, the response executing section 15 transmits, to the device which is the source of the request, the content-storage-location information whose location is indicated by the storage location address.

**[0425]** Moreover, the meta data can include a plurality of storage location addresses each indicative of a location of content-storage-location information which is set for each of units into which the content is divided at predetermined time intervals.

**[0426]** Further, the content can include a plurality of media segments, and each of the units of the content, into which the content is divided at the predetermined time intervals, can include at least one media segment.

**[0427]** Furthermore, the meta data can include (i) a content storage server list including a plurality of pieces of the content-storage-location information or (ii) a storage location address indicating a location of the content storage server list.

**[0428]** Moreover, the client 4 serving as the content playing device (i) transmits the request for the meta data to the server 2, (ii) receives the meta data as a response to the request for the meta data, and (iii) acquires the content in accordance with the meta data thus received.

**[0429]** Further, in a case where (i) a response/request executing section (content acquiring means) 35 receives the content storage server list included in the meta data thus received, or (ii) the response/request executing section (content acquiring means) 35 transmits the request for the content-storage-location information with use of the storage location address which is included in the meta data thus received, and, as a response to the request, receives the content storage server list, the response/request executing section 35 acquires the content from another content distributing device located at an address indicated by one of the pieces of the content-storage-location information included in the content storage server list thus received.

**[0430]** Moreover, in a case where a receiving speed at which the response/request executing section 35 receives the content is slower than a predetermined receiving speed, the acquisition location selecting section (acquisition location changing means) 38 switches the server 2 (e.g., the server 2b) from which the content is acquired to another server 2 (e.g., the server 2c) located at another address included in the content storage server list.

Content meta data: MPD data]

**[0431]** Each of Figs. 35, 36, and 38 shows an example of how the MPD data, which is the meta data of the content used in Embodiment 5, is described. Fig. 35 is an example in which an external resource is not referred to. The content is fragmented by a predetermined unit, and is, for transmission, media-segmented, in the same manner as Embodiment 3. In (a) of Fig. 35, "content1/0.mp4", "content1/1.mp4", and the like indicate media segments

of the content 1, for example. In the example shown in (a) of Fig. 35, the content 1 is divided into 12 media segments.

**[0432]** The MPD data is data of a markup language format, and employs "MPD" as a root element. A value of an attribute "minBufferTime" of an MPD start tag indicates an initial buffering time period which is necessary to play a video smoothly. A value of an attribute "type" indicates a default value of an attribute "type" of a "Representation" tag (described later). That is, a value of the attribute "type" indicates whether a representation whose attribute "type" is not designated in the "Representation" tag is on-demand streaming delivery or live streaming delivery. Further, an attribute "mediaPresentationDuration" indicates a playing time period of the content. In the present example, the playing time period of the content is described as being 120 seconds.

**[0433]** "Period", which is a sub-element of "MPD", indicates that information related to a video to be played within a certain time period (period) is described in a range between a corresponding Period start tag and a corresponding Period end tag. An attribute "id" of the Period start tag is information for specifying each Period included in the content provided with use of the MPD, and a unique value is set to each Period.

**[0434]** "Group", which is a sub-element of "Period", indicates that at least one sub-element "Representation" described in the range between a Group start tag and a Group end tag belongs to the same representation group.

**[0435]** That is, "Group" indicates that only one representation is selected, and media segments (target data to be played) of the only one representation are played in a corresponding time period. Note that, representations belonging to the same group might be different from each other in play quality such as an image size, a frame rate, and a bit rate, but are identical to each other in the content to be played. For example, in the example shown in (b) of Fig. 35, two representations (the content 1 and the content 2) are described. In this case, it is possible to play the content by selecting either one of the two representations.

**[0436]** Further, in (a) of Fig. 35, an attribute "mimeType" of the Group start tag indicates, for example, a sort of codec used in media segments constituting the representation. Furthermore, an attribute "lang" indicates a language of the representation belonging to the Group.

**[0437]** Moreover, in the range between the Group start tag and the Group end tag, a sub-element "SegmentInfoDefault" is described. The "SegmentInfoDefault" is such that common information, which is shared by all the representations in the range between the Group start tag and the Group end tag, is described. In the present example, the "SegmentInfoDefault" element further includes, as a sub-element, a "BaseURL" element. In a range between a Base URL start tag and a Base URL end tag, a common URL is described. With use of such a URL and the following URL information of the representations, it is possible to determine a device to be re-

ferred. As shown in (a) of Fig. 35, it is possible to describe a plurality of Base URLs.

**[0438]** The representations constituting the Group are described with use of "Representation" tag. An attribute "bandwidth" of a Representation start tag, shown in (a) of Fig. 35, indicates a bit rate of the representation.

**[0439]** In a range between a Representation start tag and a Representation end tag, a sub-element "Segment" is used to indicate that there is media segment information. An URL from which a media segment belonging to the representation is acquired is described with use of an attribute "sourceURL" of a start tag of a sub-element "Url" of the Segment tag. These Urls are described for corresponding media segments. Note that, in a case where there is a common part between these Urls, it is possible to describe the Urls with use of the BaseURL tag described above.

**[0440]** In the example shown in Fig. 35, the BaseURL tag is used, and a Url of each media segment has no description indicating a host. Accordingly, a Url of a media segment is created by using information indicated by the BaseURL tag. That is, an access to a first media segment is made with use of a Url created as "http://srv2.example.com/content1/0.mp4" which is obtained with use of (i) the BaseURL tag "http://srv2.example.com/" and (ii) the Ur1 tag "content1/0.mp4".

**[0441]** As described above, in a case where a client acquires each media segment, a Url of each media segment is created and acquired, on the basis of an analysis result of MPD data.

**[0442]** Next, the following description deals with how to refer to an external resource with use of the MPD data, with reference to Figs. 36 through 38.

**[0443]** Details of the present example are explained with use of the example shown in (a) of Fig. 35, which example employs one representation.

**[0444]** As described above, in the MPD data shown in (a) of Fig. 35, address information of a server in which a corresponding media segment is stored is described with use of the BaseURL tag. Here, there are a plurality of BasURL tags. That is, the client can select one of the plurality of Base URL tags depending on a condition, so as to acquire the media segment under an optimum condition.

**[0445]** However, generally, the MPD data is created when the content is stored in the server2. For this reason, even if information on an optimum server is collected and described at a time that the MPD data is created, it is highly possible that a network status or information on such an optimum server might have been changed at a time that the content is actually accessed with use of the MPD data. Further, even if, for example, a server which works at a higher speed than the above server is added to deliver the content after the MPD data is created, it is impossible to use such a high-speed server unless the MPD data thus created is recreated.

**[0446]** In view of this, a function of a link to an external resource of the MPD is used. Fig. 36 shows an example

of such an MPD data. In Fig. 36, a description of "xlink" is used as the attribute of the Group start tag, in place of the server information (information described with use of the BaseURL tag) described with use of the Group tag in (a) of Fig. 35, the description of each representation, the description of each of the media segments (information described with use of the Representation tag) constituting the representation, and the like. The xlink is a function of referring to an external resource. In a case where data including the description of the xlink is analyzed, it is possible to execute the analysis by acquiring and taking in the external resource linked by the xlink. As shown in Fig. 36, a URL of an external resource linked by an attribute "xlink:href" is described. The attribute "xlink:actuate" is such that at what stage the external resource indicated by "xlink:href" is acquired is described. The "xlink:actuate" is classified into "onRequest", with which the external resource is acquired if necessary, and "onLoad" with which the external resource is acquired at the same time as acquisition of the MPD data. In the present example, the "onRequest", with which the external resource is acquired if necessary, is used.

**[0447]** Fig. 37 is a view showing an example of data of an external resource (http://example.com/content1/resource 1.xml). The MPD data shown in Fig. 36 takes in the external resource shown in Fig. 37 with use of the xlink, and becomes MPD data which is identical to the MPD data shown in (a) of Fig. 35.

**[0448]** Further, in the present example, in order to execute control more finely, the MPD data is divided into short Periods with use of the Period tag described above, and each of the Periods takes in the external resource with use of the xlink. The MPD data shown in (a) of Fig. 35 and the external resource shown in Fig. 37 are such that the content is described with use of one Period. Accordingly, even if the external resource is taken in, it is merely possible to reflect a condition obtained at a time that the content is started to be played. That is, in a case where the content is a long-time content, there might be a case where, even if a certain server is selected as the optimum server at the time that the content is started to be played, the certain server thus selected might not be the optimum server anymore during a time period in which the content is played, due to a change in a condition of the network or a change in a condition of the certain server or conditions of other servers. Moreover, in a case where a server which is the most appropriate server at a final phase of acquisition of the content (i) has not been selected at a time that the content is started to be played and (ii) has not been described as the external resource, it is impossible to select the server.

**[0449]** In view of this, (a) of Fig. 38 shows an example in which the MPD data is divided into a plurality of Periods, and an external resource is taken in with use of the xlink in each of the plurality of Periods. (b) through (d) of Fig. 38 show examples of the external resource thus taken in.

**[0450]** Each of the external resources shown in (b)

through (d) of Fig. 38 has four media segments. In the present example, one media segment equals 10 seconds. That is, one Period in the MPD data shown in (a) of Fig. 38 equals 40 seconds, and an external resource is taken in per acquisition of 40-second data.

**[0451]** With the arrangement in which the server information included in the MPD data is provided as an external resource, it is possible to create and provide an external resource in response to a request received from the client, which external resource (i) has not been created in advance and (ii) reflects a condition of a distribution system or a network at a time that the client makes a request.

**[0452]** Further, in the present example, by setting a Period to be short, it becomes possible to reflect the condition of the network or the conditions of the servers finely. With the arrangement, it is possible to provide information in accordance with a condition at a time that MPD data is used, even if timing that the MPD data, which is meta data of a content, is created, and timing that the MPD data is used, are different from each other.

Process carried out by each device]

**[0453]** Next, the following description deals with processes carried out by the server 2 and the client 4, with reference to Figs. 39 and 40. A process carried out by proxies 3a, 3b, and 3c of Embodiment 5 is identical to a process of Embodiment 3, and therefore an explanation of the process is omitted here for the sake of simple explanation.

Process carried out by server 2]

**[0454]** First, the following description deals with a process carried out by the server 2a in accordance with Embodiment 5, with reference to Fig. 39. Fig. 39 is a flowchart showing an example of the process carried out by the server 2a. A process which is identical to a process of Embodiment 4 of the present invention has the same sign as that of the process of Embodiment 4.

**[0455]** The response executing section 15 is ready to receive a request message from the client 4. The response executing section 15 receives the request message from the client 4 via the server communication section 11 (S901).

**[0456]** On receipt of the request message, the server 2 determines which one of a request for a content, a request for xlink data, and a request for MPD data (which is meta data of the content) the request message thus received is (S902, S903, S904). In a case where it is determined that the request message is the request for the meta data of the content (YES in S904), the server 2 reads out designated MPD data from the content storage section 5 (S905), and transmits the MPD data to a device which is a source of the request (S906).

**[0457]** In a case where it is determined that the request message is the request for acquisition of external re-

source data with use of the xlink (YES in S903), the response executing section 15 instructs the acquisition location specifying section 18 to specify a plurality of servers as a server from which the content is acquired by the content playing device. On receipt of such an instruction from the response executing section 15, the acquisition location specifying section 18 reads out content-storage-location information from the content-storage-location information storage section 13 (S603).

**[0458]** The acquisition location specifying section 18 creates a content storage server list on the basis of the content-storage-location information (S701). The acquisition location specifying section 18 determines whether or not the content storage server list could be created (S702). In a case where (i) the content-storage-location information could not be read out or it was determined that the content-storage-location information stored in the content-storage-location information storage section 13 was old information, for example, and, as a result, (ii) the content storage server list could not be created, the acquisition location specifying section 18 instructs the response executing section 15 to update the content-storage-location information. On receipt of the instruction to update the content-storage-location information, the response executing section 15 transmits, to each of other servers 2 connected to the network, an inquiry as to whether or not each of the other servers 2 has the content thus requested (S703).

**[0459]** The response executing section 15 notifies the acquisition location specifying section 18 of a result of a response received from each of the other servers 2. On the basis of the result of the response thus notified, the acquisition location specifying section 18 requests the content-storage-location information storage section 13 to update the content-storage-location information, so that the content-storage-location information storage section 13 updates the content-storage-location information (S704). Then, the acquisition location specifying section 18 creates the content storage server list again (S705).

**[0460]** The response executing section 15 creates external resource data which instructs the client 4 to acquire the content thus requested from one of the servers 2 each being located at an address indicated by a corresponding one of pieces of address information included in the content storage server list created by the acquisition location specifying section 18 (S907). Then, the response executing section 15 transmits the external resource data to the content playing device (S908).

**[0461]** Meanwhile, in a case where it is determined that the request is the request for the content (YES in S902), the server 2 transmits the content to the source of the request. A process carried out here is identical to a process of a server 2 in accordance with Embodiment 3 (S606 through S610 shown in Fig. 22), and therefore an explanation of the process is omitted here for the sake of simple explanation.

**[0462]** In a case where the request is not the request

for the content, the request for the xlink data, or the request for the MPD data (NO in S904), the server 2 carries out a process corresponding to the request thus received. For example, in a case where the server 2 receives a GET request for data of a web page including a link to the content, or a HEAD request for a file size of the content, a time stamp of a file, or the like, the server 2 carries out a process corresponding to such a request.

Process carried out by client 4 serving as content playing device]

**[0463]** Next, the following description deals with a process carried out by the client 4 serving as the content playing device, with reference to Fig. 40. Fig. 40 is a flowchart showing an example of the process carried out by the client 4 serving as the content playing device.

**[0464]** The response/request executing section 35 transmits, to the server 2, a request message which requests transmission of MPD data corresponding to a content (S921). The response/request executing section 35 receives, as a response to the request message, a response message including the MPD data (S922). The meta data thus received is analyzed by the response/request executing section 35 so that a content to be played is determined (S923).

**[0465]** Next, the response/request executing section 35 acquires an address of the content (media segment) to be played, on the basis of a result of the analysis of the MPD data. Here, in order to acquire the address from the MPD meta data, the response/request executing section 35 determines whether or not data of an external resource indicated by an xlink of the MPD data is necessary (S924). In a case where it is determined that the data of the external resource is unnecessary to acquire the address (NO in S924), the response/request executing section 35 analyzes the MPD data, and extracts a content storage server list. On the basis of the content storage location list thus extracted, the acquisition location selecting section 38 selects one of pieces of address information of servers, in the same manner as a process of a client 4 in accordance with Embodiment 4 (S623 through S634 in Fig. 23) (S928). On the other hand, in a case where it is determined that the data of the external resource is necessary to acquire the address (YES in S924), the response/request executing section 35 requests the external resource data with use of the address of the external resource indicated by the xlink (S925).

**[0466]** On receipt of the external resource data thus requested from a server 2 indicated by the xlink (S926), the response/request executing section 35 replaces, with the external resource data thus received, a part of the MPD data thus received, which part is indicated by the xlink with which the external resource is acquired (S927). That is, the response/request executing section 35 updates the MPD data. Then, the response/request executing section 35 analyzes the MPD data, so as to extract the content storage server list. On the basis of the content

storage location list thus extracted, the acquisition location selecting section 38 selects one of the pieces of the address information of the servers, in the same manner as the process of the client 4 in accordance with Embodiment 4 (S623 through S634 in Fig. 23) (S928). The response/request executing section 35 creates a request for a media segment on the basis of (i) address information of the server thus selected and (ii) address information of a media segment to be acquired. Then, the response/request executing section 35 transmits the request thus created to the server thus selected (S929). Then, the response/request executing section 35 receives media segments sequentially, in the same manner as Embodiment 4 (S724). Note that, a process for receiving media segments is identical to a process of Embodiment 4, and therefore is shown as "S724" in Fig. 40.

**[0467]** Here, in a case where the response/request executing section 35 receives all movie fragments, the response/request executing section 35 checks whether or not all the movie fragments of the content thus requested have been received (S632). In a case where the response/request executing section 35 confirms that all the movie fragments have been received (YES in S632), the process is finished.

**[0468]** On the other hand, in a case where the response/request executing section 35 determines that there is any movie fragment which has not been received (NO in S632), the response/request executing section 35 determines whether or not a next media segment can be acquired, by determining whether or not all media segments included in a corresponding Period, which is a target to be played in the MPD data, have been received (S930). In a case where all the media segments in the corresponding Period have been received (YES in S930), the response/request executing section 35 acquires a next Period (S931).

**[0469]** In a case where, in the corresponding Period, there is a media segment which can be received, the response/request executing section 35 starts to carry out a process of receiving such a media segment. Then, the client status determining section 36 determines whether or not a device from which such a media segment is received should be changed, in the same manner as Embodiment 3 (S725). A determination method here is identical to a determination method of Embodiment 4. Then, acquisition of media segments is continued.

Example 5]

**[0470]** The following description deals with details of Embodiment 5 more specifically, with use of Example 5 shown in Fig. 41. Example 5 is a view showing an example operation sequence of a content distribution system 1 which instructs a client 4, which serves as a content playing device, to acquire a content from one of pieces of address information included in a content storage server list.

**[0471]** Note that, in Example 5, as a premise, a content

1 having a format shown in Fig. 21, MPD data which (i) is meta data of the content 1 and (ii) has a structure shown in Fig. 38, and external resource data are stored in a content storage section 5 of each of servers 2a, 2b, and 2c. Further, a media segment of the content 1, stored in each of the servers 2a, 2b, and 2c, has been similarly divided into a plurality of media segments.

**[0472]** Furthermore, as in Example 4, one session is defined as a process from a time that the content playing device transmits a request message to a time that the content playing device receives a response message in response to the request message.

**[0473]** As shown in Fig. 41, in a session 1010, a client 4g transmits, to the server 2a, a request message which requests transmission of MPD data of the content 1 (request 1011). In a case where the server 2a receives the request 1011, a response executing section 15 reads out the MPD data thus requested from the content storage section 5a (process 1012), and transmits a response message to the client 4g (response 1013).

**[0474]** In a case where the client 4g receives the response 1013, a response/request executing section 35 analyzes the MPD data thus received, so as to acquire a media segment (process 1020). Then, in a case where it is determined that the MPD data includes an instruction to refer to an external resource which is necessary to acquire, for example, location information of the media segment, the response/request executing section 35 acquires the external resource from the server 2.

**[0475]** Next, in a session 1030, the client 4g transmits a request message which requests external resource data (request 1031).

**[0476]** In a case where the server 2a receives the request 1031, a response executing section 15 instructs an acquisition location specifying section 18 to specify a plurality of servers 2. On receipt of such an instruction, the acquisition location specifying section 18 creates a storage location server list on the basis of content-storage-location information stored in a content-storage-location information storage section 13 (process 812). Here, in the content-storage-location information storage section 13, there is no server information related to a content 1 (process 813). Accordingly, in order to create the content storage server list, the acquisition location specifying section 18 transmits, to the servers 2b and 2c via the response executing section 15, an inquiry as to storage information of the content 1 (requests 814 and 815). Then, the acquisition location specifying section 18 obtains (i) responses (responses 816 and 817) to the inquiry and (ii) response times of such responses. Then, the acquisition location specifying section 18 creates the content storage server list on the basis of such results of the responses (process 818). Next, the response executing section 15 (i) acquires external resource data thus requested from a content storage section 5a, (ii) updates the external resource data with use of information of the content storage location list thus created, and (iii) creates external resource data thus requested (process 1032).

Then, the response executing section 15 transmits, to the client 4g via a server communication section 11, a response message including the external resource data thus created (response 1033).

**[0477]** In a case where the client 4g receives the response 1033, the client 4g updates, with use of the external resource data, the MPD data which has been already received. The acquisition location selecting section 38 selects, with use of the MPD data thus updated, a server 2 as a device from which the content is acquired (process 1040). A selection method here can be such that a server 2 described at a top of the list is selected, or, if information (such as priorities) is added, a server 2 is selected on the basis of such information. Further, in a case where delay information is received from a client status determining section 36, it is possible to select, in consideration of such delay information, a server 2 as the device from which the content is acquired.

**[0478]** In a case where the server 2 (here, the server 2b) is selected, the response/request executing section 35 creates, on the basis of the MPD, a URL to acquire a media segment, and start acquiring media segments sequentially (session 1050). Details of a process of acquiring media segments are identical to those of a process of Example 3 (Example 4), and therefore are omitted here for the sake of simple explanation.

**[0479]** In a case where acquisition of all media segments in a Period of the MPD data is completed, the client 4g start acquiring media segments included in a next Period. In a case where acquisition of external resource data with use of an xlink is necessary to acquire media segment information, the client 4g transmits a request for an external resource to an address described in a corresponding xlink, in the same manner as the session 1030.

**[0480]** Then, both the client 4 and the server 2 repeat operations of sessions 1010 through 1050, so as to acquire all the media segments. Playing is thus completed.

[Solution to Problem]

**[0481]** In order to achieve aforementioned object, a content distributing device for transmitting, in response to a request, a content to a source which has transmitted the request, in accordance with the present invention, includes: determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; content transmitting means for transmitting, in response to the request, the content thus requested to the relaying device in a case where the determining means determines that the source is the relaying device; content-storage-location information generating means for generating content-storage-location information by associating (A) the content transmitted by the content transmitting means with (B) an address of the relaying device, which is a

destination to which the content is transmitted, or an address of the content playing device, to which the content is transferred from the relaying device; and content-acquiring-location instructing means for transmitting, in response to the request, an instruction to the content playing device which is the source in a case where the determining means determines that the source is the content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (ii) a content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested.

**[0482]** In order to achieve aforementioned object, a method for controlling content distributing device for transmitting, in response to a request, a content to a source which has transmitted the request, the method in accordance with the present invention includes: a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; a content transmitting step of transmitting, in response to the request, the content thus requested to the relaying device in a case where it is determined that, in the determining step, the source is the relaying device; a content-storage-location information generating step of generating content-storage-location information by associating (A) the content transmitted in the content transmitting step with (B) an address of the relaying device, which is a destination to which the content is transmitted, or an address of the content playing device, to which the content is transferred from the relaying device; and a content-acquiring-location instructing step of transmitting, in response to the request, an instruction to the content playing device which is the source in a case where it is determined that, in the content-storage-location information generating step, the source is the content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (ii) a content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested.

**[0483]** According to the arrangement, upon receipt of the request from the relaying device, the content transmitting means transmits the content thus requested to the relaying device which is the source, and the content-storage-location information generating means generates the content-storage-location information by associating (A) the content transmitted from the content transmitting means with (B) the address of the relaying device, which is the destination of the content, or the address of the content playing device, to which the content is transferred from the relaying device. Further, upon receipt of the request from the content playing device, the content-

acquiring-location instructing means transmits, to the content playing device which is the source, the instruction to acquire the content from (I) the relaying device indicated by an address associated, in the content-storage-location information, with the content thus requested or (II) the content playing device indicated by an address associated, in the content-storage-location information, with the content thus requested. Here, the relaying device and the content playing device possesses the content thus acquired, and the content-storage-location information is information indicative of which relaying device or content playing device possesses a content.

**[0484]** That is, the content distributing device associates (A) a content which has been transmitted before with (B) a relaying device or a content playing device which possesses the content, and, upon receipt of a request from a certain content playing device, the content distributing device does not directly transmit the content to the content playing device which is the source, but transmits, to the content playing device which is the source, an instruction to acquire the content from a relaying device or a content playing device which possesses the content thus requested. The content playing device, which is the source, acquires the content thus requested from a designated relaying device or a designated content playing device. Therefore, if the designated relaying device or the designated content playing device possesses the content, it is possible to complete transmission and reception of the content with use of only (A) the content playing device which is the source and (B) the designated relaying device or the designated content playing device. That is, the content playing device, which is the source, can acquire content, without carrying out a process for transmitting the content.

**[0485]** This makes it possible to reduce (A) a load of a network, which is used to transmit data from the content distributing device, and (B) a load of the content distributing device. Among processes carried out by the content distributing device, the relaying device, and the content playing device, a process for transmitting and receiving the content is a process which applies the heaviest load, and the process applies the heaviest load of the network among the content distributing device, the relaying device, and the content playing device. However, even if, for example, the number of content playing devices is increased and the number of requests to the content distributing devices is therefore increased, it is possible to reduce (A) an increase in load of the network which is used to transmit data from the content distributing device and (B) an increase in load of the content distributing device. Therefore, a large number of content playing devices can acquire contents, without increasing throughput of the content distributing device or capacity of the network.

**[0486]** It is preferable that the content distributing device in accordance with the present invention determine that, in a case where the request contains transmission path information indicative of a transmission path via



which the request is transferred, the source is a relaying device and, in a case where the request does not contain the transmission path information, the source is a content playing device.

**[0487]** According to the arrangement, the determining means determines that, in a case where the request contains transmission path information indicative of a transmission path via which the request is transferred, the source is a relaying device and, in a case where the request does not contain the transmission path information, the source is a content playing device. That is, the determining means determines that the source is the content playing device in a case where the request is directly transmitted from the content playing device, whereas the content acquiring device is the relaying device in a case where the request is transmitted from a device other than the content playing device.

**[0488]** As described above, a content distributing device transmits a requested content to a relaying device in a case where the relaying device is a source which has transmitted a request, whereas, in a case where a content playing device is the source, the content distributing device transmits, to the content playing device, an instruction to acquire the requested content from a relaying device or a content playing device which possesses the requested content. The content distributing device can, therefore, always transmit the content to the content playing device via a designated relaying device or a designated content playing device. Accordingly, in a case where the designated relaying device or the designated content playing device possesses the content requested by the content playing device which is the source, the content distributing device does not need to transmit the content to the content playing device. This makes it possible to reduce (A) the load of the network which is used to transmit data from the content distributing device and (B) the load of the content distributing device.

**[0489]** Further, it is preferable that, in the content distributing device in accordance with the present invention, in a case where there are a plurality of pieces of the content-storage-location information which contain a plurality of addresses, respectively, each of the plurality of addresses being associated with the content thus requested, the content-acquiring-location instructing means (A) create a content-storage-location list containing the plurality of addresses included in the plurality of pieces of content-storage-location information and (B) transmit, to the content playing device which is the source, an instruction to acquire the content from (I) a relaying device indicated by an address contained in the content-storage-location list thus created or (II) a content playing device indicated by an address contained in the content-storage-location list thus created.

**[0490]** According to the arrangement, in a case where there are a plurality of pieces of the content-storage-location information which includes a plurality of addresses, respectively, each of the plurality of addresses being associated with the content thus requested, the content-

acquiring-location instructing means (A) creates a content-storage-location list including the plurality of addresses included in the plurality of pieces of content-storage-location information and (B) transmit, to the content playing device which is the source, an instruction to acquire the content from (I) a relaying device indicated by an address included in the content-storage-location list thus created or (II) a content playing device indicated by an address included in the content-storage-location list thus created.

**[0491]** Accordingly, the content playing device, which is the source, selects (A) the relaying device indicated by the address included in the content-storage-location list or (B) the content playing device indicated by the address included in the content-storage-location list, and acquires the content from the relaying device or the content playing device thus selected. The content playing device, which is the source, can therefore acquire the content from an optimum device depending on a status of the content playing device and a status of the relaying device or the content playing device which possesses the content.

**[0492]** Further, it is preferable that, the content distributing device in the present invention, the content-storage-location information generating means generate the content-storage-location information by associating (A) the content which has been transmitted by the content transmitting means with (B) date and time when the content transmitting means has transmitted the content; and the content-acquiring-location instructing means create the content-storage-location list by (I) arranging the plurality of addresses, which are contained in the plurality of pieces of content-storage-location information, on the basis of the date and time associated with the content and (II) adding priorities to the plurality of addresses so that an address having later date and time gets a higher priority.

**[0493]** According to the arrangement, in the content-storage-location information, (A) a content, (B) a relaying device or a content playing device which possess the content, and (C) date and time when the content has been transmitted to the relaying device or the content playing device, i.e., date and time when the relaying device or the content playing device has held the content are associated with one another. Then the content-acquiring-location instructing means creates the content-storage-location list by (I) arranging the plurality of addresses, which are contained in the plurality of pieces of content-storage-location information, on the basis of date and time associated with the content identification information and (II) adding priorities to the plurality of addresses so that an address having later date and time gets a higher priority.

**[0494]** That is, the content-storage-location list includes the plurality of relaying devices or the plurality of content playing devices which possess the content requested by the content playing device which is the source, so that the plurality of relaying devices or the

plurality of content playing devices are arranged in order of time, specifically, in order of time when each of the plurality of relaying devices or the plurality of content playing devices has possessed the content. The content playing device, which is the source, can therefore select, as, e.g., a device from which the content is acquired, a relaying device or a content playing device which has stored the content recently.

**[0495]** There may occur, for example, a case where a content transmitted by the content distributing device is updated to obtain a new data or a case where a relaying device or a content playing device which possesses content discards the content thus held or modifies the content. Even in such a case, the content playing device which is the source can surely acquire the content same as a content transmitted by the content distributing device by acquiring the content from a relaying device or a content playing device which contains the latest date and time when the relaying device or the content playing device has stored the content. This makes it possible to surely acquire the content same as a content transmitted by the content distributing device.

**[0496]** Further, it is preferable that a content distributing device in accordance with the present invention further include: distance calculating means for calculating, on the basis of an address contained in any one of the plurality of pieces of content-storage-location information, a physical or network-structural distance between (A) a relaying device or a content playing device which is indicated by the address and (B) the content playing device that the content-acquiring-location instructing means instructs on a device from which the content is acquired, wherein: the content-acquiring-location instructing means creates the content-storage-location list by (I) arranging the plurality of addresses, which are contained in the respective plurality of pieces of content-storage-location information, on the basis of distances calculated by the distance calculating means, and (II) adding priorities to the plurality of addresses so that an address having a shorter distance gets a higher priority.

**[0497]** According to the arrangement, the distance calculating means calculates, on the basis of an address contained in any one of the plurality of pieces of content-storage-location information, a physical or network-structural distance between (A) a relaying device or a content playing device which is indicated by the address and (B) the content playing device that the content-acquiring-location instructing means instructs on a device from which the content is acquired. Further, the content-acquiring-location instructing means creates the content-storage-location list by (I) arranging the plurality of addresses, which are contained in the respective plurality of pieces of content-storage-location information, on the basis of distances calculated by the distance calculating means, and (II) adding priorities to the plurality of addresses so that an address having a shorter distance gets a higher priority.

**[0498]** That is, the content-storage-location list in-

cludes a plurality of relaying devices and a plurality of content playing devices, each of which possesses the content requested by the content playing device which (i) is the source and (ii) the content-acquiring-location instructing means instructs on a device from which the content is acquired. The plurality of relaying devices and the plurality of content playing devices are listed so that a device having a shorter physical or network-structural distance gets a higher priority. Accordingly, with reference to the content-storage-location list, the content playing device which is the source can, for example, select, as a device from which the content is acquired, a relaying device or a content playing device which is the nearest from the content playing device itself. This makes it possible to reduce a load of the network in a case where the content playing device which is the source acquires the content.

**[0499]** Further, it is preferable that the content distributing device in accordance with the present invention further include transmission record creating means for creating a response transmission record by associating (A) a destination to which a response is transmitted in response to the request with (B) date and time when the response has been transmitted, wherein, with reference to response transmission record created by the transmission record creating means, the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from a relaying device or a content playing device which (i) is indicated by an address that the content-storage-location information associates with the content thus requested and (ii) is not included in the response transmission record within a predetermined time period

**[0500]** According to the arrangement, transmission record creating means makes a response transmission record by associating (A) a destination to which a response is transmitted in response to the request with (B) date and time when the response has been transmitted. Then, with reference to response transmission record created by the transmission record creating means, the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from a relaying device or a content playing device which (i) is indicated by an address that the content-storage-location information associates with the content thus requested and (ii) is not included in the response transmission record within a predetermined time period.

**[0501]** By referring the response transmission record, the content-acquiring-location instructing means can predict date and time when the relaying device or the content playing device, which has been the destination to which the response is transmitted, has received the response. To put it another way, the content-acquiring-location instructing means can predict date and time when the relaying device or the content playing device has carried out transmission of the request, reception of

the response, or a process regarding the transmission of the request and the reception of the response. Accordingly, "a destination which is not included in the response transmission record within a predetermined time period" means a relaying device or a content playing device which is considered not to have carried out a process regarding the transmission and the reception of the response within the predetermined time period.

**[0502]** That is, the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content thus requested from a relaying device or a content playing device which (i) possesses the content thus requested and (ii) is considered not to have carried out a process regarding transmission and reception of the content within a predetermined time period. Accordingly, when the content playing device, which is the source, acquires content from a relaying device or a content playing device designated by the content-acquiring-location instructing means, it is possible to reduce a delay caused by an increase in throughput of the relaying device or the content playing device from which the content is acquired.

**[0503]** Further, a content playing device in accordance with the present invention (A) transmits a request to the content distributing device, (B) receives a content-storage-location list in response to the request, and (C) acquires the content thus requested from a relaying device or a content playing device which is indicated by an address included in the content-storage-location list thus received, wherein, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, the relaying device or the content playing device, from which the content is acquired, is changed to a relaying device or a content playing device which is indicated by another address included in the content-storage-location list.

**[0504]** According to the arrangement, the content playing device selects one of a relaying device and a content playing device which is indicated by an address included in the content-storage-location list thus received. In a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed when the content playing device acquires the content thus requested from the relaying device or the content playing device thus selected, the content playing device changes the relaying device or the content playing device thus selected to a relaying device or a content playing device which is indicated by another address included in the content-storage-location list.

**[0505]** The content playing device can, therefore, keep a receiving speed required to acquire content faster than a predetermined receiving speed, and can stably acquire the content without causing a huge delay.

**[0506]** The delay in acquiring of content may be caused by, for example, the following reasons: an increase in throughput of a relaying device or a content playing device from which the content is acquired; or a deterioration in communication status of a network between (i) the

content playing device and (ii) the relaying device or the content playing device from which the content is acquired. The content playing device changes, due to the delay, the relaying device or the content playing device, from which the content is acquired, to another relaying device or another content playing device, so that it is possible to effectively use resources for (i) the relaying device or the content playing device from which the content is acquired and (ii) a network between a content playing device for acquiring the content and the relaying device or the content playing device from which the content is acquired.

**[0507]** A content playing device in accordance with the present invention (A) transmits a request to the content distributing device, (B) receives the content-storage-location list in response to the request, and (C) acquires a requested content from a relaying device or a content playing device indicated by an address which is the highest on the content-storage-location list thus received, and, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, the content playing device changes a relaying device or a content playing device, from which the requested content is to be acquired, to a relaying device or a content playing device indicated by an address which is the second highest on the content-storage-location list.

**[0508]** According to the arrangement, the content playing device selects a relaying device or a content playing device which is indicated by an address which is the highest on the content-storage-location list thus received. In a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed when the content playing device acquires the content thus requested from the relaying device or the content playing device thus selected, the content playing device changes the relaying device or the content playing device thus selected, from which the requested content is to be acquired, to a relaying device or a content playing device indicated by another address which is the second highest on the content-storage-location list.

**[0509]** The content playing device can, therefore, keep a receiving speed required to acquire content faster than a predetermined receiving speed, and can stably acquire the content without causing a huge delay.

**[0510]** Further, in a case where priorities are assigned to each of relaying devices and content playing devices on the basis of date and time when each relaying device or content playing device has stored the content, the content playing device for acquiring the content can acquire the content from a relaying device or a content playing device which has the highest possibility to possess the content same as that transmitted by the content distributing device. It is therefore possible to quickly and surely acquire the content same as that transmitted by the content distributing device. Meanwhile, in a case where priorities are assigned to each of relaying devices and content playing devices on the basis of a distance between each relaying device or content playing device, from

which the content is acquired, and the content playing device for acquiring the content, the content playing device acquires the content from a relaying device or a content playing device having a shortest distance with respect to the content playing device for acquiring the content. It is therefore possible to stably acquire the content while reducing a load of a network.

**[0511]** Further, a content distributing system in accordance with the present invention includes: the content distributing device; a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device; and a content playing device for requesting the content distributing device to transmit a content and acquiring the content thus requested from a device designated by the content distributing device.

**[0512]** According to the arrangement, the content distributing system has an effect same as that of the content distributing device.

**[0513]** Further, a content distributing device for transmitting, in response to a request, a content to a source which has transmitted the request, in accordance with the present invention, includes: determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; content-storage-location information acquiring means for acquiring, in response to the request, an address of another content distributing device possessing the content thus requested, among predetermined other content distributing devices, in a case where the determining means determines that the source is a content playing device; and content-acquiring-location instructing means for transmitting, to the content playing device which is the source, an instruction to acquire the content from the another content distributing device which is indicated by the address acquired by the content-storage-location information acquiring means.

**[0514]** Further, a method for controlling a content distributing device for transmitting, in response to a request, a content to a source which has transmitted the request, the method in accordance with the present invention, includes: a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested; a content-storage-location information acquiring step of acquiring, in response to the request, an address of another content distributing device including the content thus requested, among content distributing devices connected to the content distributing device, in a case where it is determined that, in the determining step, the source is the content playing device; and a content-acquiring-location instructing step of transmitting, to the content playing device which is the source, an instruction to acquire

the content from the another content distributing device which is indicated by the address acquired in the content-storage-location information acquiring step.

**[0515]** According to the arrangement, the determining means determines whether a source which has transmitted a request is a relaying device or a content playing device, and, in a case where the determining means determines that the source is a content playing device, the content-storage-location information acquiring means acquires, in response to the request, an address of another content distributing device including the content thus requested, among predetermined content distributing devices. Then, the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from the another content distributing device which is indicated by the address acquired by the content-storage-location information acquiring means.

**[0516]** That is, in a case where the content distributing device receives a request from a content playing device, the content distributing device does not directly transmit a content to the content playing device which is a source which has transmitted the request, but transmits, to the content playing device which is the source, an instruction to acquire the content from another content distributing device which possesses the content thus requested. The content playing device, which is the source, acquires the content thus requested from the designated another content distributing device. That is, the content distributing device, has received the request from the content playing device, can acquire content, without carrying out a process for transmitting the content.

**[0517]** In a case where a load of a network which is used to transmit data from the content distributing device received the request from the content playing device and a load of the content distributing device are relatively large, the content distributing device causes another content distributing device to acquire the content. This makes it possible to distribute the load of the network which is used to transmit data from the content distributing device and the load of the content distributing device.

**[0518]** Further, it is preferable that, in a content distributing device in accordance with the present invention, the content-storage-location information acquiring means transmits, to the predetermined other content distributing devices, an inquiry as to whether or not the predetermined other content distributing devices include the content thus requested, so as to acquire the address of the another content distributing device that has responded, to the inquiry, that the content distributing device possesses the content thus requested.

**[0519]** According to the arrangement, the content-storage-location information acquiring means transmits, to the predetermined other content distributing devices, an inquiry as to whether or not the predetermined other content distributing devices include the content thus requested, so as to acquire the address of the another content distributing device that has responded, to the inquiry, that

the content distributing device possesses the content thus requested.

**[0520]** The content-acquiring-location instructing means transmits, to the content playing device, an instruction to acquire the content on the basis of the address acquired by the content-storage-location information acquiring means having carrying out the inquiry. This makes it possible to transmits, to the content playing device, the instruction to acquire the content on the basis of highly accurate (correct) information. The content playing device can therefore surely acquire the content thus requested.

**[0521]** Further, it is preferable that the content distributing device in accordance with the present invention further include content-storage-location information generating means for (A) generating content-storage-location information by associating (i) the address of the another content distributing device including the content, which address has been acquired by the content-storage-location information acquiring means, with (ii) the content identification information indicative of the content and (B) causing a storage section to store the content-storage-location information, wherein the content-storage-location information acquiring means (I) reads out the content-storage-location information from the storage section, and (II) acquires the address, associated with the content identification information, from the content-storage-location information in a case where the content-storage-location information thus read out contains the content identification information indicative of the content thus requested, or transmits the inquiry to thereby acquire the address of the another content distributing device possessing the content thus requested in a case where the content-storage-location information thus read out does not contain the content identification information indicative of the content thus requested.

**[0522]** According to the arrangement, content-storage-location information generating means (A) generates content-storage-location information by associating (i) the address of the another content distributing device including the content, which address has been acquired by the content-storage-location information acquiring means, with (ii) the content identification information indicative of the content and (B) causing a storage section to store the content-storage-location information. Then, the content-storage-location information acquiring means (I) reads out the content-storage-location information from the storage section, and (II) acquires the address associated with the content identification information in a case where the content-storage-location information thus read out contains the content identification information indicative of the content thus requested, or transmits the inquiry to thereby acquire the address of the another content distributing device possessing the content thus requested in a case where the content-storage-location information thus read out does not contain the content identification information indicative of the content thus requested.

**[0523]** That is, in a case where the content-storage-location information acquiring means acquires the address of the another content distributing device including the content thus requested, the content-storage-location information acquiring means refers the content-storage-location information stored in the storage section. In a case where the storage section stores the content-storage-location information including the content identification information indicative of the content thus requested, i.e., in a case where the content-storage-location information acquiring means has acquired before an address of another content distributing device including the content, the content-storage-location information acquiring means acquires the address from the content-storage-location information stored in the storage section.

**[0524]** Meanwhile, in a case where the content-storage-location information acquiring means refers the content-storage-location information stored in the storage section and the storage section does not store the content-storage-location information including the content identification information indicative of the content thus requested, i.e., in a case where the content-storage-location information acquiring means has never acquired an address of another content distributing device including the content, the content-storage-location information acquiring means transmits the inquiry, so as to acquire an address of another content distributing device including the content thus requested.

**[0525]** It is predicted that another content distributing device, indicated by an address included in the content-storage-location information stored in the storage section, would include the content indicated by the content identification information corresponding to the address.

**[0526]** It is therefore possible to acquire the address of the another content distributing device including the content thus requested by carrying out a simple process, i.e., by using the address which has been acquired before. This makes it possible to reduce a process load of the content distributing device.

**[0527]** Further, it is preferable that, in the content distributing device in accordance with the present invention, the content-storage-location information acquiring means (i) acquire a plurality of addresses included in a plurality of content distributing devices, respectively, each of the plurality of content distributing devices including the content thus requested, and (ii) create a content storage server list including the plurality of addresses thus acquired and content identification information indicative of the content; and the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from the content distributing device indicated by the address included in the content storage server list created by the content-storage-location information acquiring means.

**[0528]** According to the arrangement, the content-storage-location information acquiring means (i) acquires a

plurality of addresses included in a plurality of content distributing devices, respectively, each of the plurality of content distributing devices including the content thus requested, and (ii) creates a content storage server list including the plurality of addresses thus acquired and content identification information indicative of the content; and the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from the content distributing device indicated by the address included in the content storage server list created by the content-storage-location information acquiring means.

**[0529]** That is, the content playing device can acquire the content thus requested from one of the other content distributing devices. This makes it possible to distribute the load of the network which is used to transmit data from the content distributing device and the load of the content distributing device.

**[0530]** Further, a content playing device for (A) transmitting a request to the content distributing device, (B) receiving the content storage server list in response to the request, and (C) acquiring a requested content from another content distributing device indicated by one of a plurality of addresses included in the content storage server list thus received, the content playing device in accordance with the present invention, includes acquiring location changing means for, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, changing the another content distributing device, from which the requested content is to be acquired, to another content distributing device indicated by another address, which is different from the one of the plurality of addresses, included in the content storage server list.

**[0531]** According to the arrangement, the content playing device selects another content distributing device indicated by one of the plurality of addresses included in the content storage server list thus received. In a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed when the content playing device acquires the content thus requested from the another content distributing device thus selected, the content playing device changes the another content distributing device, from which the requested content is to be acquired, to a still another content distributing device indicated by another address, which is different from the one of the plurality of addresses, included in the content storage server list.

**[0532]** The content playing device can, therefore, keep a receiving speed required to acquire content faster than a predetermined receiving speed, and can stably acquire the content without causing a huge delay.

**[0533]** The delay in acquiring of content may be caused by, for example, the following reasons: an increase in throughput of a content distributing device from which the content is acquired; or a deterioration in communication status of a network between the content playing device and the content distributing device from which the

content is acquired. The content playing device changes, due to the delay, the content distributing device, from which the content is acquired, to another content distributing device which is different from the content distributing device, so that it is possible to effectively use resources for (i) the content distributing device from which the content is acquired and (ii) a network between a content playing device for acquiring the content and the content distributing device from which the content is acquired.

**[0534]** Further, a content distributing system in accordance with the present invention includes the content distributing device, a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device; and the content playing device for requesting the content distributing device to transmit the content, and acquiring the content thus requested from a device designated by the content distributing device.

**[0535]** According to the arrangement, the content distributing system has an effect same as that of the content distributing device.

**[0536]** Further, the content distributing device for transmitting, in response to a request, data to a source which has transmitted the request, the content distributing device in accordance with the present invention includes: managing means for managing (A) content and (B) meta data of the content, the meta data containing (i) content-storage-location information in which content identification information for specifying the content and an address of another content distributing device including the content are associated with each other or (ii) a storage-location address indicative of a location of the content-storage-location information; request determining means for determining whether the request is a content request or a meta data request; and transmitting means for transmitting the content thus requested to the source in a case where the request determining means determines that the request is the content request, and for transmitting the meta data thus requested to the source in a case where the request determining means determines that the request is the meta data request.

**[0537]** Further, a method for controlling a content distributing device for transmitting, in response to a request, data to a source which has transmitted the request, the method managing (A) content and (B) meta data of the content, the meta data containing (i) content-storage-location information in which content identification information for specifying the content and an address of another content distributing device including the content are associated with each other or (ii) a storage-location address indicative of a location of the content-storage-location information, the method in accordance with the present invention includes a request determining step of determining whether the request is a content request or a meta data request; and a transmitting step of transmitting the content thus requested to the source in a case where it is determined that, in the request determining step, the

request is the content request, and of transmitting the meta data thus requested to the source in a case where it is determined that, in the request determining step, the request is the meta data request.

**[0538]** According to the arrangement, the managing means manages (A) content and (B) meta data of the content, the meta data containing (i) content-storage-location information in which content identification information for specifying the content and an address of another content distributing device including the content are associated with each other or (ii) a storage-location address indicative of a location of the content-storage-location information, and the request determining means determines whether the request is a content request or a meta data request, and the transmitting means transmits the content thus requested to the source in a case where the request determining means determines that the request is the content request, and for transmitting the meta data thus requested to the source in a case where the request determining means determines that the request is the meta data request.

**[0539]** That is, upon receipt of a content request from the content playing device, the content distributing device directly transmits the content to the content playing device which is the source. Meanwhile, upon receipt of a meta data request from the content playing device, the content distributing device does not directly transmit the content to the content playing device which is the source, but transmits, to the content playing device which is the source, an instruction to acquire the content from another content distributing device including the content thus requested. The content playing device which is the source acquires the content thus requested from a designated content distributing device. That is, in a case where the content playing device requests to acquire the content with use of meta data of the content, not the content distributing device which has received the meta data request, but the another content distributing carries out a process for transmitting the content. The content playing device, which is the source, can therefore acquire the content.

**[0540]** Therefore, in a case where (i) the content playing device requests to acquire a content with use of meta data of the content and (ii) a load of a network which is used to transmit data from the content distributing device which has received the request from the content playing device and a load of the content distributing device are relatively large, the content playing device acquires the content from another content distributing device. This makes it possible to distribute the load of the network which is used to transmit the data from the content distributing device and the load of the content distributing device.

**[0541]** It is preferable that, in the content distributing device in accordance with the present invention, the request determining means determine whether the request is the content request, the meta data request, or a content-storage-location information request including the

storage-location address; and, in a case where the request determining means determines that the request is the content-storage-location information including the storage-location address, the transmitting means transmit, to a device which is the source, the content-storage-location information whose location is indicated by the storage-location address.

**[0542]** According to the arrangement, the request determining means determines whether the request is the content request, the meta data request, or a content-storage-location information request including the storage-location address; and, in a case where the request determining means determines that the request is the content-storage-location information including the storage-location address, the transmitting means transmits, to a device which is the source, the content-storage-location information whose location is indicated by the storage-location address.

**[0543]** Therefore, upon receipt of the meta data request from the content playing device, the content distributing device can send the content playing device with not only content-storage-location information indicative of a device from which the content corresponding to the meta data is acquired, but also content-storage-location information which indicates, when the content distributing device receives the content storage-location information request from the playing device, a device from which the content corresponding to the meta data is acquired. By, for example, transmitting the content-storage-location information request when the content is played, it is possible to know the another content playing device including the content can, therefore, know the another content distributing device including the content when the content is played.

**[0544]** Accordingly, the content distributing device can send the content playing device with highly accurate (correct) information, and the content playing device can reduce such an error that the content playing device cannot acquire a requested content from a device from which the content is acquired. This makes it possible to stably acquire the content.

**[0545]** Further, it is preferable that a content distributing device in accordance with the present invention further include: content-storage-location information acquiring means for transmitting, to predetermined other content distributing devices, an inquiry as to whether or not the predetermined other content distributing devices contain a predetermined content, and acquiring an address of a content distributing device that has responded, to the inquiry, that the content distributing device includes the predetermined content; content-storage-location information generating means for (A) generating content-storage-location information by associating (i) the address of the content distributing device including the predetermined content, which address has been acquired by the content-storage-location information acquiring means, with (ii) the content identification information for specifying the predetermined content and (B) causing a

storage section to store the content-storage-location information; and update determining means for determining whether to update the content-storage-location information stored in the storage section, wherein, in a case where the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines to update the content-storage-location information whose location is indicated by the storage-location address, the content-storage-location information acquiring means transmits the inquiry to acquire the address, the content-storage-location information generating means generates the content-storage-location information based on the address, and the transmitting means transmits the content-storage-location information to a device which is the source, or, in a case where the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines not to update the content-storage-location information whose location is indicated by the storage-location address, the transmitting means transmits, to the device which is the source, the content-storage-location information whose location is indicated by the storage-location address.

**[0546]** According to the arrangement, the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines to update the content-storage-location information whose location is indicated by the storage-location address, the content-storage-location information acquiring means transmits the inquiry to acquire the address, the content-storage-location information generating means generates the content-storage-location information based on the address, and the transmitting means transmits the content-storage-location information to a device which is the source. Meanwhile, the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines not to update the content-storage-location information whose location is indicated by the storage-location address, the transmitting means transmits, to the device which is the source, the content-storage-location information whose location is indicated by the storage-location address.

**[0547]** That is, in a case where the update determining means determines not to update the content-storage-location information stored in the storage section when the content-storage-location information acquiring means acquires the address of the content distributing device including the requested content, the content-storage-location information acquiring means acquires the address from the content-storage-location information stored in the storage section. Meanwhile, in a case where the update determining means determines to update the con-

tent-storage-location information stored in the storage section, the content-storage-location information acquiring means transmits the inquiry to thereby acquire the address of the content distributing device including the content thus requested.

**[0548]** Here, "a case where the update determining means determines to update the content-storage-location information whose location is indicated by the storage-location address" means, for example, a case where the content-storage-location information stored in the storage section is old, or a case where the content-storage-location information including content identification information indicative of the requested content is not stored in the storage section, i.e., a case where the content-storage-location information acquiring means has never acquired before an address of a content distributing device including the content.

**[0549]** Therefore, in a case where the update determining means determines not to update the content-storage-location information stored in the storage section, the content-storage-location information acquiring means can acquire the address of the content distributing device including the required content by carrying out a simple process, i.e. by using the address which has been acquired before. This makes it possible to reduce the process load of the content distributing device.

**[0550]** Further, by updating content-storage-location information stored in the storage section, the content distributing device can send the content playing device with highly accurate (correct) information, and the content playing device can reduce such an error that the content playing device cannot acquire a requested content from a device from which the content is acquired. This makes it possible to stably acquire the content.

**[0551]** Further, it is preferable that, in the content distributing device in accordance with the present invention, the meta data can include a plurality of storage location addresses each indicative of a location of content-storage-location information which is set per unit into which the content is divided at a predetermined time interval.

**[0552]** According to the arrangement, the meta data can include a plurality of storage location addresses each indicative of a location of content-storage-location information which is set per unit into which the content is divided at a predetermined time interval. Therefore, the content distributing device can send, to the content playing device, pieces of content-storage-location information, each of which is sent per unit obtained by dividing the content by a predetermined time interval.

**[0553]** The content distributing device can therefore improve a degree of freedom of a device from which a content is acquired more than that of the content playing device. This makes it possible to distribute the load of the network which is used to transmit data from the content distributing device and the load of the content distributing device.

**[0554]** Further, it is preferable that, in the content distributing device in accordance with the present invention,



the content include a plurality of media segments; and the content divided by the predetermined time interval include at least one media segment.

**[0555]** According to the arrangement, the content divided by the predetermined time interval includes at least one media segment. It is therefore possible to manage, per unit of a media segment, the degree of freedom of the device from which the content is acquired.

**[0556]** Further, it is preferable that, in the content distributing device in accordance with the present invention, the meta data include a content storage server list containing the plurality of pieces of content-storage-location information or a storage location address indicative of a location of the content storage server list.

**[0557]** According to the arrangement, the meta data includes a content storage server list including the plurality of pieces of content-storage-location information or a storage-location address indicative of the content storage server list. The content distributing device can therefore cause the content playing device to acquire one of the other content distributing devices which correspond to the meta data. This makes it possible to distribute the load of the network which is used to transmit data from the content distributing device and the load of the content distributing device.

**[0558]** Further, a content playing device for transmitting a meta data request to the content distributing device, receiving the meta data in response to the request, and acquiring the content in accordance with the meta data thus received, the content playing device in the present invention includes content acquiring means for acquiring the content from another content distributing device indicated by an address included in any one of the plurality of pieces of content-storage-location information contained in the content storage server list, when said content playing device receives the content storage server list included in the meta data thus received, or when said content playing device transmits a content-storage-location information request with use of a storage-location address included in the meta data thus received and receives, in response to the request, the content storage server list; and acquiring location changing means for, in a case where a receiving speed required to acquire the content of the content acquiring means is slower than a predetermined receiving speed, changing the another content distributing device, from which the content is to be acquired, to a still another content distributing device indicated by another address, which is different from the address, included in the content storage server list.

**[0559]** According to the arrangement, when the content playing device receives the content storage server list, the content acquiring means acquires, on the basis of the meta data thus received, the content from another content distributing device indicated by an address included in one of pieces of content-storage-location information included in the content storage server list thus received. In a case where a receiving speed required for the content acquiring means to acquire the content is

slower than the predetermined receiving speed, the acquiring location changing means changes the another content distributing device, from which the content is to be acquired, to a still another content distributing device indicated by another address, which is different from the address, included in the content storage server list.

**[0560]** The content playing device can, therefore, keep a receiving speed required to acquire content faster than a predetermined receiving speed, and can stably acquire the content without causing a huge delay.

**[0561]** The delay in acquiring of content may be caused by, for example, the following reason: an increase in throughput of a content distributing device from which the content is acquired; or a deterioration in communication status of a network between the content playing device and the content distributing device from which the content is acquired. The content playing device changes, due to the delay, the content distributing device, from which the content is acquired, to another content distributing device which is different from the content distributing device, so that it is possible to effectively use resources for (i) the content distributing device from which the content is acquired and (ii) a network between a content playing device for acquiring the content and the content distributing device from which the content is acquired.

**[0562]** Further, a content distributing system in accordance with the present invention includes the content distributing device, a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device, the content playing device for transmitting, to the content distributing device, a content request to request the content so as to acquire the content from the content distributing device, and for transmitting, to the content distributing device, a meta data request to request meta data of the content, receiving the meta data from the content distributing device, and acquiring, on the basis of the meta data thus received, the content corresponding to the meta data.

**[0563]** According to the arrangement, the content distributing system has an effect same as that of the content distributing device.

**[0564]** Note that the content distributing device may be achieved by a computer. In this case, the present invention encompasses a controlling program for realizing the content distributing device with use of computer by operating the computer as each means of the content distributing device and a computer readable recording medium in which the controlling program is stored.

[Supplementary description]

**[0565]** The present invention is not limited to the description of the embodiments above, and can be modified in numerous ways by a skilled person as long as such modification falls within the scope of the claims. An embodiment derived from a proper combination of technical means disclosed in different embodiments is also en-

comprised in the technical scope of the present invention.

**[0566]** Each block of the server 2, the proxy 3, and the client 4, especially, the server controlling section 14, the proxy controlling section 22, and the client control section 32 can be configured by means of hardware logic or can alternatively be realized by software with use of a CPU (Central Processing Unit) as follows.

**[0567]** Specifically, the server 2, the proxy 3, and the client 4 each include: a CPU, such as an MPU, for executing commands of a program to realize each function; a ROM (Read Only Memory) which stores the program; a RAM (Random Access Memory) in which the program is developed in an executable format; and a storage device (recording medium), such as a memory, which stores the program and various kinds of data. Further, the object of the present invention can be also realized in such a manner that: recording media are provided to the server 2, the proxy 3, and the client 4, respectively, which recording media have stored program codes (execution mode program, intermediate code program, and source program) (serving as software for realizing the aforementioned functions) of control programs of the server 2, the proxy 3, and the client 4 so as to be readable by a computer; and the program codes stored in the recording medium are read out and carried out by the computer (or CPU or MPU).

**[0568]** Examples of the recording medium encompass: tapes such as a magnetic tape and a cassette tape; disks such as magnetic disks (e.g., a floppy (registered trademark) disk and a hard disk) and optical disks (e.g., a CD-ROM, an MO, an MD, a DVD, and a CD-R); cards such as an IC card (including a memory card) and an optical card; and semiconductor memories (e.g., a mask ROM, an EPROM, an EEPROM, and a flash ROM).

**[0569]** Further, the server 2, the proxy 3, and the client 4 may be configured to be connected to a communication network, and the program code may be supplied via the communication network. The communication network is not particularly limited, and examples of the communication network encompass the Internet, an intranet, an extranet, a LAN, an ISDN, a VAN, a CATV communication network, a virtual private network, a telephone network, a mobile communication network, and a satellite communication network. In addition, a transmission medium constituting the communication network is not particularly limited, and examples of the transmission medium encompass: wired transmission media such as IEEE1394, a USB, a power-line carrier, a cable TV line, a telephone line, and an ADSL; and wireless transmission media such as infrared rays (e.g., IrDA and a remote controller), Bluetooth (registered trademark), 802.11 wireless, an HDR, a cell-phone network, and a satellite line, and a digital terrestrial network. Note that the present invention may be also realized by a computer data signal which has the program codes specified with electronic transmission and is embedded in a carrier wave.

#### Industrial Applicability

**[0570]** The present invention can be applied to a content distributing system for distributing a content to a client from a server in response to a request transmitted from the client for playing the content, and a content distributing device, a relaying device, and a content playing device for configuring the content distributing system.

#### 10 Reference Signs List

#### [0571]

1, 1a, 1b, 1c content distributing system  
 2 server (content distributing device)  
 3 proxy (relaying device)  
 4 client (content playing device, relaying device)  
 15 response executing section (determining section, content transmitting means, content-acquiring-location instructing means, transmission record creating means, request determining section, transmitting means)  
 20 16 relaying device specifying section (content-acquiring-location instructing means, distance calculating means)  
 25 17 content-storage-location information generating section (content-storage-location information generating means)  
 18 acquiring location specifying section (content-storage-location information acquiring means, content-storage-location information generating means, update determining means)  
 30 35 response/request executing section (content acquiring means)  
 35 37 relaying device selecting section (relaying device changing means)  
 38 acquiring location specifying section (acquiring location changing means)

#### 40 Claims

1. A content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request,  
 45 the content distributing device comprising:

determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested;  
 50 content transmitting means for transmitting, in response to the request, the content thus requested to the relaying device in a case where the determining means determines that the source is a relaying device;

- content-storage-location information generating means for generating content-storage-location information by associating (A) content identification information for specifying the content transmitted by the content transmitting means with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and  
 content-acquiring-location instructing means for transmitting, in response to the request, an instruction to the content playing device which is the source in a case where the determining means determines that the source is a content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address that the content-storage-location information associates with the content identification information indicative of the content thus requested or (ii) a content playing device indicated by an address that the content-storage-location information associates with the content identification information indicative of the content thus requested.
2. The content distributing device as set forth in claim 1, wherein  
 the determining means determines that, in a case where the request contains transmission path information indicative of a transmission path via which the request is transferred, the source is a relaying device and, in a case where the request does not contain the transmission path information, the source is a content playing device.
3. The content distributing device as set forth in claim 1 or 2, wherein  
 in a case where there are a plurality of pieces of the content-storage-location information which include a plurality of addresses, respectively, each of the plurality of addresses being associated with the content identification information indicative of the content thus requested, the content-acquiring-location instructing means (A) creates a content-storage-location list including the plurality of addresses included in the plurality of pieces of content-storage-location information and (B) transmit, to the content playing device which is the source, an instruction to acquire the content from (I) a relaying device indicated by an address included in the content-storage-location list thus created or (II) a content playing device indicated by an address included in the content-storage-location list thus created.
4. The content distributing device as set forth in claim 3, wherein:
- the content-storage-location information generating means generates the content-storage-location information by associating (A) the content identification information indicative of the content which has been transmitted by the content transmitting means with (B) date and time when the content transmitting means has transmitted the content; and  
 the content-acquiring-location instructing means creates the content-storage-location list by (I) arranging the plurality of addresses, which are contained in the plurality of pieces of content-storage-location information, on the basis of date and time associated with the content identification information and (II) adding priorities to the plurality of addresses so that an address having later date and time gets a higher priority.
5. A content distributing device as set forth in claim 3, further comprising:  
 distance calculating means for calculating, on the basis of an address included in any one of the plurality of pieces of content-storage-location information, a physical or network-structural distance between (A) a relaying device or a content playing device which is indicated by the address and (B) the content playing device that the content-acquiring-location instructing means instructs on a device from which the content is acquired, wherein:  
 the content-acquiring-location instructing means creates the content-storage-location list by (I) arranging the plurality of addresses, which are included in the respective plurality of pieces of content-storage-location information, on the basis of distances calculated by the distance calculating means, and (II) adding priorities to the plurality of addresses so that an address having a shorter distance gets a higher priority.
6. A content distributing device as set forth in any one of claims 1 through 5, further comprising:  
 transmission record creating means for creating a response transmission record by associating (A) a destination to which a response is transmitted in response to the request with (B) date and time when the response has been transmitted, wherein, with reference to response transmission record created by the transmission record creating means, the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to

acquire the content from a relaying device or a content playing device which (i) is indicated by an address that the content-storage-location information associates with the content thus requested and (ii) is not included in the response transmission record within a predetermined time period.

7. A content playing device for (A) transmitting a request to a content distributing device recited in claim 3, (B) receiving the content-storage-location list in response to the request, and (C) acquiring the content thus requested from a relaying device or a content playing device which is indicated by an address included in the content-storage-location list thus received,

wherein, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, the relaying device or the content playing device, from which the content is acquired, is changed to a relaying device or a content playing device which is indicated by another address included in the content-storage-location list.

8. A content playing device for (A) transmitting a request to a content distributing device recited in claim 4 or 5, (B) receiving the content-storage-location list in response to the request, and (C) acquiring a requested content from a relaying device or a content playing device indicated by an address which is the highest on the content-storage-location list thus received,

the content playing device comprising relaying device changing means for, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, changing a relaying device or a content playing device, from which the requested content is to be acquired, to a relaying device or a content playing device indicated by an address which is the second highest on the content-storage-location list.

9. A content distributing system, comprising:

a content distributing device recited in any one of claims 1 through 6;

a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device; and

a content playing device for requesting the content distributing device to transmit a content and acquiring the content thus requested from a device designated by the content distributing device.

10. A method for controlling content distributing device

for transmitting, in response to a request, a content to a source which is a sender of the request, the method comprising:

a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested;

a content transmitting step of transmitting, in response to the request, the content thus requested to the relaying device in a case where it is determined that, in the determining step, the source is a relaying device;

a content-storage-location information generating step of generating content-storage-location information by associating (A) content identification information for indicating the content transmitted in the content transmitting step with (B) an address of the relaying device, which is a destination to which the content is to be transmitted, or an address of the content playing device, to which the content is to be transferred from the relaying device; and

a content-acquiring-location instructing step of transmitting, in response to the request, an instruction to the content playing device which is the source in a case where it is determined that, in the content-storage-location information generating step, the source is the content playing device, which instruction is to acquire the content from (i) a relaying device indicated by an address associated, in the content-storage-location information, with the content identification information indicative of the content thus requested or (ii) a content playing device indicated by an address associated, in the content-storage-location information, with the content identification information indicative of the content thus requested.

11. A content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request,

the content distributing device comprising:

determining means for determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested;

content-storage-location information acquiring means for acquiring, in response to the request, an address of another content distributing device possessing the content thus requested, among predetermined other content distributing

- devices, in a case where the determining means determines that the source is a content playing device; and  
 content-acquiring-location instructing means for transmitting, to the content playing device which is the source, an instruction to acquire the content from the another content distributing device which is indicated by the address acquired by the content-storage-location information acquiring means.
12. The content distributing device as set forth in claim 11, wherein  
 the content-storage-location information acquiring means transmits, to the predetermined other content distributing devices, an inquiry as to whether or not the predetermined other content distributing devices include the content thus requested, so as to acquire the address of the another content distributing device that has responded, to the inquiry, that the content distributing device possesses the content thus requested.
13. A content distributing device as set forth in claim 12, further comprising  
 content-storage-location information generating means for (A) generating content-storage-location information by associating (i) the address of the another content distributing device including the content, which address has been acquired by the content-storage-location information acquiring means, with (ii) the content identification information indicative of the content and (B) causing a storage section to store the content-storage-location information, wherein  
 the content-storage-location information acquiring means (I) reads out the content-storage-location information from the storage section, and (II) acquires the address, associated with the content identification information, from the content-storage-location information in a case where the content-storage-location information thus read out contains the content identification information indicative of the content thus requested, or transmits the inquiry to thereby acquire the address of the another content distributing device possessing the content thus requested in a case where the content-storage-location information thus read out does not contain the content identification information indicative of the content thus requested.
14. The content distributing device as set forth in any one of claims 11 through 13, wherein:  
 the content-storage-location information acquiring means (i) acquires a plurality of addresses included in a plurality of content distributing devices, respectively, each of the plurality of content distributing devices including the content thus requested, and (ii) creates a content storage server list including the plurality of addresses thus acquired and content identification information indicative of the content; and  
 the content-acquiring-location instructing means transmits, to the content playing device which is the source, an instruction to acquire the content from the another content distributing device indicated by the address included in the content storage server list created by the content-storage-location information acquiring means.
15. A content playing device for (A) transmitting a request to a content distributing device recited in claim 14, (B) receiving the content storage server list in response to the request, and (C) acquiring a requested content from another content distributing device indicated by one of a plurality of addresses included in the content storage server list thus received, the content playing device comprising  
 acquiring location changing means for, in a case where a receiving speed required to acquire the content is slower than a predetermined receiving speed, changing the another content distributing device, from which the requested content is to be acquired, to another content distributing device indicated by another address, which is different from the one of the plurality of addresses, included in the content storage server list.
16. A content distributing system, comprising:  
 a content distributing device recited in any one of claims 11 through 14,  
 a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device; and  
 the content playing device for requesting the content distributing device to transmit the content, and acquiring the content thus requested from a device designated by the content distributing device.
17. A method for controlling a content distributing device for transmitting, in response to a request, a content to a source which is a sender of the request, the method comprising:  
 a determining step of determining whether the source is (A) a relaying device for receiving the content thus requested and possessing and transferring the content to a content playing device or (B) the content playing device for playing the content thus requested;

- a content-storage-location information acquiring step of acquiring, in response to the request, an address of another content distributing device including the content thus requested, among other content distributing devices connected to the content distributing device, in a case where it is determined that, in the determining step, the source is the content playing device; and
- a content-acquiring-location instructing step of transmitting, to the content playing device which is the source, an instruction to acquire the content from the another content distributing device which is indicated by the address acquired in the content-storage-location information acquiring step.
18. A content distributing device for transmitting, in response to a request, data to a source which has transmitted the request, the content distributing device comprising:
- managing means for managing (A) content and (B) meta data of the content, the meta data containing (i) content-storage-location information in which content identification information for specifying the content and an address of another content distributing device including the content are associated with each other or (ii) a storage-location address indicative of a location of the content-storage-location information;
- request determining means for determining whether the request is a content request or a meta data request; and
- transmitting means for transmitting the content thus requested to the source in a case where the request determining means determines that the request is the content request, and for transmitting the meta data thus requested to the source in a case where the request determining means determines that the request is the meta data request.
19. The content distributing device as set forth in claim 18, wherein:
- the request determining means determines whether the request is the content request, the meta data request, or a content-storage-location information request including the storage-location address; and
- in a case where the request determining means determines that the request is the content-storage-location information request including the storage-location address, the transmitting means transmits, to a device which is the source, the content-storage-location information whose location is indicated by the storage-location address.
20. A content distributing device as set forth in claim 19, further comprising:
- content-storage-location information acquiring means for transmitting, to predetermined other content distributing devices, an inquiry as to whether or not the predetermined other content distributing devices includes a predetermined content, and acquiring an address of another content distributing device that has responded, to the inquiry, that the another content distributing device includes the predetermined content; content-storage-location information generating means for (A) generating content-storage-location information by associating (i) the address of the another content distributing device including the predetermined content, which address has been acquired by the content-storage-location information acquiring means, with (ii) the content identification information for specifying the predetermined content and (B) causing a storage section to store the content-storage-location information; and
- update determining means for determining whether to update the content-storage-location information stored in the storage section, wherein
- in a case where the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines to update the content-storage-location information whose location is indicated by the storage-location address, the content-storage-location information acquiring means transmits the inquiry to acquire the address, the content-storage-location information generating means generates the content-storage-location information based on the address, and the transmitting means transmits the content-storage-location information to a device which is the source, or, in a case where the request determining means determines that the request is the content-storage-location information request including the storage-location address and the update determining means determines not to update the content-storage-location information whose location is indicated by the storage-location address, the transmitting means transmits, to the device which is the source, the content-storage-location information whose location is indicated by the storage-location address.
21. The content distributing device as set forth in claim 20, wherein
- the meta data can include a plurality of storage location addresses each indicative of a location of con-

tent-storage-location information which is set per unit into which the content is divided at a predetermined time interval.

22. The content distributing device as set forth in claim 21, wherein:

the content includes a plurality of media segments; and  
the content divided at a predetermined time interval includes at least one media segment.

23. The content distributing device as set forth in any one of claims 20 to 22, wherein  
the meta data includes a content storage server list containing the plurality of pieces of content-storage-location information or a storage location address indicative of a location of the content storage server list.

24. A content playing device for transmitting a meta data request to a content distributing device recited in claim 23, receiving the meta data in response to the request, and acquiring the content in accordance with the meta data thus received,  
the content playing device comprising  
content acquiring means for acquiring the content from another content distributing device indicated by an address included in any one of the plurality of pieces of content-storage-location information contained in the content storage server list, when said content playing device receives the content storage server list included in the meta data thus received, or when said content playing device transmits a content-storage-location information request with use of a storage-location address included in the meta data thus received and receives, in response to the request, the content storage server list; and  
acquiring location changing means for, in a case where a receiving speed required to acquire the content of the content acquiring means is slower than a predetermined receiving speed, changing the another content distributing device, from which the content is to be acquired, to a still another content distributing device indicated by another address, which is different from the address, included in the content storage server list.

25. A content distributing system, comprising:

a content distributing device recited in any one of claims 18 to 23,  
a relaying device for requesting the content distributing device to transmit a content, possessing the content thus requested, and transferring the content thus requested to a content playing device,  
the content playing device for transmitting, to

the content distributing device, a content request to request the content so as to acquire the content from the content distributing device, and for transmitting, to the content distributing device, a meta data request to request meta data of the content, receiving the meta data from the content distributing device, and acquiring, on the basis of the meta data thus received, the content corresponding to the meta data.

26. A method for controlling a content distributing device for transmitting, in response to a request, data to a source which has transmitted the request, the method managing (A) content and (B) meta data of the content, the meta data containing (i) content-storage-location information in which content identification information for specifying the content and an address of another content distributing device including the content are associated with each other or (ii) a storage-location address indicative of a location of the content-storage-location information,  
the method comprising  
a request determining step of determining whether the request is a content request or a meta data request; and  
a transmitting step of transmitting the content thus requested to the source in a case where it is determined that, in the request determining step, the request is the content request, and of transmitting the meta data thus requested to the source in a case where it is determined that, in the request determining step, the request is the meta data request.
27. A controlling program for causing a content distributing device recited in any one of claims 1 through 6, 11 through 14, and 18 through 23 to operate, the controlling program causing a computer to function as each means.
28. A computer readable recording medium in which a controlling program recited in claim 27 is recorded.

FIG. 1

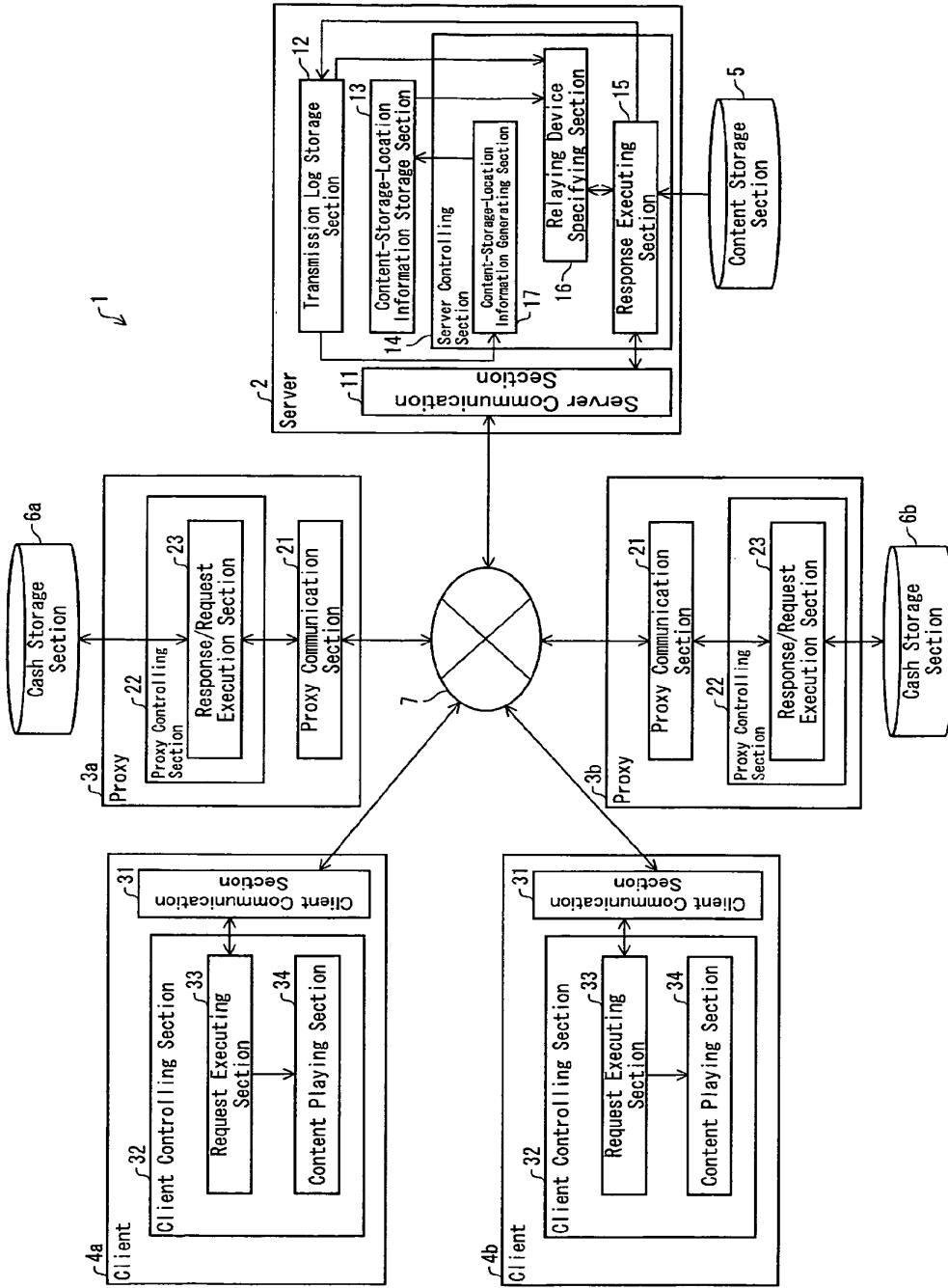




FIG. 2

Transmission Log

Date	Address Of Destination	Transmitted Contents	Content ID
Sun, 31 May 2013 13:52:22 GMT	http://example-client1.com	305 Use Proxy (proxy1)	content1
Sun, 31 May 2013 13:53:38 GMT	http://example-proxy1.com	200 OK	content1
Sun, 31 May 2013 15:02:11 GMT	http://example-client2.com	305 Use Proxy (proxy2)	content2
Sun, 31 May 2013 15:03:08 GMT	http://example-proxy2.com	200 OK	content2
Mon, 01 Jun 2013 08:04:06 GMT	http://example-client1.com	305 Use Proxy (proxy2)	content2
Mon, 01 Jun 2013 08:05:30 GMT	http://example-proxy2.com	304 Not Modified	content2
⋮	⋮	⋮	⋮

FIG. 3

Content-Storage-Location Information

Date	Content ID	Address Of Storage Location
Sun, 31 May 2013 13:53:38 GMT	content1	http://example-proxy1.com
Sun, 31 May 2013 15:03:08 GMT	content2	http://example-proxy2.com
Mon, 01 Jun 2013 08:05:30 GMT	content2	http://example-proxy2.com
⋮	⋮	⋮

FIG. 4

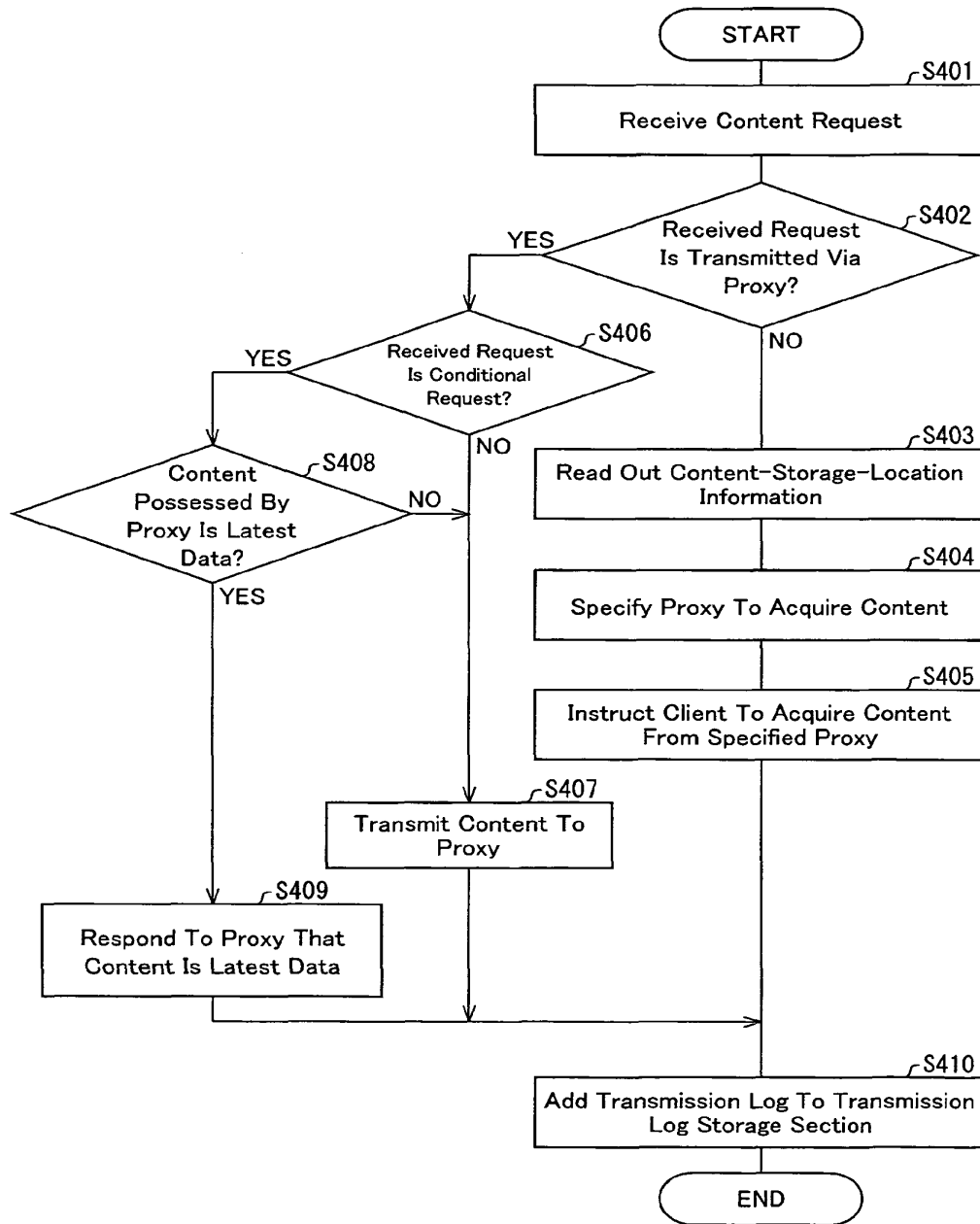


FIG. 5

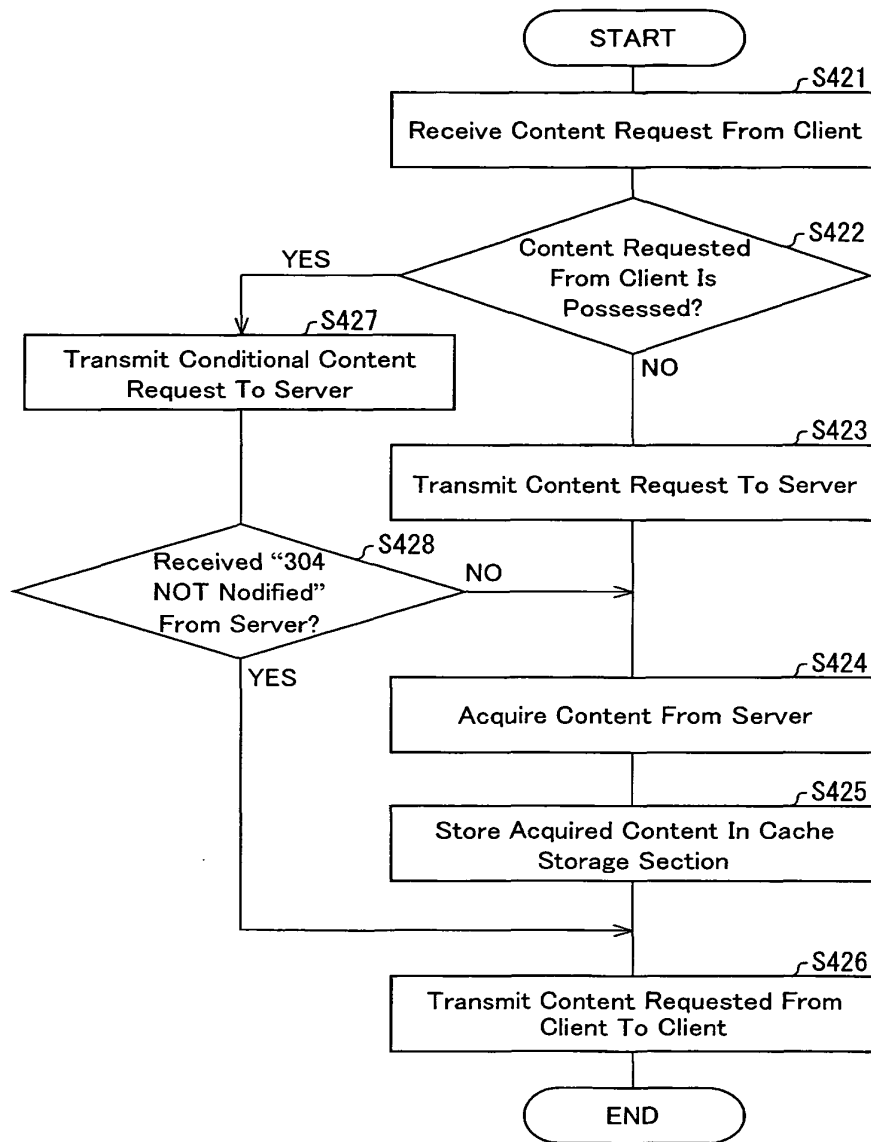


FIG. 6

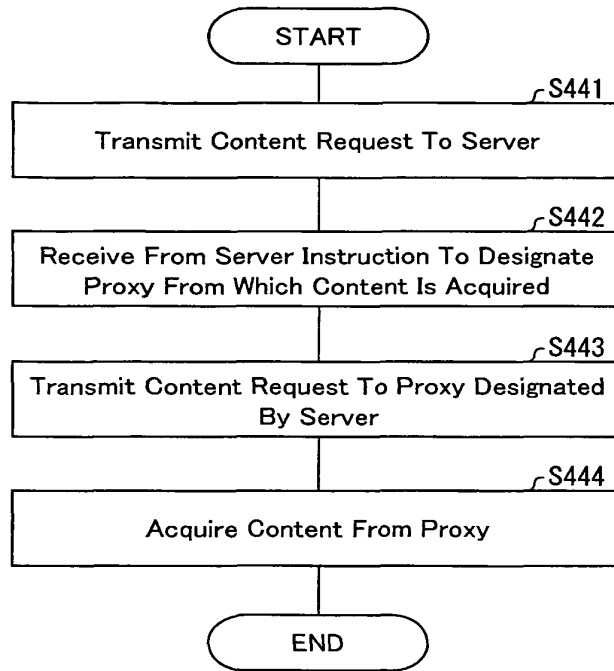


FIG. 7

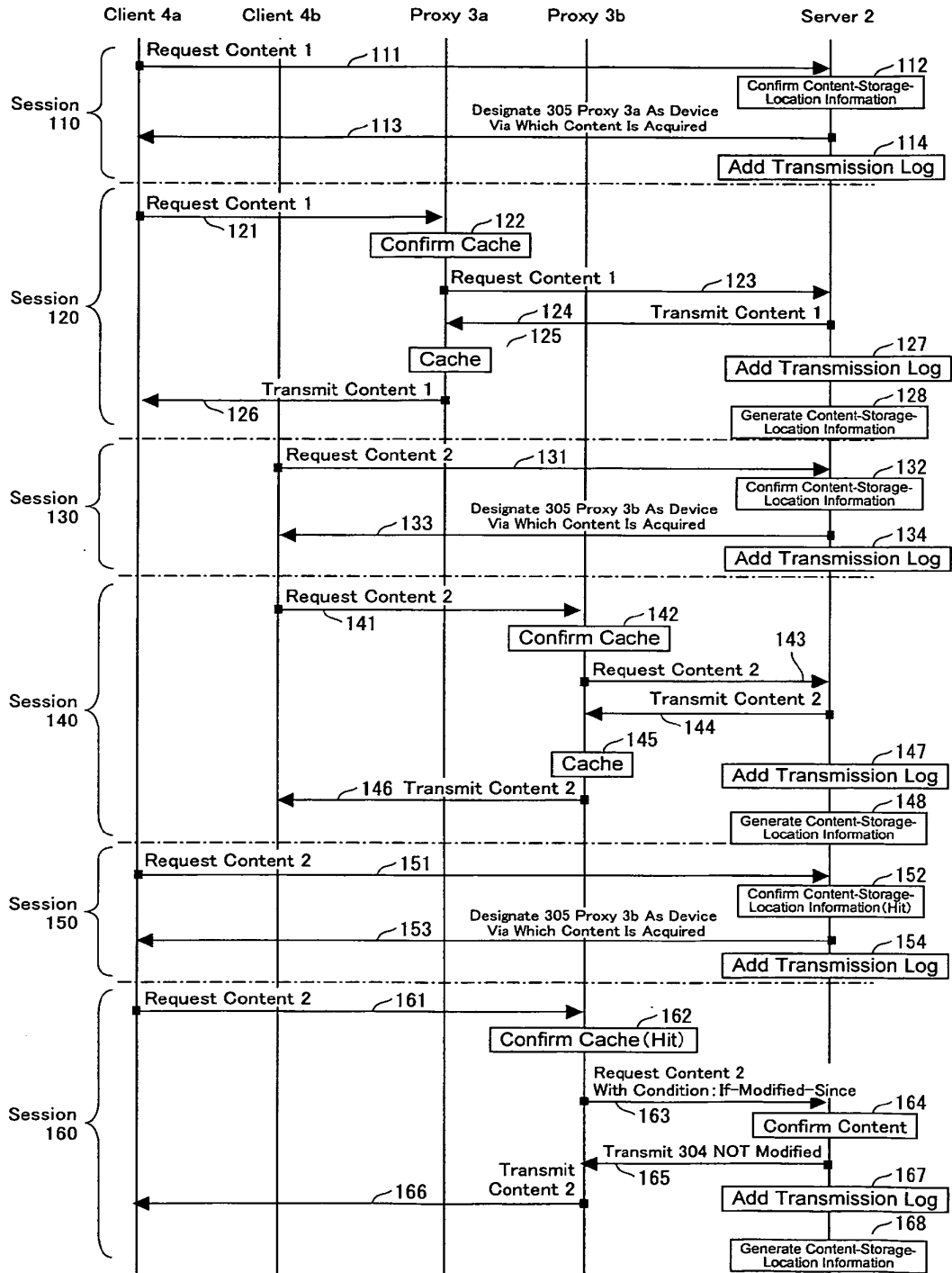


FIG. 8

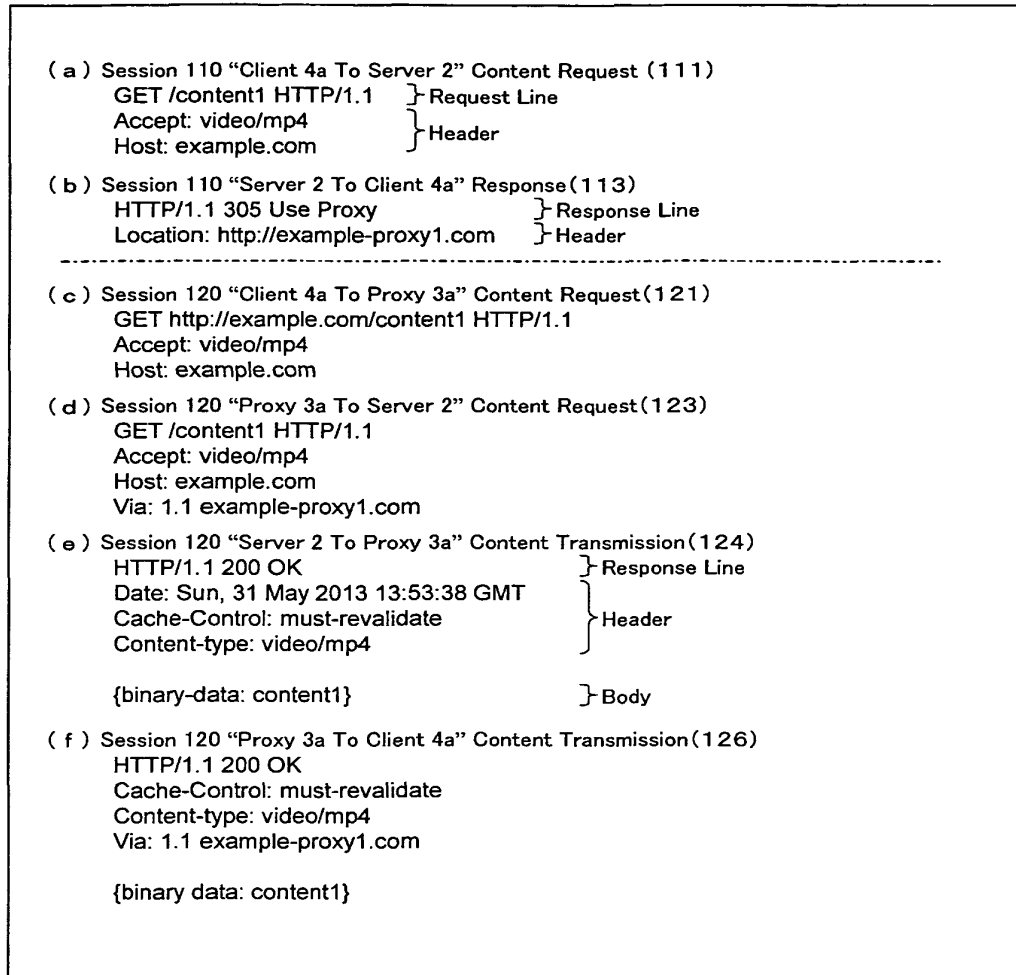


FIG. 9

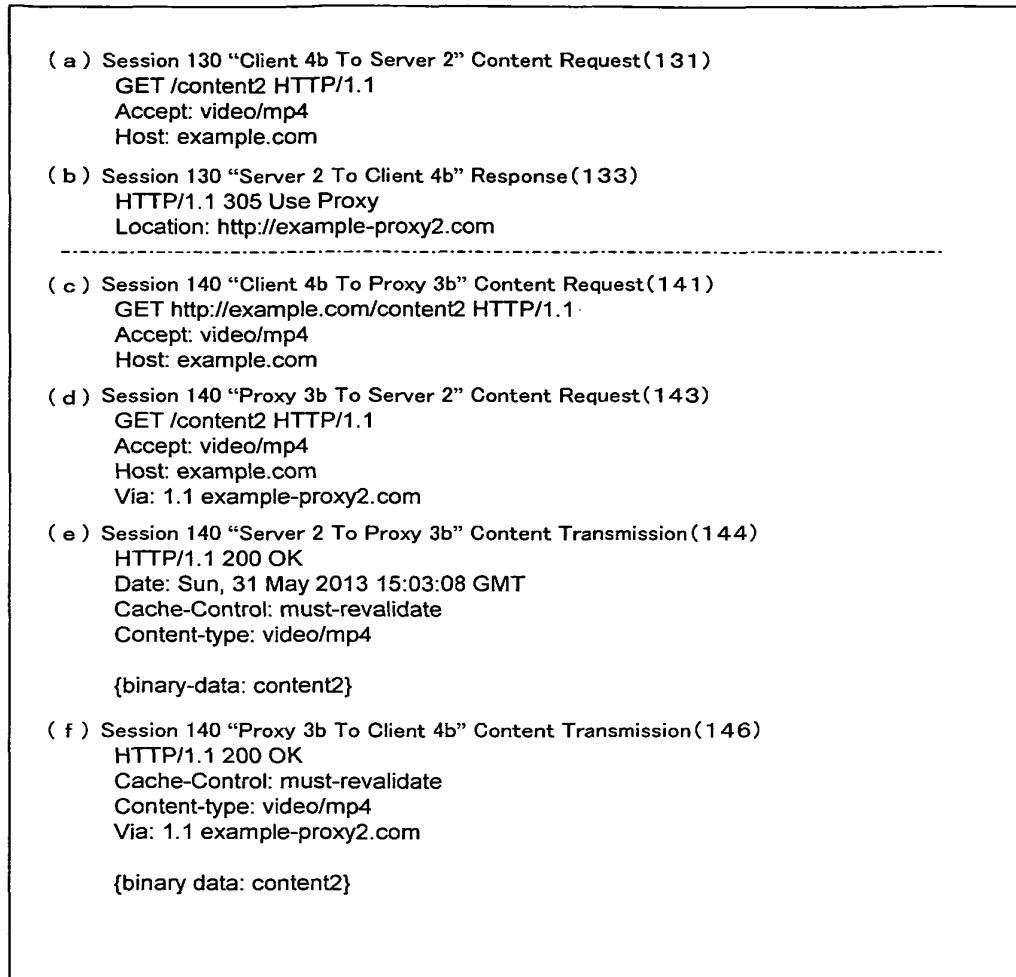




FIG. 10

( a ) Session 150 "Client 4a To Server 2" Content Request (151)  
GET /content2 HTTP/1.1  
Accept: video/mp4  
Host: example.com

( b ) Session 150 "Server 2 To Client 4a" Response (153)  
HTTP/1.1 305 Use Proxy  
Location: http://example-proxy2.com

---

( c ) Session 160 "Client 4a To Proxy 3b" Content Request (161)  
GET http://example.com/content2 HTTP/1.1  
Accept: video/mp4  
Host: example.com

( d ) Session 160 "Proxy 3b To Server 2" Conditional Content Request (163)  
GET /content2 HTTP/1.1  
If-Modified-Since Sun, 31 May 2013 15:03:08 GMT  
Accept: video/mp4  
Host: example.com  
Via: 1.1 example-proxy2.com

( e ) Session 160 "Server 2 To Proxy 3b" Response (165)  
HTTP/1.1 304 Not Modified  
Date: Mon, 01 Jun 2013 08:05:30 GMT

( f ) Session 160 "Proxy 3b To Client 4a" Content Transmission (166)  
HTTP/1.1 200 OK  
Cache-Control: must-revalidate  
Content-type: video/mp4  
Via: 1.1 example-proxy2.com

{binary data: content2}

FIG. 11

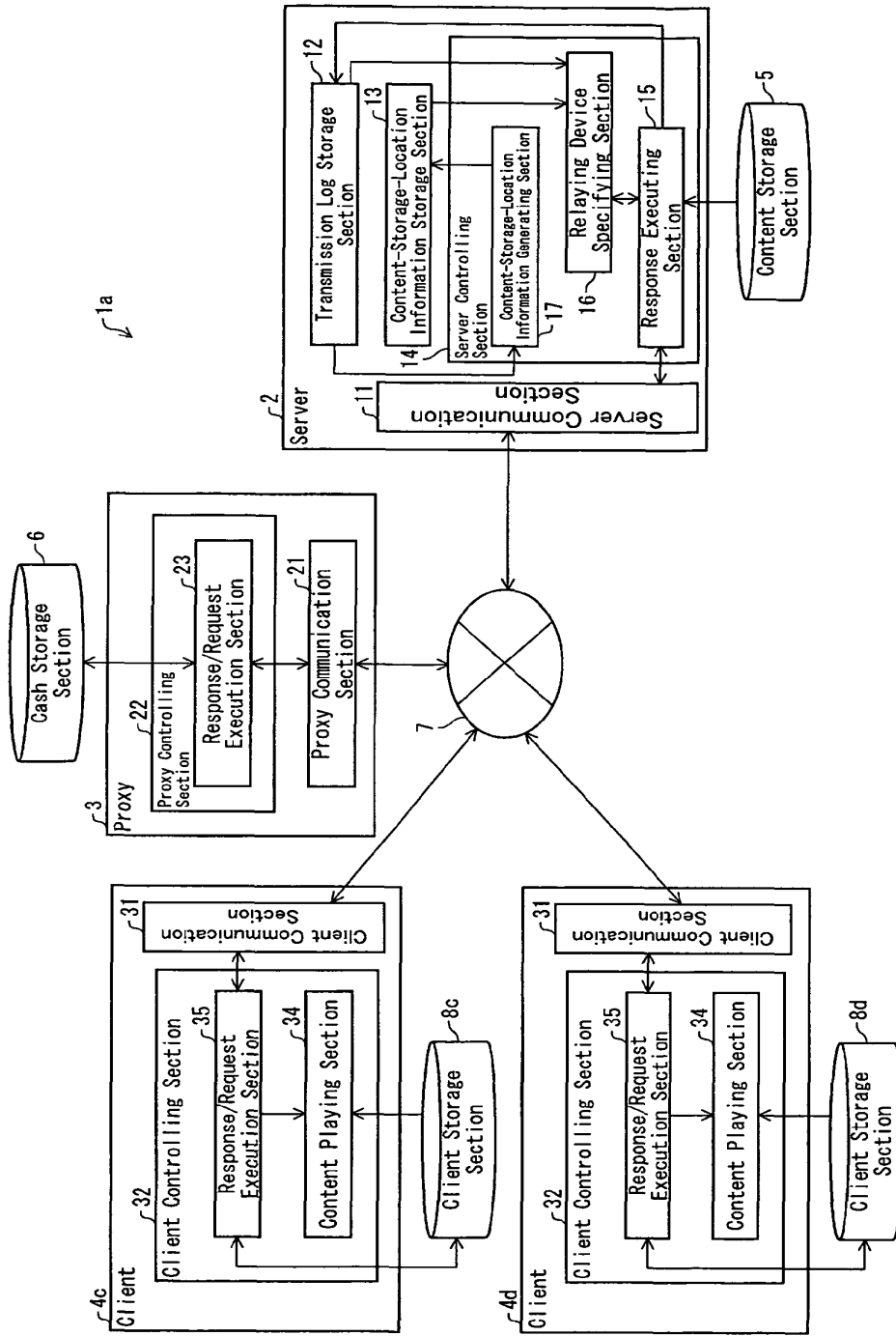


FIG. 12

Transmission Log

Date	Address Of Destination	Transmitted Contents	Content ID
41 Sun, 31 May 2013 13:52:22 GMT	http://example-client1.com	305 Use Proxy (proxy1)	content1
42 Sun, 31 May 2013 13:53:38 GMT	http://example-proxy1.com	200 OK	content1
43 Mon, 01 Jun 2013 08:04:10 GMT	http://example-client2.com	305 Use Proxy (client1)	content1
44 Mon, 01 Jun 2013 08:05:30 GMT	http://example-client1.com	304 Not Modified	content1
⋮	⋮	⋮	⋮

FIG. 13

Content-Storage-Location Information

Date	Content ID	Address Of Storage Location
Sun, 31 May 2013 13:53:38 GMT	content1	http://example-proxy1.com
Sun, 31 May 2013 13:53:38 GMT	content1	http://example-client1.com
Mon, 01 Jun 2013 08:05:30 GMT	content1	http://example-client1.com
Mon, 01 Jun 2013 08:05:30 GMT	content1	http://example-client2.com
⋮	⋮	⋮

45  
46  
47  
48

FIG. 14

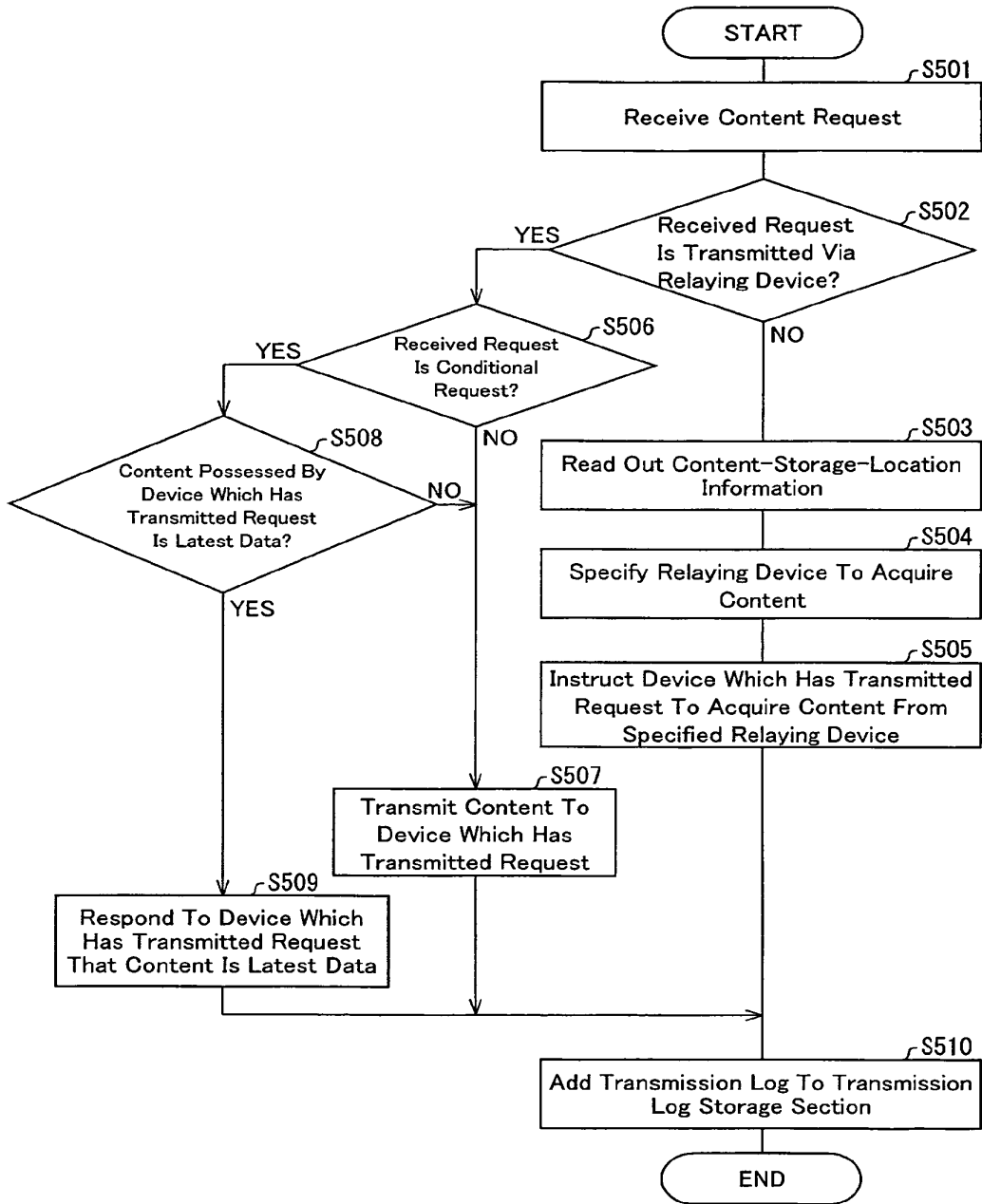


FIG. 15

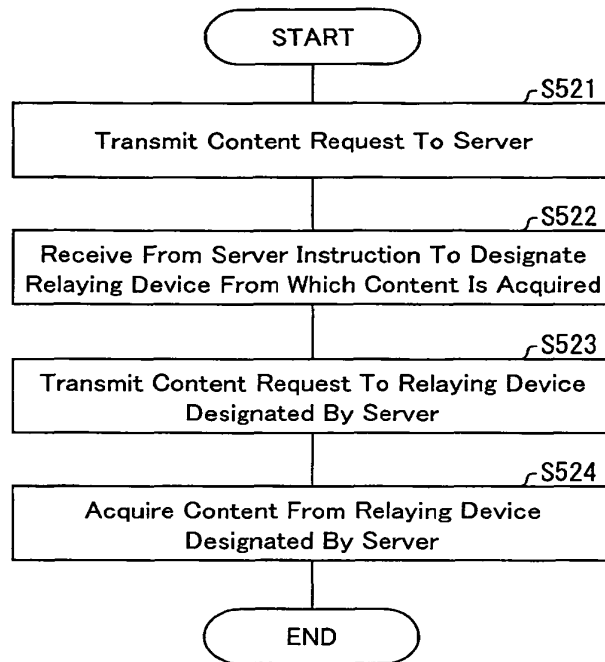


FIG. 16

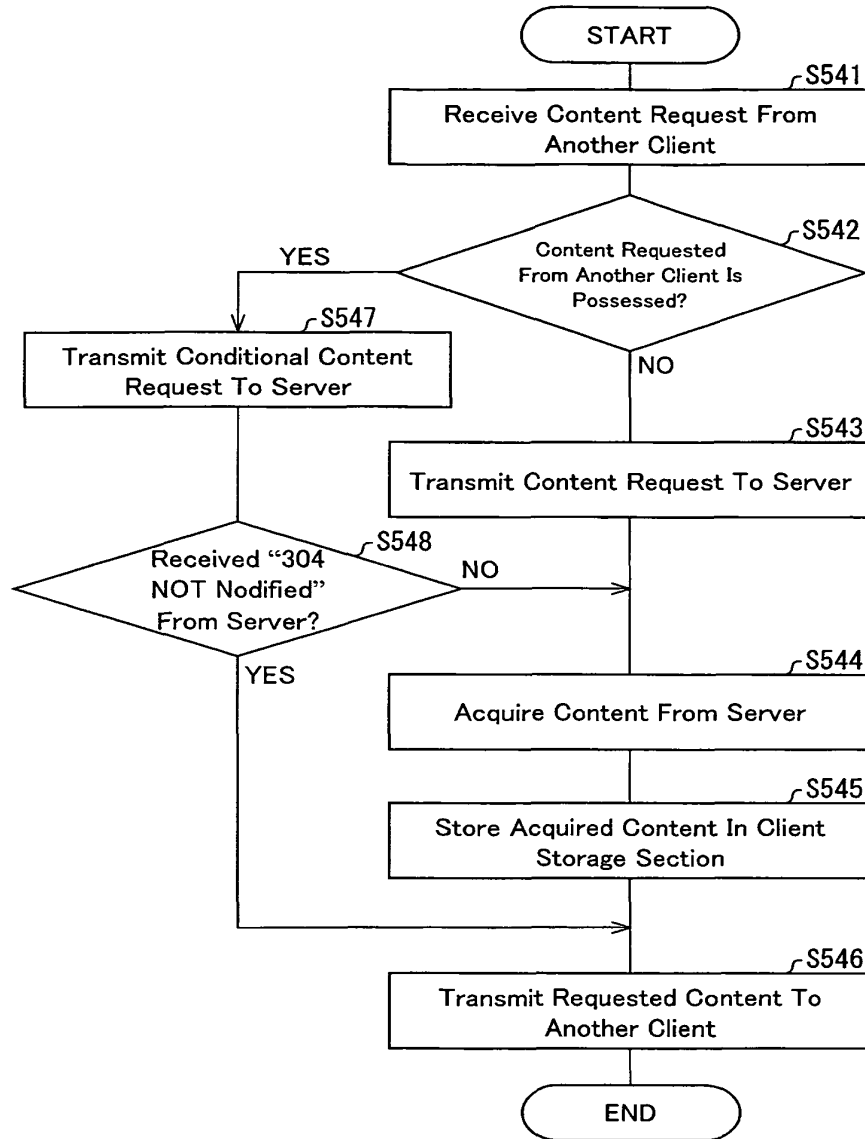


FIG. 17

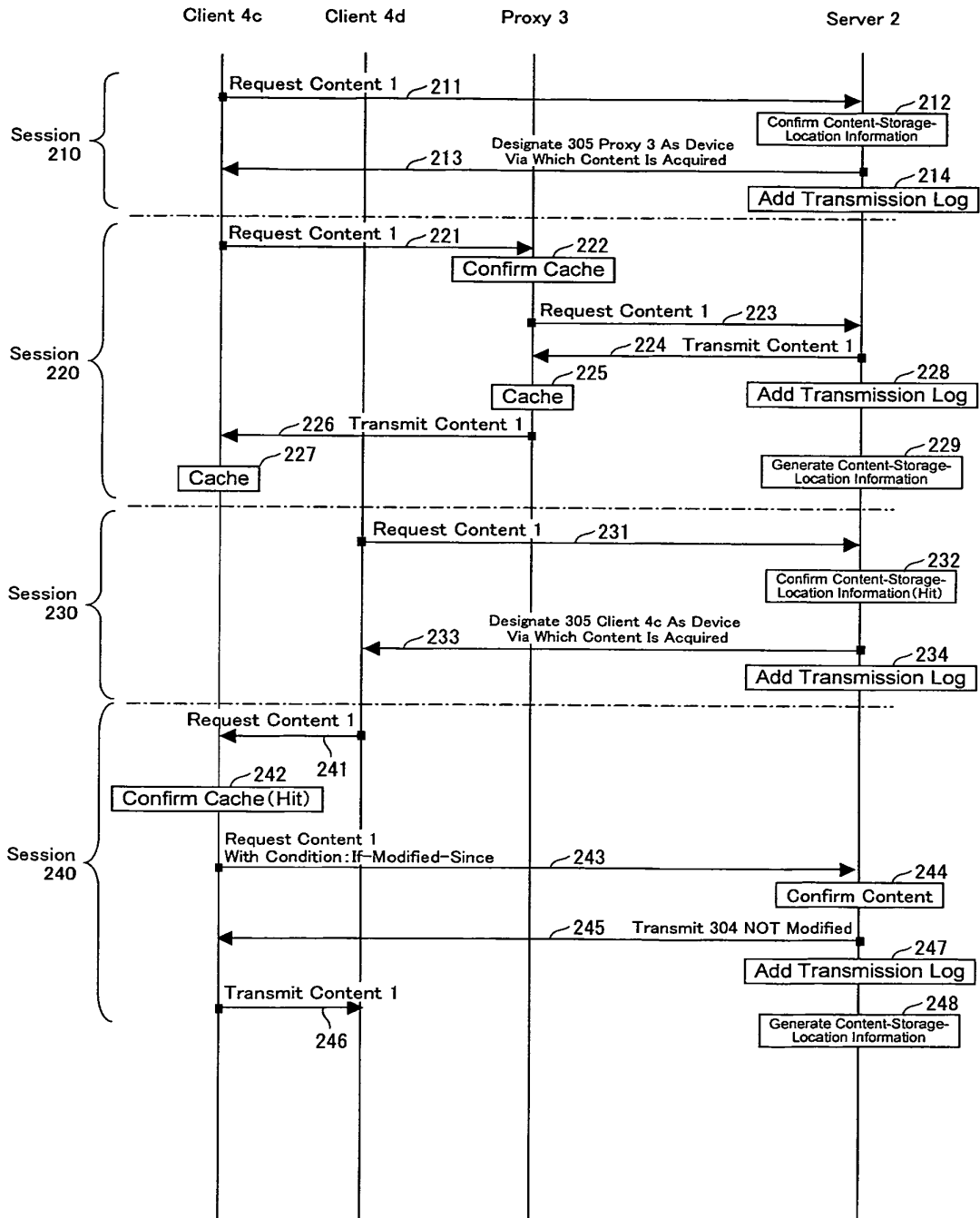




FIG. 18

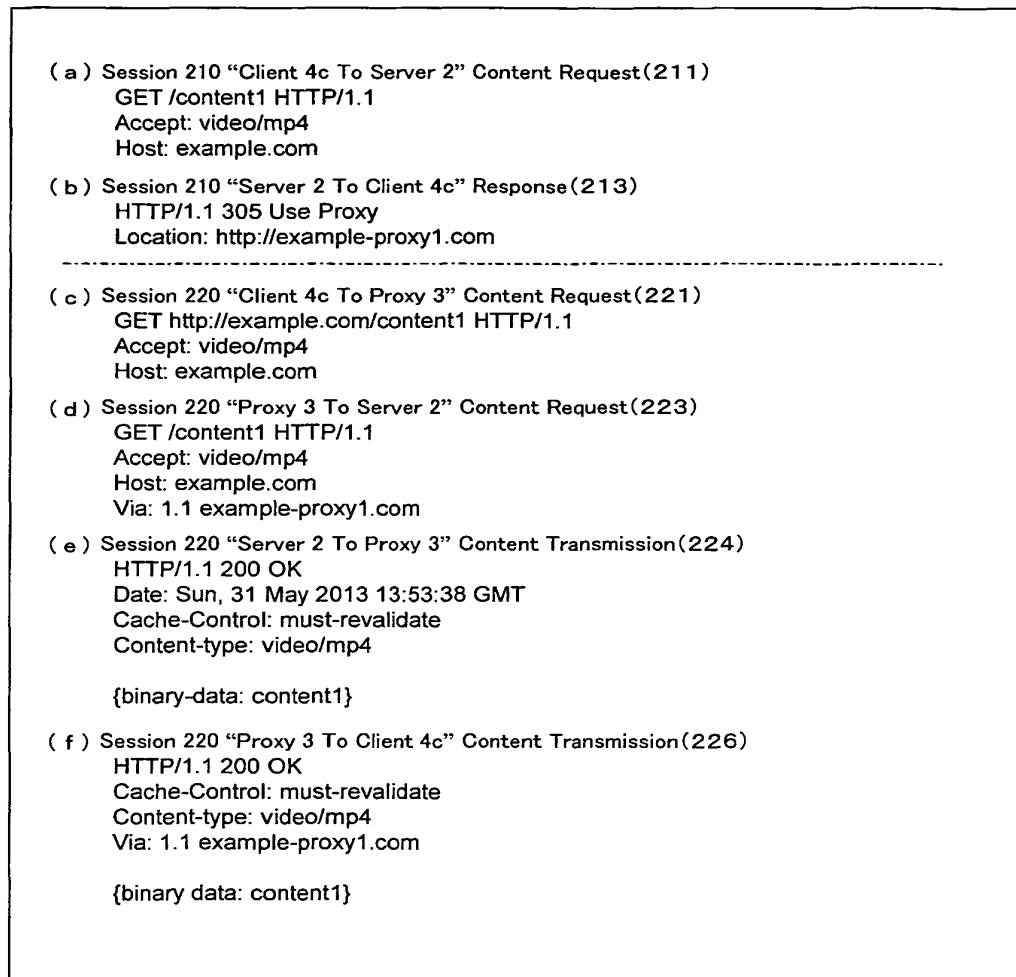


FIG. 19

( a ) Session 230 "Client 4d To Server 2" Content Request(231)  
GET /content1 HTTP/1.1  
Accept: video/mp4  
Host: example.com

( b ) Session 230 "Server 2 To Client 4d" Response(233)  
HTTP/1.1 305 Use Proxy  
Location: http://example-client1.com

---

( c ) Session 240 "Client 4d To Client 4c" Content Request(241)  
GET http://example.com/content1 HTTP/1.1  
Accept: video/mp4  
Host: example.com

( d ) Session 240 "Client 4c To Server 2" Conditional Content Request(243)  
GET /content1 HTTP/1.1  
If-Modified-Since Sun, 31 May 2013 13:53:38 GMT  
Accept: video/mp4  
Host: example.com  
Via: 1.1 example-client1.com

( e ) Session 240 "Server 2 To Client 4c" Response(245)  
HTTP/1.1 304 Not Modified  
Date: Mon, 01 Jun 2013 08:05:30 GMT

( f ) Session 240 "Client 4c To Client 4d" Content Transmission(246)  
HTTP/1.1 200 OK  
Cache-Control: must-revalidate  
Content-type: video/mp4  
Via: 1.1 example-client1.com

{binary data: content1}

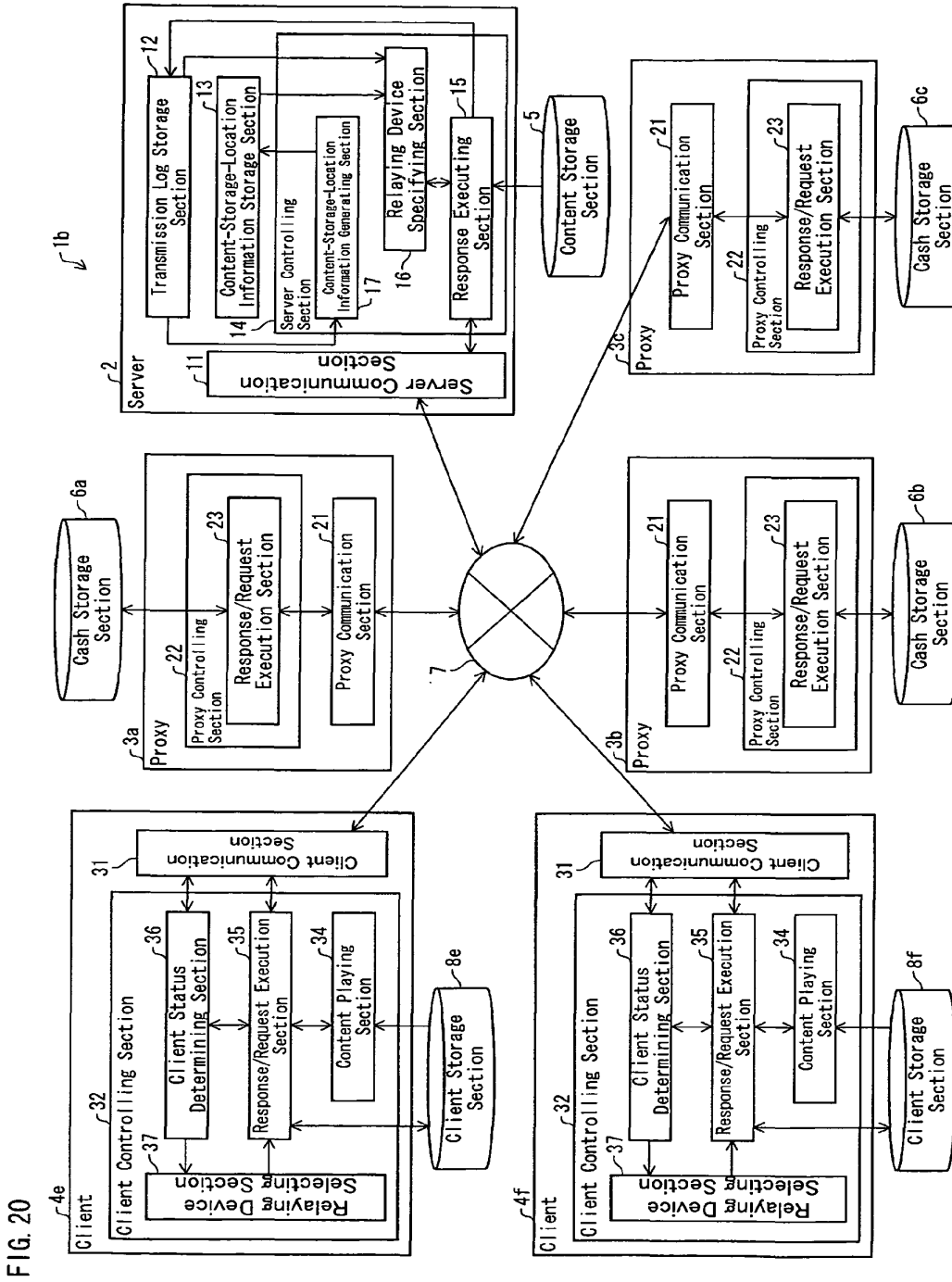


FIG. 20

FIG. 21

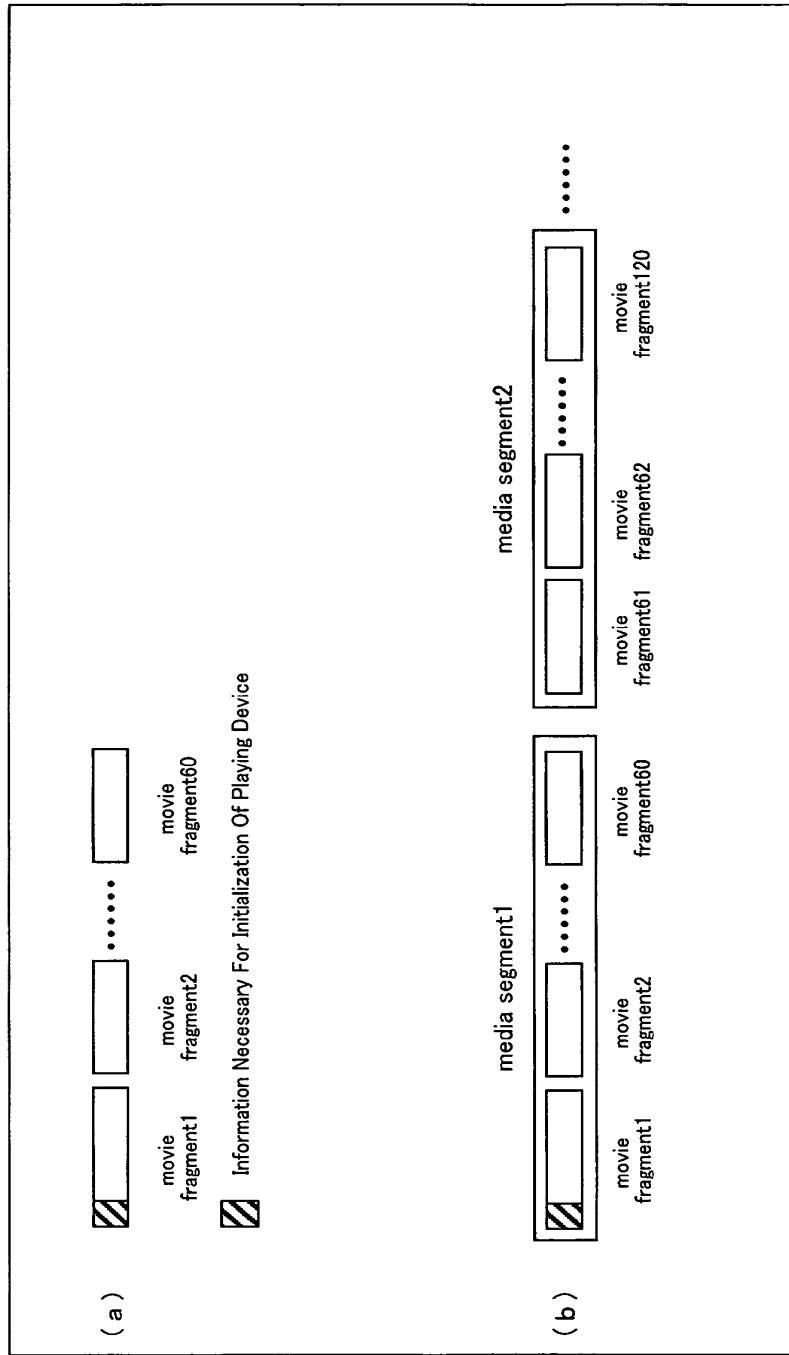


FIG. 22

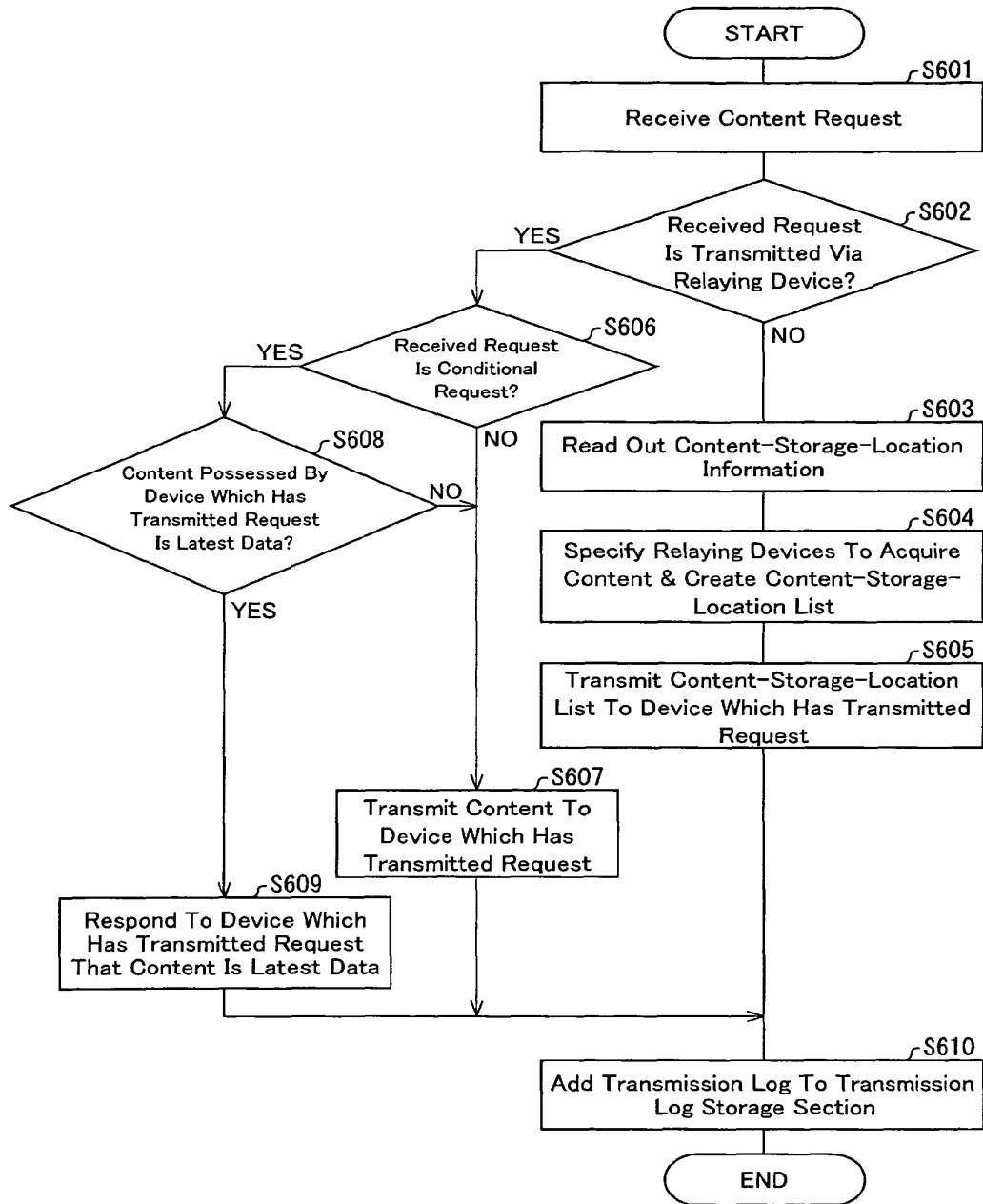


FIG. 23

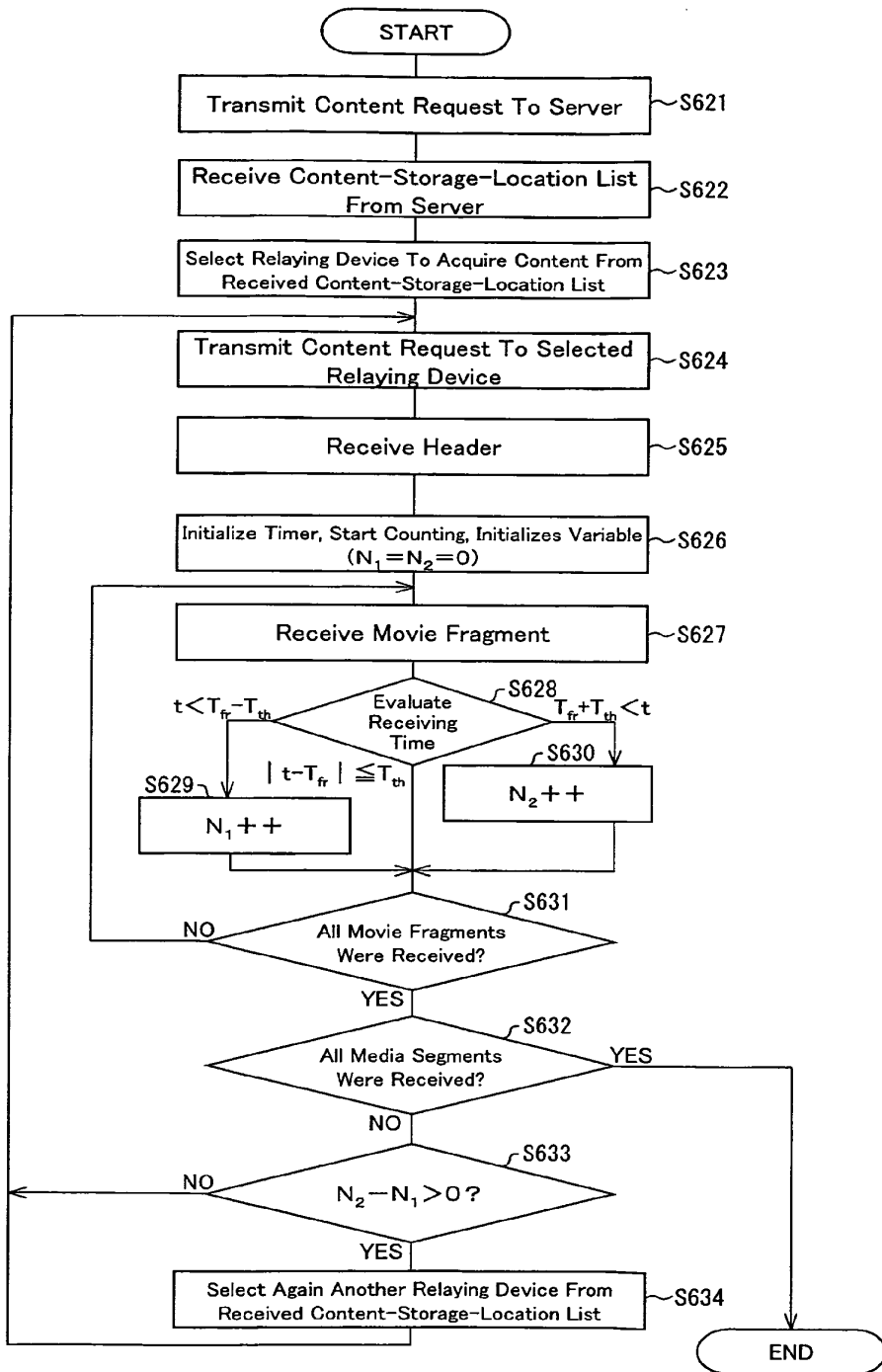


FIG. 24

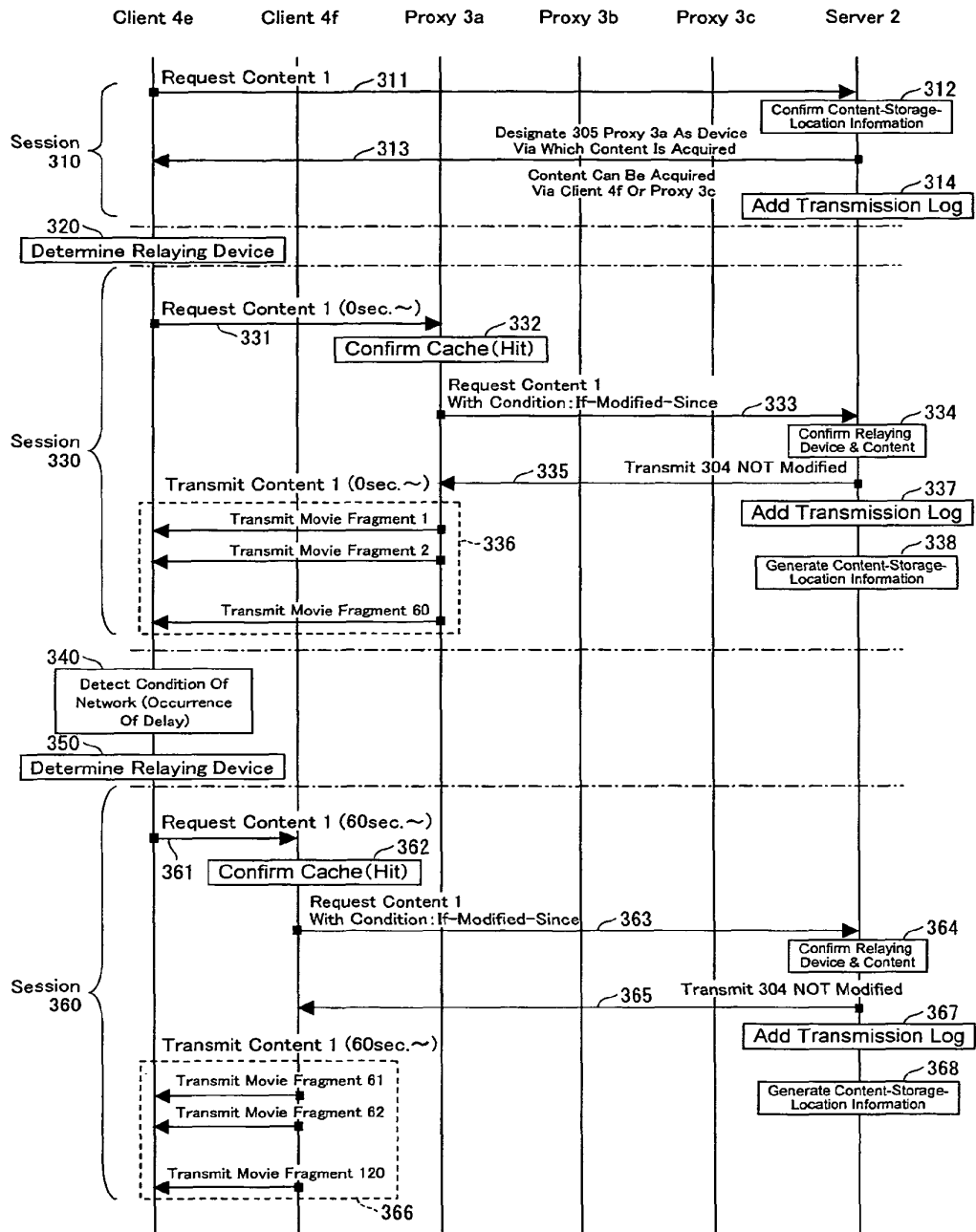


FIG. 25

Transmission Log

	Date	Address Of Destination	Transmitted Contents	Content ID
51	Thu, 28 May 2013 8:36:58 GMT	http://example-client2.com	305 Use Proxy (proxy3)	content1
52	Thu, 28 May 2013 8:38:05 GMT	http://example-proxy3.com	200 OK	content1
	⋮	⋮	⋮	⋮
53	Sat, 30 May 2013 15:35:58 GMT	http://example-proxy1.com	304 Not Modified	content1
	⋮	⋮	⋮	⋮
54	Sun, 31 May 2013 13:52:22 GMT	http://example-client1.com	305 Use Proxy (proxy1, client2, proxy3)	content1
55	Sun, 31 May 2013 13:53:38 GMT	http://example-proxy1.com	304 Not Modified	content1
56	Sun, 31 May 2013 13:55:03 GMT	http://example-client2.com	304 Not Modified	content1
	⋮	⋮	⋮	⋮



FIG. 26

Content-Storage-Location Information

	Date	Content ID	Address Of Storage Location
61	Thu, 28 May 2013 8:38:05 GMT	content1	http://example-proxy3.com
62	Thu, 28 May 2013 8:38:05 GMT	content1	http://example-client2.com
63	Sat, 30 May 2013 15:35:58 GMT	content1	http://example-proxy1.com
64	Sun, 31 May 2013 13:53:38 GMT	content1	http://example-proxy1.com
65	Sun, 31 May 2013 13:53:38 GMT	content1	http://example-client1.com
66	Sun, 31 May 2013 13:55:03 GMT	content1	http://example-client2.com
67	Sun, 31 May 2013 13:55:03 GMT	content1	http://example-client1.com
	⋮	⋮	⋮

FIG. 27

```

(a) Session 310 "Client 4e To Server 2" Content Request(311)
    GET /content1/0 HTTP/1.1
    Accept: video/mp4, multipart/media-segment
    Host: example.com

(b) Session 310 "Server 2 To Client 4e" Response(313)
    HTTP/1.1 305 Use Proxy
    Location: http://example-proxy1.com
    X-Alternative-Proxy-List: http://example-client2.com, http://example-proxy3.com
-----
(c) Session 330 "Client 4e To Proxy 3a" Content Request(331)
    GET http://example.com/content1/0 HTTP/1.1
    Accept: video/mp4, multipart/media-segment
    Host: example.com

(d) Session 330 "Proxy 3a To Server 2" Content Request(333)
    GET /content1/0 HTTP/1.1
    If-Modified-Since: Sat, 30 May 2013 15:35:58 GMT
    Accept: video/mp4, multipart/media-segment
    Host: example.com
    Via: 1.1 example-proxy1.com

(e) Session 330 "Server 2 To Proxy 3a" Response(335)
    HTTP/1.1 304 Not Modified
    Date: Sun, 31 May 2013 13:53:38 GMT

(f) Session 330 "Proxy 3a To Client 4e" Content Transmission(336)
    HTTP/1.1 200 OK
    Content-type: mutipart/media-segment; boundary=THIS_STRING_SEPARATES
    Content-Location: http://example.com/content1/0
    Cache-Control: must-revalidate
    Via: 1.1 example-proxy1.com
    X-Media-Segment-Index: 1/60

    --THIS_STRING_SEPARATES
    Content-type: video/mp4
    X-Timestamp: 0.0
    {binary-data: movie fragment 1}
    .
    .
    .
    --THIS_STRING_SEPARATES
    Content-type: video/mp4
    X-Timestamp: 59.0
    {binary-data: movie fragment 60}

    --THIS_STRING_SEPARATES--

```

FIG. 28

```

(a) Session 360 "Client 4e To Client 4f" Content Request(361)
  GET http://example.com/content1/1 HTTP/1.1
  Accept: video/mp4, multipart/media-segment
  Host: example.com

(b) Session 360 "Client 4f To Server 2" Content Request(363)
  GET /content1/1 HTTP/1.1
  If-Modified-Since: Thu, 28 May 2013 8:38:05 GMT
  Accept: video/mp4, multipart/media-segment
  Host: example.com
  Via: 1.1 example-client2.com

(c) Session 360 "Server 2 To Client 4f" Response(365)
  HTTP/1.1 304 Not Modified
  Date: Sun, 31 May 2013 13:55:03 GMT

(d) Session 360 "Client 4f To Client 4e" Content Transmission(366)
  HTTP/1.1 200 OK
  Content-type: multipart/media-segment; boundary=THIS_STRING_SEPARATES
  Content-Location: http://example.com/content1/1
  Cache-Control: must-revalidate
  Via: 1.1 example-client2.com
  X-Media-Segment-Index: 2/60

  --THIS_STRING_SEPARATES
  Content-type: video/mp4
  X-Timestamp: 60.0
  {binary-data: movie fragment 61}
  .
  .
  .
  --THIS_STRING_SEPARATES
  Content-type: video/mp4
  X-Timestamp: 119.0
  {binary-data: movie fragment 120}

  --THIS_STRING_SEPARATES--

```

FIG. 29

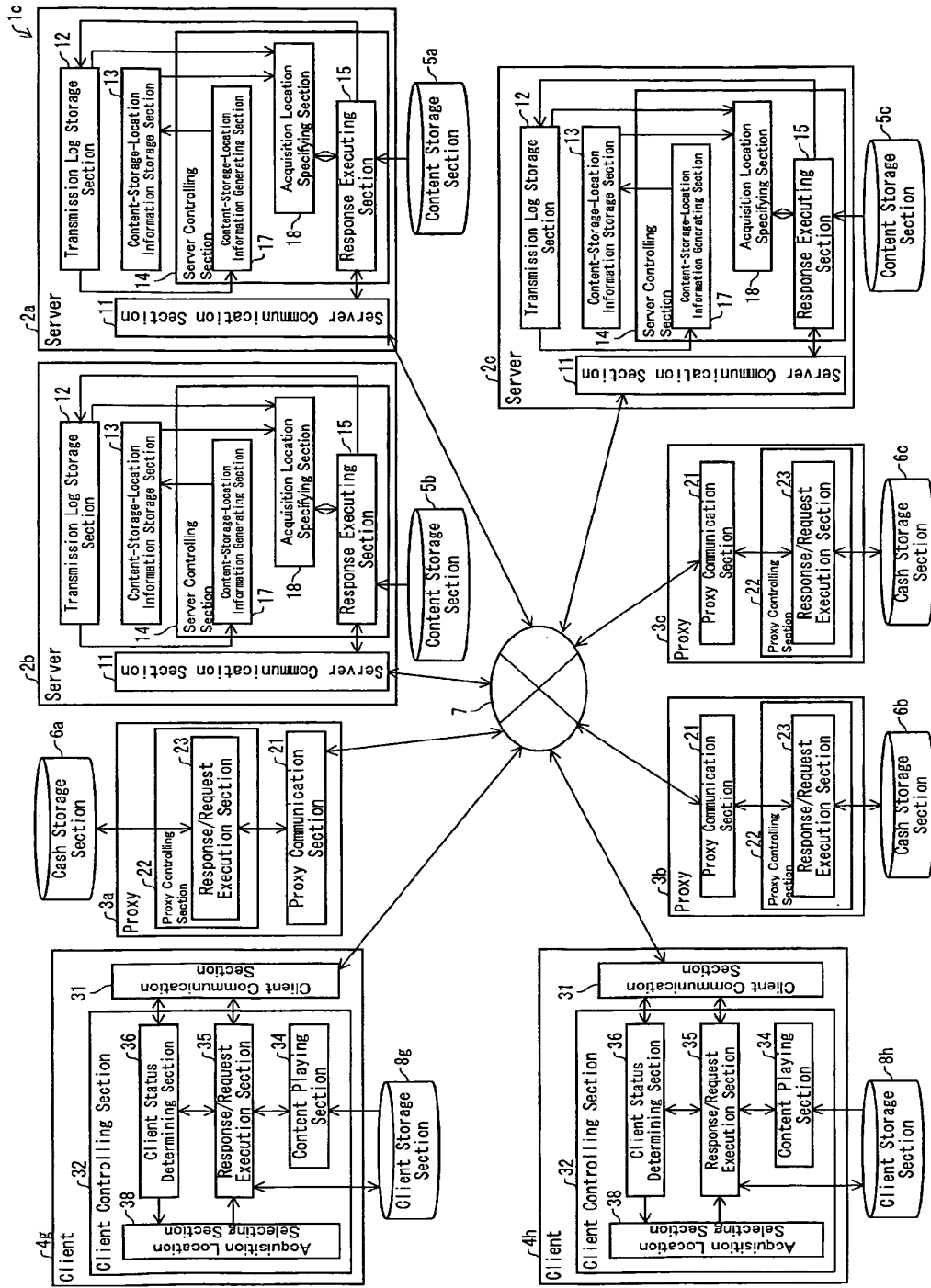


FIG. 30

Content-Storage-Location Information

	Date	Content ID	Address Of Storage Location
71	Sun, 31 May 2013 13:53:38 GMT	content1	http://example-proxy1.com
72	Sun, 31 May 2013 13:53:38 GMT	content1	http://example-client1.com
73	Mon, 01 Jun 2013 08:05:30 GMT	content1	http://example-client1.com
74	Mon, 01 Jun 2013 08:05:30 GMT	content1	http://example-client2.com
	:	:	:
75	Tue, 25 Feb 2014 15:32:10 GMT	content1	http://srv2.example.com

FIG. 31

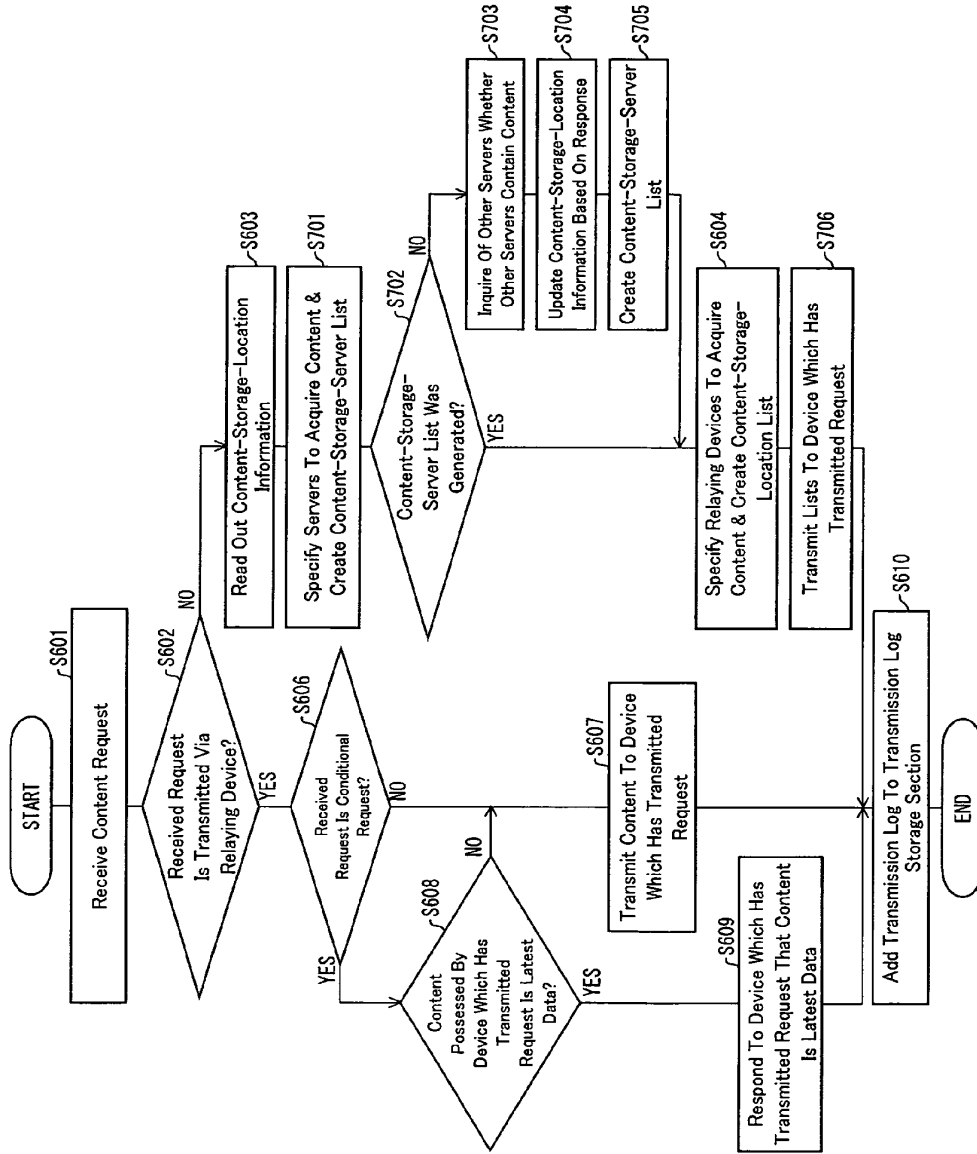


FIG. 32

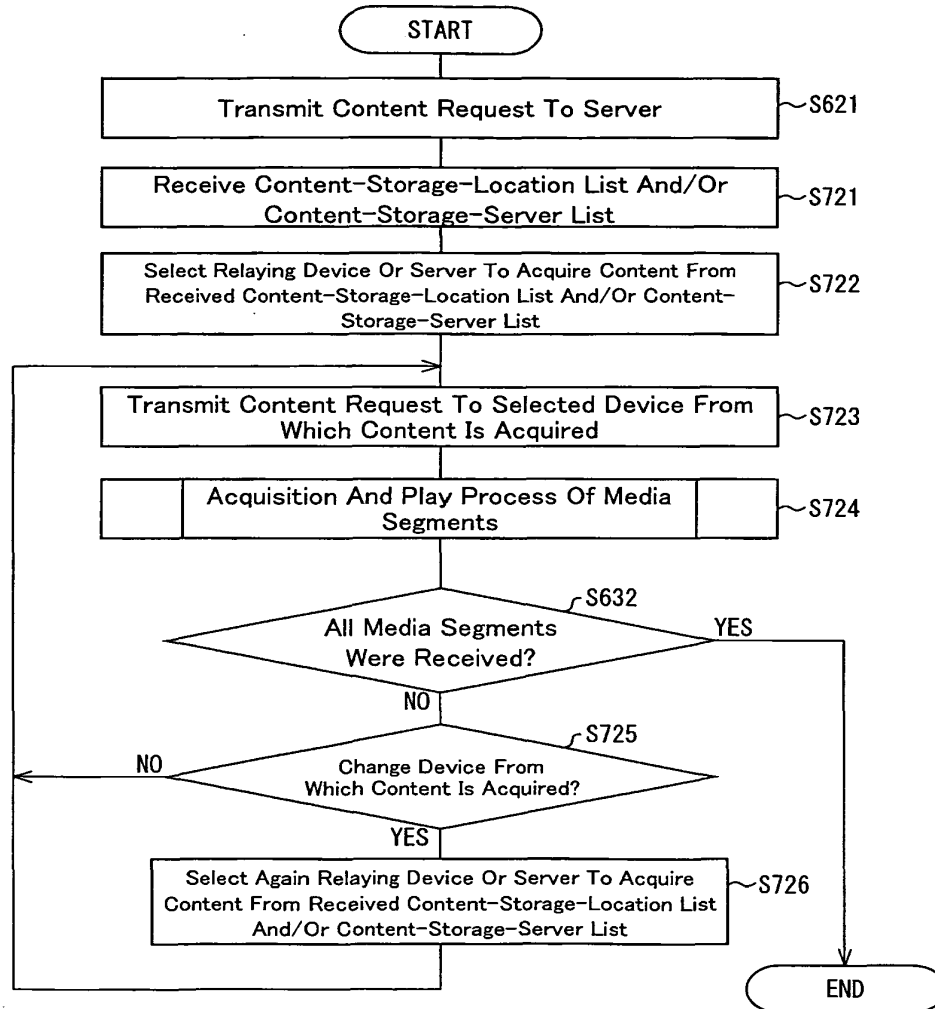


FIG. 33

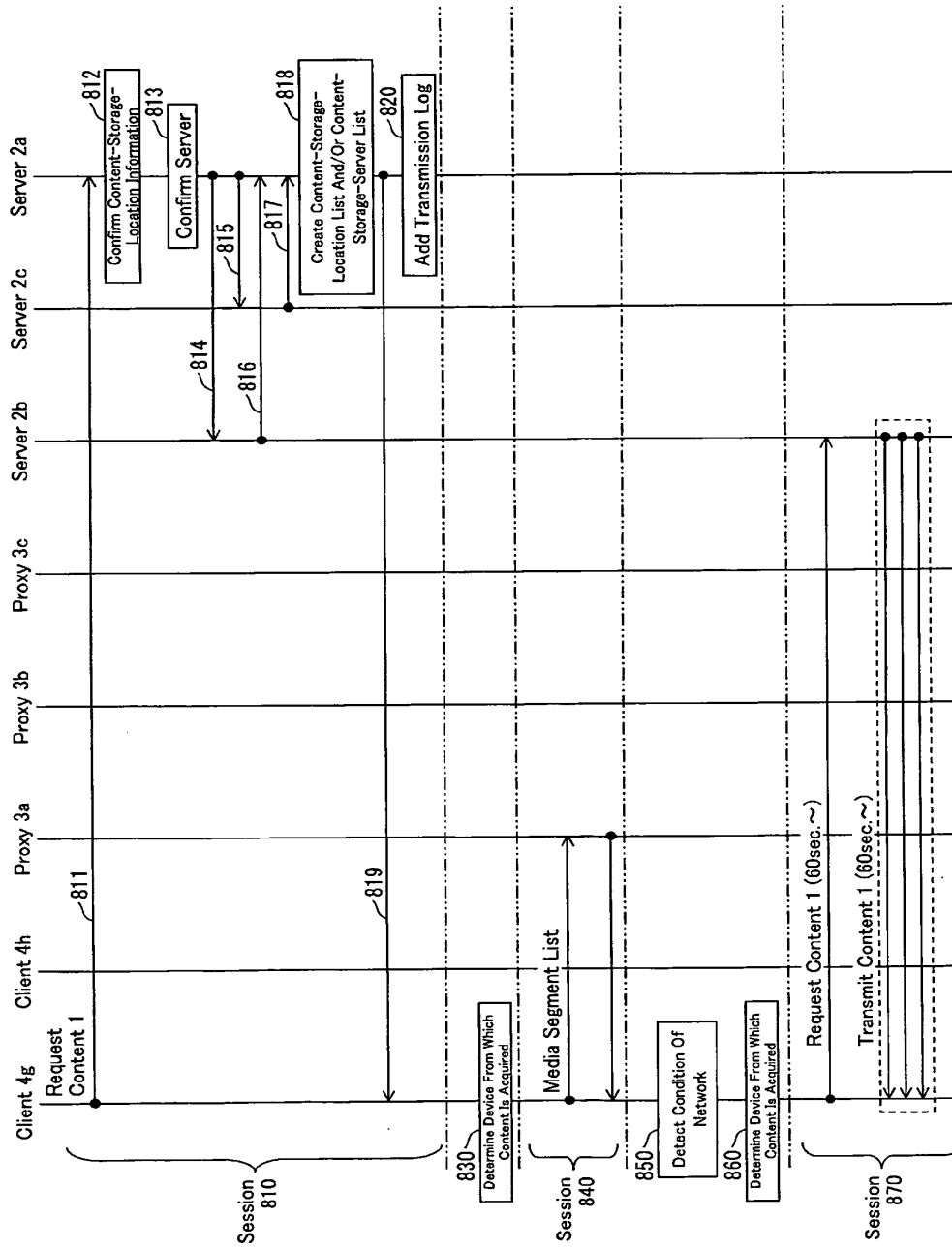




FIG. 34

( a ) Session 810 "Server 2a To Client 4g" Response(819)

HTTP1.1 305 Use Proxy

Location: <http://example-proxy1.com>

X-Alternative-Proxy-List: <http://example-client2.com>, <http://example-proxy3.com>

X-Alternative-Server-List: <http://srv2.example.com>, <http://srv3.example.com>

( b ) Session 810 "Server 2a To Client 4g" Response(819)

HTTP1.1 303 See Other

Location: <http://srv2.example.com/content1/0>

X-Alternative-Server-List: <http://srv3.example.com>

FIG. 35

```

(a) <?xml version="1.0" encoding="UTF-8" standalone="no" ?>
    <MPD
      mediaPresentationDuration="PT120S"minBufferTime="PT1S" type="OnDemand" >

      <Period id="0">
        <Group mimeType="video/3gpp; codecs='avc1,mp4a'" lang="jp">
          <SegmentInfoDefault >
            <BaseURL>http://srv2.example.com/</BaseURL>
            <BaseURL>http://srv3.example.com/</BaseURL>
          </SegmentInfoDefault>
          <Representation bandwidth="19000000" id="1">
            <SegmentInfo>
              <Url sourceURL="content1/0.mp4"/>
              <Url sourceURL="content1/1.mp4"/>
              <Url sourceURL="content1/2.mp4"/>
              <Url sourceURL="content1/3.mp4"/>
              <Url sourceURL="content1/4.mp4"/>
              <Url sourceURL="content1/5.mp4"/>
              <Url sourceURL="content1/6.mp4"/>
              <Url sourceURL="content1/7.mp4"/>
              <Url sourceURL="content1/8.mp4"/>
              <Url sourceURL="content1/9.mp4"/>
              <Url sourceURL="content1/10.mp4"/>
              <Url sourceURL="content1/11.mp4"/>
            </SegmentInfo>
          </Representation>
        </Group>
      </Period>
    </MPD>

(b) <?xml version="1.0" encoding="UTF-8" standalone="no" ?>
    <MPD
      mediaPresentationDuration="PT120S"minBufferTime="PT1S" type="OnDemand" >

      <Period id="0">
        <Group mimeType="video/3gpp; codecs='avc1,mp4a'" lang="jp">
          <SegmentInfoDefault >
            <BaseURL>http://srv2.example.com/</BaseURL>
            <BaseURL>http://srv3.example.com/</BaseURL>
          </SegmentInfoDefault>
          <Representation bandwidth="19000000" id="1">
            <SegmentInfo>
              <Url sourceURL="content1/0.mp4"/>
              <Url sourceURL="content1/1.mp4"/>
              <Url sourceURL="content1/2.mp4"/>
            </SegmentInfo>
          </Representation>
          <Representation bandwidth="4000000" id="2">
            <SegmentInfo>
              <Url sourceURL="content2/0.mp4"/>
              <Url sourceURL="content2/1.mp4"/>
              <Url sourceURL="content2/2.mp4"/>
            </SegmentInfo>
          </Representation>
        </Group>
      </Period>
    </MPD>

```

FIG. 36

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<MPD
  mediaPresentationDuration="PT120S"minBufferTime="PT1S" type="OnDemand" >

  <Period id="0">
    <Group
      xlink:href ="http://example.com/content1/resource1.xml"
      xlink:actuate="onRequest"
      mimeType="video/3gpp; codecs='avc1,mp4a'" lang="jp"/>
    </Period>
  </MPD>
```

FIG. 37

```
<Group mimeType="video/3gpp; codecs='avc1,mp4a'" lang="jp">
  <SegmentInfoDefault >
    <BaseUrl>http://srv2.example.com/</BaseUrl>
    <BaseUrl>http://srv3.example.com/</BaseUrl>
  </SegmentInfoDefault>
  <Representation bandwidth="19000000" id="1">
    <SegmentInfo>
      <Url sourceURL="content1/0.mp4"/>
      <Url sourceURL="content1/1.mp4"/>
      <Url sourceURL="content1/2.mp4"/>
      <Url sourceURL="content1/4.mp4"/>
      <Url sourceURL="content1/5.mp4"/>
      <Url sourceURL="content1/6.mp4"/>
      <Url sourceURL="content1/70.mp4"/>
      <Url sourceURL="content1/8.mp4"/>
      <Url sourceURL="content1/9.mp4"/>
      <Url sourceURL="content1/10.mp4"/>
      <Url sourceURL="content1/11.mp4"/>
    </SegmentInfo>
  </Representation>
</Group>
```

FIG. 38

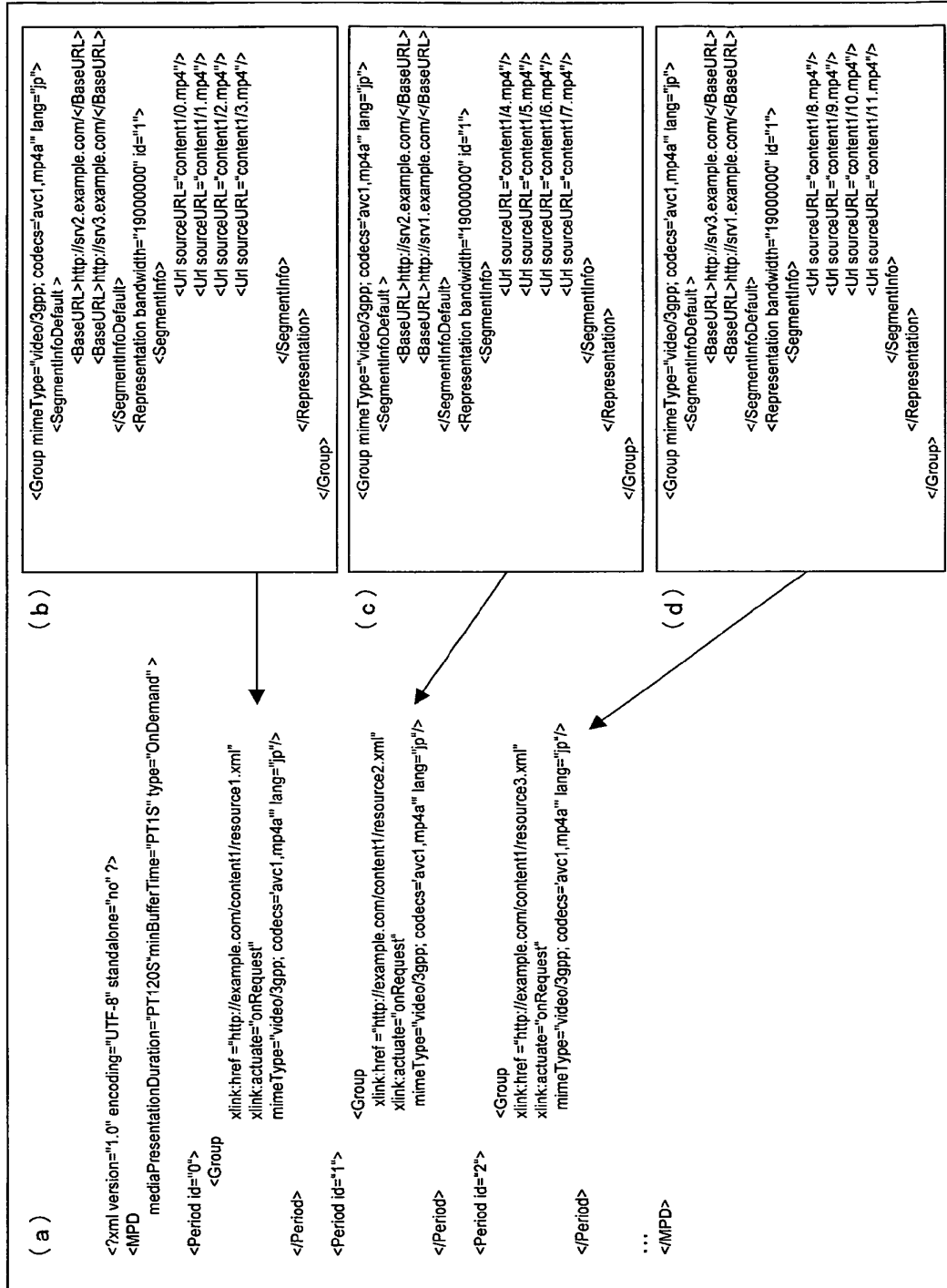


FIG. 39

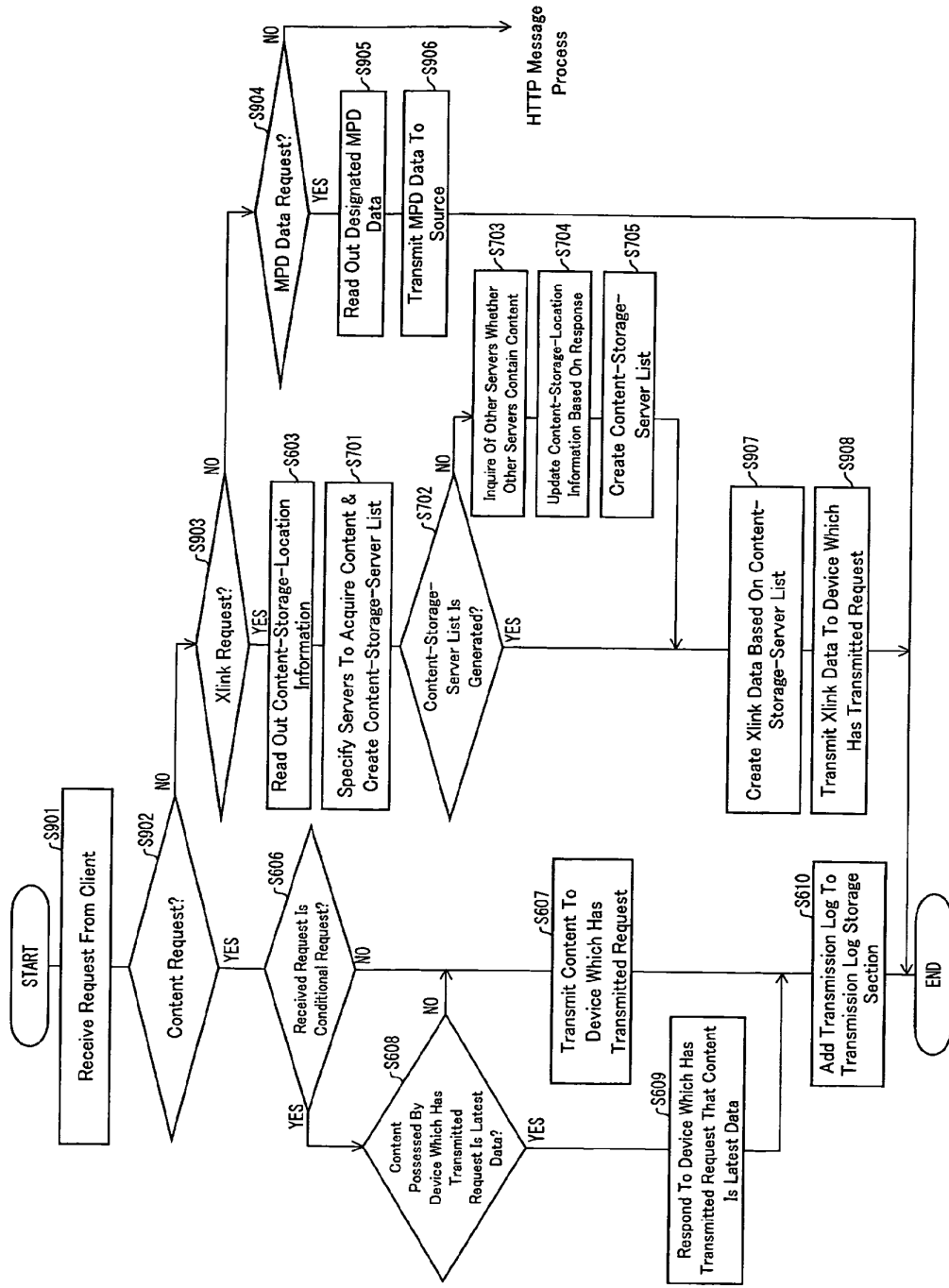


FIG. 40

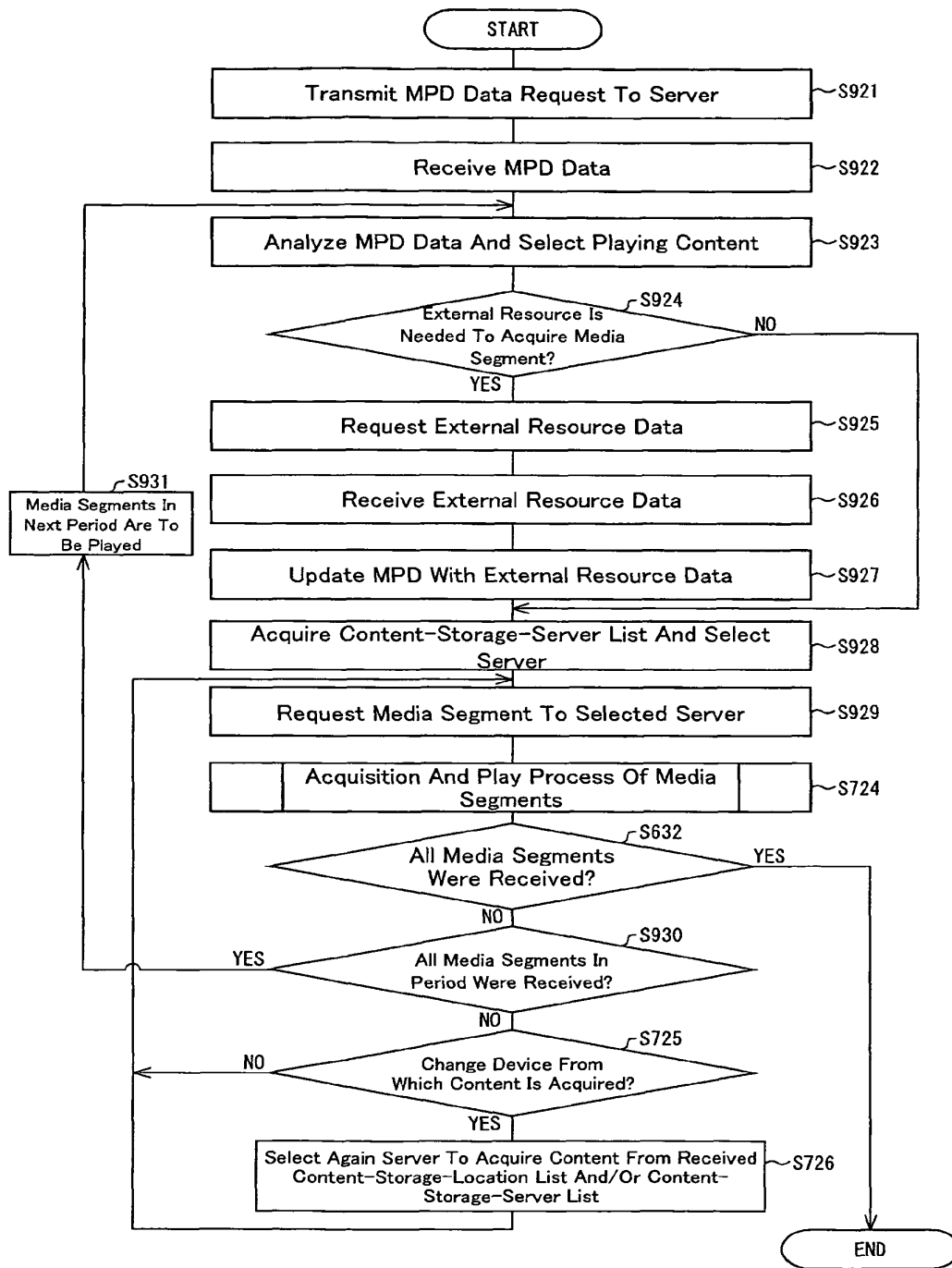
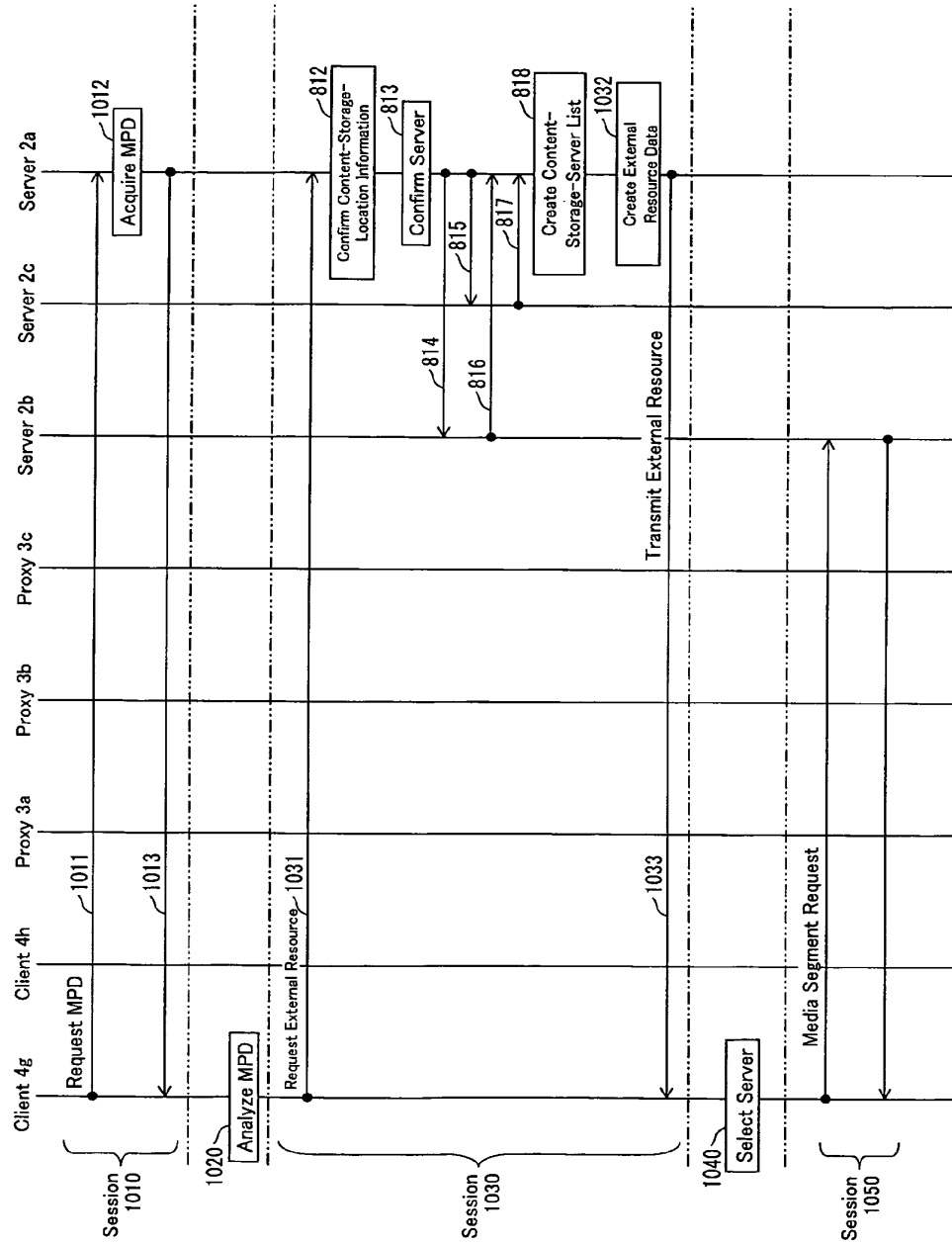


FIG. 41





INTERNATIONAL SEARCH REPORT		International application No. PCT/JP2011/066279
<b>A. CLASSIFICATION OF SUBJECT MATTER</b> H04N7/173(2011.01)i, G06F13/00(2006.01)i, H04L12/56(2006.01)i  According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) H04N7/173, G06F13/00, H04L12/56  Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2011 Kokai Jitsuyo Shinan Koho 1971-2011 Toroku Jitsuyo Shinan Koho 1994-2011  Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	JP 2003-223378 A (Fujitsu Ltd.), 08 August 2003 (08.08.2003), paragraphs [0029] to [0031]; fig. 3 & US 2003/0145066 A1 & EP 1331788 A2	1-28
A	R. Fielding et al., RFC 2616 Hypertext Transfer Protocol HTTP/1.1, 1999.06, "14.45 Via"	2
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 19 August, 2011 (19.08.11)		Date of mailing of the international search report 30 August, 2011 (30.08.11)
Name and mailing address of the ISA/ Japanese Patent Office		Authorized officer
Facsimile No.		Telephone No.

Form PCT/ISA/210 (second sheet) (July 2009)

**REFERENCES CITED IN THE DESCRIPTION**

*This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.*

**Patent documents cited in the description**

- JP 2005110244 A [0004]



Europäisches Patentamt  
European Patent Office  
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number  
EP 14 18 2547

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	EP 2 597 869 A1 (SHARP KK [JP]) 29 May 2013 (2013-05-29) * abstract * * paragraph [0038] * * paragraph [0044] - paragraph [0045] * * paragraph [0048] * * paragraph [0064] - paragraph [0068] * * paragraph [0081] * * paragraph [0118] * * paragraph [0400] - paragraph [0401] * -----	1-11	INV. H04L29/06 H04N21/462
A	US 2003/204602 A1 (HUDSON MICHAEL D [US] ET AL) 30 October 2003 (2003-10-30) * abstract * * paragraph [0013] * * paragraph [0017] - paragraph [0018] * * paragraph [0047] * * paragraph [0064] * * claims 6,7 * -----	1-11	
A	FIELDING R ET AL: "RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1", INTERNET CITATION, June 1999 (1999-06), XP002196143, Retrieved from the Internet: URL:http://www.rfc-editor-org/ [retrieved on 2002-04-15] * the whole document * -----	1-11	TECHNICAL FIELDS SEARCHED (IPC)  H04L H04N
A	WO 2010/090562 A1 (ERICSSON TELEFON AB L M [SE]; THYNI TOMAS [SE]; WELIN ANNIKKI [SE]; GO) 12 August 2010 (2010-08-12) * abstract * * page 4, paragraph 2 * * page 7, paragraph 1 * * page 13, last paragraph - page 14, paragraph 1 * * page 15, line 3 - line 19 * ----- -/--	1-11	
3 The present search report has been drawn up for all claims			
Place of search Munich		Date of completion of the search 30 July 2015	Examiner Cichra, Michael
CATEGORY OF CITED DOCUMENTS X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons ..... & : member of the same patent family, corresponding document			

EPO FORM 1503 03/02 (P04C01)



Europäisches Patentamt  
European Patent Office  
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number  
EP 14 18 2547

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	US 2012/124173 A1 (DE PRADIPTA [IN] ET AL) 17 May 2012 (2012-05-17) * abstract * * paragraph [0003] * * paragraph [0025] * * paragraph [0039] * * paragraph [0043] * * claim 1 *	12-16	TECHNICAL FIELDS SEARCHED (IPC)
A	----- WO 2011/068784 A1 (AZUKI SYSTEMS INC [US]; MA KEVIN J [US]; BARTOS RADIM [US]; XU JIANGUO) 9 June 2011 (2011-06-09) * abstract * * page 5, line 7 - page 5, line 22 * * page 14, line 9 - page 14, line 11 * * page 25, line 12 - page 25, line 28 * * page 27, line 20 - page 28, line 16 *	12-16	
A	----- US 2002/069241 A1 (NARLIKAR GIRIJA [US] ET AL) 6 June 2002 (2002-06-06) * abstract * * paragraph [0009] * * paragraph [0018] * * paragraph [0020] * * paragraph [0027] * * claim 12 * -----	12-16	
The present search report has been drawn up for all claims			
Place of search Munich		Date of completion of the search 30 July 2015	Examiner Cichra, Michael
CATEGORY OF CITED DOCUMENTS X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons ..... & : member of the same patent family, corresponding document	

3  
EPO FORM 1503 03/02 (P04C01)



Application Number  
EP 14 18 2547

### CLAIMS INCURRING FEES

The present European patent application comprised at the time of filing claims for which payment was due.

- Only part of the claims have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due and for those claims for which claims fees have been paid, namely claim(s):
- No claims fees have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due.

### LACK OF UNITY OF INVENTION

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

see sheet B

- All further search fees have been paid within the fixed time limit. The present European search report has been drawn up for all claims.
- As all searchable claims could be searched without effort justifying an additional fee, the Search Division did not invite payment of any additional fee.
- Only part of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the inventions in respect of which search fees have been paid, namely claims:
- None of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims, namely claims:
- The present supplementary European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims (Rule 164 (1) EPC).

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

1. claims: 1-11

Retrieving content by a client (1st device) from a content server (2nd server) via a proxy (2nd device), which was allocated by a broker (1st server) on request of the client (1st device).

---

2. claims: 12-16

Retrieving content from a server, the content being partitioned into content slices, where each content slice is retrieved via one of a group of intermediary devices.

---

**ANNEX TO THE EUROPEAN SEARCH REPORT  
ON EUROPEAN PATENT APPLICATION NO.**

EP 14 18 2547

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

30-07-2015

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 2597869 A1	29-05-2013	CN 103119958 A	22-05-2013
		EP 2597869 A1	29-05-2013
		US 2013117413 A1	09-05-2013
		WO 2012011450 A1	26-01-2012
-----			
US 2003204602 A1	30-10-2003	US 2003204602 A1	30-10-2003
		US 2003204605 A1	30-10-2003
		US 2003204613 A1	30-10-2003
		US 2009049185 A1	19-02-2009
		US 2009055506 A1	26-02-2009
		US 2009055547 A1	26-02-2009
		US 2009210549 A1	20-08-2009
		US 2010011061 A1	14-01-2010
		US 2015106437 A1	16-04-2015
-----			
WO 2010090562 A1	12-08-2010	CN 102308549 A	04-01-2012
		EP 2394409 A1	14-12-2011
		US 2011282945 A1	17-11-2011
		WO 2010090562 A1	12-08-2010
-----			
US 2012124173 A1	17-05-2012	US 2012124173 A1	17-05-2012
		US 2013124689 A1	16-05-2013
-----			
WO 2011068784 A1	09-06-2011	US 2012265892 A1	18-10-2012
		WO 2011068784 A1	09-06-2011
-----			
US 2002069241 A1	06-06-2002	NONE	
-----			

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

# On the Leakage of Personally Identifiable Information Via Online Social Networks

Balachander Krishnamurthy  
AT&T Labs – Research  
Florham Park, NJ USA  
bala@research.att.com

Craig E. Wills  
Worcester Polytechnic Institute  
Worcester, MA USA  
cew@cs.wpi.edu

## Abstract

For purposes of this paper, we define “Personally identifiable information” (PII) as information which can be used to distinguish or trace an individual’s identity either alone or when combined with other information that is linkable to a specific individual. The popularity of Online Social Networks (OSN) has accelerated the appearance of vast amounts of personal information on the Internet. Our research shows that it is possible for third-parties to link PII, which is leaked via OSNs, with user actions both within OSN sites and elsewhere on non-OSN sites. We refer to this ability to link PII and combine it with other information as “leakage”. We have identified multiple ways by which such leakage occurs and discuss measures to prevent it.

## Categories and Subject Descriptors

C.2 [Computer-Communication Networks]: Network Protocols—*applications*

## General Terms

Measurement

## Keywords

Online Social Networks, Privacy, Personally Identifiable Information

## 1. INTRODUCTION

For purposes of this paper, “Personally identifiable information” (PII) is defined as information which can be used to distinguish or trace an individual’s identity either alone or when combined with other public information that is linkable to a specific individual. The growth in identity theft has increased concerns regarding unauthorized disclosure of PII. Over half a billion people are on various Online Social Networks (OSNs) and have made available a vast amount of personal information on these OSNs. OSN users make

their information available (subject to the privacy policy of the OSN) to the authorized list of other OSN users, such as their ‘friends’. Their profiles form a part of their online identity.

There has been a steady increase in the use of third-party servers, which provide content and advertisements for Web pages belonging to first-party servers. Some third-party servers are aggregators, which track and aggregate user viewing habits across different first-party servers, often via tracking cookies. Earlier, in [6] we showed that a few third-party tracking servers dominate across a number of popular Online Social Networks. Subsequently, in [7] we found that the penetration of the top-10 third-party servers across a large set of popular Web sites had grown from 40% in October 2005 to 70% in September 2008. A key question that has not been examined to our knowledge is whether PII belonging to any user is being leaked to these third-party servers via OSNs. Such leakage would imply that third-parties would not just know the viewing habits of *some* user, but would be able to associate these viewing habits with a specific person.

In this work we have found such leakage to occur and show how it happens via a combination of HTTP header information and cookies being sent to third-party aggregators. We show that *most users on OSNs are vulnerable to having their OSN identity information linked with tracking cookies*.<sup>1</sup> Unless an OSN user is aware of this leakage and has taken preventive measures, it is currently trivial to access the user’s OSN page using the ID information. The two immediate consequences of such leakage: First, since tracking cookies have been gathered for several years from *non-OSN* sites as well, it is now possible for third-party aggregators to associate identity with those *past* accesses. Second, since users on OSNs will continue to visit OSN *and* non-OSN sites, such actions in the *future* are also liable to be linked with their OSN identity.

Tracking cookies are often opaque strings with hidden semantics known only to the party setting the cookie. As we also discovered, they may include visible identity information and if the same cookie is sent to an aggregator, it would constitute another vector of leakage. Due to the longer lifetime of tracking cookies, if the identity of the person is established even once, then aggregators could internally associate the cookie with the identity. As the same tracking cookie is sent from different Websites to the aggregator, the user’s

<sup>1</sup>We have shared this information to all the OSNs we studied so that they may make informed decisions regarding preventative measures and subscriber notification.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2001 ACM 0-89791-88-6/97/05 ...\$5.00.



movements around the Internet can now be tracked *not* just as an IP address, but be associated with the unique identifier used to store information about users on an OSN. This OSN identifier is a pointer to PII about the user.

Cookies and other tracking mechanisms on the Internet have been prevalent for a long time. The general claim of aggregators is that they create profiles of users based on their Internet behavior, but do not gather or record PII. Although we do not know that aggregators are recording PII, we demonstrate with this work that it is undeniable that information *is* available to them. Aggregators do not have to take any action to receive this information. As part of requests, they receive OSN identifiers with pointers to the PII or in some cases, directly receive pieces of PII. This PII information can be joined with information from tracking cookies obtained from the user's traversal to any site that triggers a visit to the same aggregator. The ability to link information across traversals on the Internet coupled with the wide range of daily actions performed by hundreds of millions of users on the Internet raises privacy issues, particularly to the extent users may not understand the consequences of having their PII information available to aggregators.

OSNs do have privacy policies on which OSN users rely when setting up and maintaining their account. These policies typically state that OSNs provide *non*-identifying information to third-parties as an aid in serving advertisements and other services. Many users, however may not understand the implications. The availability of a user's OSN identifier allows a third-party access to a user's name and other linkable PII that can identify a user. The goal of this work is not a legal examination of privacy policies, but to bring a technical examination of the observed leakage to the community's attention.

Section 2 enumerates pieces of personally identifiable information and examines the level of availability for these pieces across a number of OSNs. Section 3 describes our study of PII-related leakage in popular OSNs. Section 4 presents ways in which such leakage occurs across OSNs. Section 5 discusses techniques for possible protection against such leakage by the various parties involved in the transactions. We then look at preliminary work on the problem of PII leakage in non-OSN sites in Section 6. Section 7 concludes with a summary and description of future work.

## 2. AVAILABILITY OF PII IN OSNS

It is important to understand how the information provided to OSNs corresponds with PII and the nature of availability of such information to other users. PII is defined in [5] as referring to "information which can be used to distinguish or trace an individual's identity, such as their name, social security number, biometric records, etc. alone, or when combined with other personal or identifying information which is linked or linkable to a specific individual, such as date and place of birth, mother's maiden name, etc."

A recent report identifies a number of examples that may be considered PII [9] including: Name (full name, maiden name, mother's maiden name), personal identification number (e.g., Social Security Number), address (street or email address), telephone numbers, or personal characteristics (such as photographic images especially of face or other distinguishing characteristic, X-rays, fingerprints, or other biometric image). They can also include: asset information (IP or MAC address or a persistent static identifier that

consistently links to a particular person or a small, well-defined group of people), or information identifying personally owned property (vehicle registration or identification number).

The report also includes examples of information about an individual that is linked or linkable to one of the above (e.g., date of birth, place of birth, race, religion, weight, activities, or employment, medical, education, or financial information). A well-known result in linking pieces of PII is that most Americans (87%) can be uniquely identified from a birth date, five-digit zip code, and gender [8]. A decade-old report [4] by the Federal Trade Commission in the U.S. specifically warned about the potential of linking profiles derived via tracking cookies and information about consumers obtained offline. It should be stressed that our work focuses on additional information obtained *online*.

With this understanding of PII, we analyze its availability and accessibility in the profile information for OSN users on popular OSNs. We used the 11 OSNs in our previous work [6]: Bebo, Digg, Facebook, Friendster, Hi5, Imeem, LiveJournal, MySpace, Orkut, Twitter and Xanga. We also included a 12th OSN for this study, LinkedIn, which is a popular professionals-oriented OSN.

The pieces of PII for an OSN user include: name (first and last), location (city), zip code, street address, email address, telephone numbers, and photos (both personal and as a set). We also include pieces of information about an individual that are linkable to one of the above: gender, birthday, age or birth year, schools, employer, friends and activities/interests. We only note availability if users are specifically asked for it as part of their OSN profile; otherwise we would not expect users to provide it. We do *not* process contents of OSN users' pages to see if they have additional personal information. Not all profile elements are filled in by users and entries may of course be false.

Table 1 shows the results of our analysis with the count of OSNs (out of 12) exhibiting the given degree of availability for each piece of PII (row). The first column indicates the number of OSNs where the piece of PII is available to all users of the OSN and the user *cannot* restrict access to it. This piece may also be available to non-users of the OSN—thus a primary source of concern. The second column shows the number of OSNs where the piece of PII is available to all users in the OSN via the default privacy settings, but the user can restrict access via these settings. The third column shows a count of OSNs where there is a piece of PII that users can fill out in their profile, but by default the value is not shown to everyone. The fourth column shows the count of OSNs where a piece of PII is not part of a user's profile and thus the information is not available unless the user goes out of their way to add it on their page.

The rows are sorted in decreasing order of availability and thus leakage (personal photos are available widely while street address are rarely present). The values in the first two columns raise more privacy concerns (hence the double vertical line). Prominence is given to them as we found in [6] that default privacy settings are generally permissive allowing access to strangers in all OSNs. We also found that despite privacy controls to limit access, between 55 and 90% of users in OSNs retain default settings for viewing of profile information and 80–97% for viewing of friends. The latter two columns suggests that some OSNs are concerned about the extent of private information that may be visible on

**Table 1: PII Availability Counts in 12 OSNs**

Piece of PII	Level of Availability			
	Always Available	Available by default	Unavailable by default	Always Unavailable
Personal Photo	9	2	1	0
Location	5	7	0	0
Gender	4	6	0	2
Name	5	6	1	0
Friends	1	10	1	0
Activities	2	8	0	2
Photo Set	0	9	0	3
Age/Birth Year	2	5	4	1
Schools	0	8	1	3
Employer	0	6	1	5
Birthday	0	4	7	1
Zip Code	0	0	10	2
Email Address	0	0	12	0
Phone Number	0	0	6	6
Street Address	0	0	4	8

OSNs. As we will see later, although some pieces of PII are unavailable to others in the OSN (the later rows) they may still leak via other means.

### 3. LEAKAGE STUDY METHODOLOGY

The concentration and default availability of pieces of PII for OSNs shown in Table 1 motivates our study to examine if and how PII is leaked via OSNs. We know that OSNs use a unique identifier for each of their users as a key for storing information about them. Such an identifier can also appear as part of a URI when user performs various actions on an OSN. For example, the identifier is often shown in the Request-URI when a user views or edits their OSN profile or clicks on a friend’s picture. The use of this identifier is not a privacy concern if all interactions stay *within* the OSN, but as shown in [6] there is also interaction with third-party servers. If this interaction involves leakage of the unique identifier for a user then the third-party has a pointer to access PII of the user. The third-party may also have other information: tracking cookies with a long expiry period or source IP addresses, to join with the PII.

For the study, we log into each OSN and perform actions, such as accessing the user profile, that cause the OSN identifier to be displayed as part of the URI. We also click on displayed ads. While performing these actions we turn on the “Live HTTP Headers” [14] browser extension in Firefox, which displays HTTP request/response headers for all object retrievals. We analyze these headers to determine if any third-party servers are contacted, and if the user’s OSN identifier or specific pieces of PII are visibly sent to the third-party servers via any HTTP header. Note that we will not detect if this information is sent via opaque strings.

A set of relevant request headers are shown in Figure 1 to illustrate an actual example of such a retrieval. Here `/pagead/test_domain.js` is retrieved from the server `googleads.g.doubleclick.net` as part of retrieving the set of objects needed to display content for a page on `myspace.com`.<sup>2</sup> As shown, the browser also includes the `Referer` (sic) header and a stored cookie belonging to `doubleclick.net`.

<sup>2</sup>In all examples, an OSN identifier of “123456789” or “jdoe” is substituted for the actual identifier in our study. Cookies and other strings are also anonymized.

```
GET /pagead/test_domain.js HTTP/1.1
Host: googleads.g.doubleclick.net
Referer: http://profile.myspace.com/index.cfm?
fuseaction=user.viewprofile&friendid=123456789
Cookie: id=2015bdfb9ec|t=1234359834|et=730|cs=7aepmsks
```

**Figure 1: Sample Leakage of OSN Identifier to a Third-Party**

The `doubleclick.net` server is able to associate the user’s identifier MySpace (“friendid” is the label used in URIs by MySpace to identify users, similar to ‘id’ or ‘userid’ used in other OSNs) with the DoubleClick cookie. Armed with this information the aggregator can join its “profile” of user accesses employing this cookie with any information available via the MySpace identifier.

## 4. LEAKAGE OF PII

Using the methodology described in Section 3 we examined the results of actions performed while logged onto each of the 12 OSNs in our study. We found four types of PII leakage involving the: 1) transmission of the OSN identifier to third-party servers from the OSN; 2) transmission of the OSN identifier to third-party servers via popular external applications 3) transmission of specific pieces of PII to third-party servers; and 4) linking of PII leakage within, across, and beyond OSNs. We now describe and show specific examples of how PII is transmitted to third-party aggregators.

### 4.1 Leakage of OSN Identifier

Our initial focus in the study is on the transmission of a user’s OSN unique identifier to a third-party. Based on results in Table 1 the possession of this identifier allows a third-party to gain much PII information about a OSN user to join with the third-party profile information about a user’s activity on non-OSN sites. Analyzing the request headers we obtain via the Live HTTP Headers extension, we find that the OSN identifier is transmitted to a third-party in at least three ways: the `Referer` header, the Request-URI, or a cookie. Examples for these three types of leakage are shown in Figure 2. Note that accesses to third-party servers are often triggered *without* explicit action (e.g., clicking on an advertisement) on the user’s part.

```
GET /clk;203330889;26770264;z;u=ds&sv1=170988623...
Host: ad.doubleclick.net
Referer: http://www.facebook.com/profile.php?
id=123456789&ref=name
Cookie: id=2015bdfb9ec|t=1234359834|et=730|cs=7aepmsks
```

(a) Via Referer Header

```
GET /__utm.gif?..utmhn=twitter.com&utmip=/profile/jdoe
Host: www.google-analytics.com
Referer: http://twitter.com/jdoe
```

(b) Via Request-URI

```
GET ...&g=http%3A//digg.com/users/jdoe&...
Host: z.digg.com
Referer: http://digg.com/users/jdoe
Cookie: s_sq=...http%25253A//digg.com/users/jdoe...
```

(c) Via Cookie

**Figure 2: Leakage of OSN ID to a Third-Party**

First, OSN identifiers can leak via the `Referer` header of a request when an identifier is part of the URI for a page.

OSNs typically include the identifier as part of a URI when showing the contents of any user’s profile. As part of loading the contents for this page, the browser retrieves one or more objects from a third-party server. Each request contains the **Referer** header in the HTTP request, which passes along the OSN id. Figure 2a shows an example of this leakage where an object from the third-party **ad.doubleclick.net** is retrieved as part of a **www.facebook.com** page where the URI contains the OSN id and is thus included in the **Referer** header. In addition, the cookie for **doubleclick.net** is sent to the third-party server, which can now link this cookie with the OSN id. In testing, we observed similar examples of OSN id leakage to a third-party server via the **Referer** header in the presence of a third-party cookie for 9 of the 12 OSNs that we studied.

Second, OSN identifiers can leak to a third-party server via the request Request-URI. A typical example is shown in Figure 2b where a request to the analytics server **www.google-analytics.com** is made from a **twitter.com** page. This transmission not only allows the third-party to gather analytic information, but also to know the specific identifier of the user on the OSN. We observed such leakage for 5 of our 12 OSNs. The third-party domain **google-analytics.com** occurred in all five cases.

Third, OSN identifiers can leak to a third-party server via a first-party cookie when an OSN page contains objects from a server that appears to be part of the first-party domain, but actually belongs to a third-party aggregation server. We observed the increased use of such “hidden” third-party servers in [7] and observe similar use for OSNs in this work. In the example of Figure 2c, when we determine the authoritative DNS server for server **z.digg.com** we find that it is actually a server that is part of **omniture.com**, a large third-party tracking company [7]. Thus the browser includes the first-party cookie for **digg.com** in the request, which includes the OSN id, but the request is actually sent, because of the DNS mapping, to an **omniture.com** server. As the example shows, the OSN id is also sent via the Request-URI and **Referer** header, but this example is notable because it demonstrates another avenue of id leakage. We observed leakage of the OSN to such a “hidden” third-party server via a first-party cookie for 2 of the 12 OSNs.

In all, we observed the OSN id being leaked to a third-party server via one of these ways for 11 of the 12 OSNs. Such leakage allows the third-party to merge the OSN id with the profile of tracking information maintained by them.

The only OSN for which we did not observe such behavior is Orkut—part of the Google family of domains. Orkut requires a login via a Google account that is tracked via a Google cookie thus allowing the Orkut identifier to be directly associated with other **google.com** activity (e.g., search).

## 4.2 Leakage Via External Applications

External applications have become increasingly popular; Facebook alone has over 55,000 external applications. The applications are installed via the OSN but run on external servers not owned or operated by the OSNs themselves. The user is warned that downloading applications will result in the OSN sharing user-related information (including the identifier) with the external applications. Such sharing is required so that application providers can use them in API calls while interacting with the OSN. The user’s social graph is accessible to the application only via the OSN and

users interact with other OSN users (often their friends) via the application. Popular gaming applications and social interaction applications take advantage of the social graph to expand their reach quickly.

We observe that external applications of OSNs may themselves leak the OSN identifier to third-party aggregators. Once again, it is unclear if the OSN identifier needs to be made available to the external aggregator. While the leakage of the identifier in such cases is technically not the fault of the OSN, the user may not be aware of the secondary leakage occurring through external applications. Examples of leakage via requests involving external applications of MySpace and Facebook are shown in Figure 3.



**Figure 3: Leakage of OSN ID to a Third-Party From an External Application**

Figure 3a shows an example of a retrieved object from the third-party server **view.atdmt.com** with the MySpace identifier included in the **Referer** header. This retrieval follows a previous retrieval (not shown) where the use of a MySpace application causes the OSN identifier to be sent to the third-party server **delb.opt.fimserve.com** as part of the Request-URI. The example in Figure 3b shows a Facebook user’s identifier being passed on by the popular external application “iLike” to a third-party aggregator **google-analytics.com** via the Request-URI. Figure 3c shows leakage via the Request-URI and **Cookie** header via a different application “Kickmania!” to an advertisement tracker **adtracker.socialmedia.com**.

## 4.3 Leakage of Pieces of PII

Beyond our initial focus on leakage of the OSN identifier, we also observe cases where pieces of PII are *directly* leaked to third-party servers via the Request-URI, **Referer** header and cookies. Figure 4 shows two such examples.

In Figure 4a, the third-party server **ads.sixapart.com** is directly given a user’s age and gender via the Request-URI. This request is generated for an object on the user’s profile page. This information is obtained from profile information stored for the user on **livejournal.com**. The third-party server also receives the OSN identifier via the

```
GET /show?gender=M&age=29&country=US&language=en...
Host: ads.sixapart.com
Referer: http://jdoe.livejournal.com/profile
```

(a) Age and Gender Via Request-URI

```
GET /st?ad_type=iframe&age=29&gender=M&e=&zip=11301&...
Host: ad.hi5.com
Referer: http://www.hi5.com/friend/profile/
displaySameProfile.do?userid=123456789
Cookie: LoginInfo=M_AD_MI_MS|US_0_11301;
        Userid=123456789;Email=jdoe@email.com;
```

(b) Age, Gender, Zip and Email Via Request-URI and Cookie

Figure 4: Leakage of Pieces of PII to a Third-Party

**Referer** header, but obtains these two pieces of PII without even the need for a lookup.

In Figure 4b the server `ad.hi5.com` appears to be part of the `hi5.com` domain, but based on its authoritative DNS, it is actually served by the third-party domain `yieldmanager.com`. This third-party domain not only receives a user's age and gender, but also the user's zip code. These pieces of PII are supplied as part of the Request-URI. In addition, the first-party cookie for `hi5.com` contains the user's zip code and email address. Thus the third-party domain not only receives four pieces of PII, but the OSN is disclosing PII about the user that may not even be available to other users within the OSN. In our study, 2 of the 12 OSNs directly leak pieces of PII to third-parties via the Request-URI, **Referer** header and cookies.

#### 4.4 Linking PII Leakage

Lastly, we examine possible linkages of PII leakage across, within, and beyond OSNs.

Across OSNs, once a third-party server is leaked PII information via one OSN, it may then also be leaked information via another OSN to which the same user belongs. For example, the cookie for `doubleclick.net` shown in the examples of Figure 1 and 2a means that DoubleClick can link the PII from across both MySpace and Facebook. This linkage is important because it not only allows the aggregator to mine PII from more than one OSN, but join this PII with the viewing behavior of this user.

Within an OSN, it is possible for a third-party server to not only obtain the OSN identifier for a user, but also the identifiers for the user's friends and other users of interest within the OSN. For example, a user viewing a friend's profile will leak that friend's OSN identifier.

Finally looking beyond the OSN, the use of a third-party tracking cookie allows the PII available from the OSN to be linked with other online user activity. For example, Figure 5 shows a retrieval from a site that a user may not want to be known to others, yet is linked to the same cookie as used to access MySpace and Facebook.

```
GET /pagead/ads?client=ca-primedia-premium_js&...
Host: googleads.g.doubleclick.net
Referer: http://pregnancy.about.com/
Cookie: id=2015bdfb9ec|t=1234359834|et=730|cs=7aepmsks
```

Figure 5: Example of Third-Party Cookie for Non-OSN Server

## 5. PROTECTION AGAINST PII LEAKAGE

We have demonstrated a variety of scenarios whereby OSN identifiers and PII present on the corresponding user profiles leak via different OSNs. We now examine the parties involved in the leakage and the ways by which they can help prevent it. There are primarily four parties involved in the series of transactions: the user, third-party aggregators, the OSN, and any external applications accessed via the OSN.

Users ability to block leakage of PII range from the draconian, albeit effective, one of not disclosing any in the first place to being highly selective about the type and nature of personal information shared. Facebook applications have been created to increase awareness of information that could be used in security questions [10] and provide mechanisms for additional privacy protection [11]. Known privacy protection techniques at the browser include filtering out HTTP headers (e.g., **Referer**, **Cookie**), and refusing third-party cookies. The potential problem with the **Referer** header to leak private information was identified in 1996 (!) in the HTTP/1.0 specification [2]:

Because the source of a link may be private information or may reveal an otherwise private information source, it is strongly recommended that the user be able to select whether or not the Referer field is sent.

Firefox allows direct blocking of **Referer** header [3] or as add-on with more per-site control [1]. With user customization, some actions may cause further accesses to be affected. For example, some servers check the **Referer** header before they answer any requests, in an attempt to prevent their content from being linked to or embedded elsewhere. Protection techniques could be deployed at a proxy [12, 13] to benefit all users behind it. Recently, the HTTP Working Group has had discussions on new headers (such as the **Origin** header) to replace the **Referer** header.<sup>3</sup> Only the information needed for identifying the principal that initiated the request would be included and path or query portions of the Request-URI are excluded. The proposal has not advanced significantly. Additionally, as we have demonstrated, even if the user filtered the **Referer** header and blocked cookies, the OSN identifier is also leaked in the GET or POST request via the Request-URI.

Second, aggregators could filter out any PII-related headers that arrive at their servers and ensure that tracking mechanisms are clean of PII at all times. Publishing the hidden semantics of cookies could work as a confidence building measure; the current opaque string model implies that users will not know if different cookies received (e.g., after deleting older cookies) are being correlated.

Third, OSNs could ensure that a wide range of privacy measures are available to members. Providing strong privacy protection by default allows an OSN to distinguish itself from other competing OSNs. Techniques at OSNs are in reality much easier. Most leakage identified in this study originated from the OSN allowing the internal user identifier to be visible to the browser unnecessarily leading to the population of the **Referer** header. A straightforward solution is to strip any visible URI of userid information. Alternately the OSN could keep a session-specific value for the user's identifier or maintain an internal hash table of the ID and present a dynamically generated opaque string to the browser. If the opaque string is included in the **Referer**

<sup>3</sup>Currently available at <http://tools.ietf.org/html/draft-abarth-origin-00>.

header by the browser, no information is leaked as the external site will not be able to use the opaque string to associate with the user and thus their PII.

In some cases Facebook inserts a '#' character before the id field in its Request-URI. Since some browsers only retain the portion before '#' in a URI to be used in **Referer** headers and such, this may reduce chances of leakage. However, as our examples have shown, Facebook does not consistently follow this technique; even when consistently followed, other (non-**Referer** header related) leakage mechanisms outlined will continue to occur.

The fourth party, external applications, allow the OSN identifier to be passed through to external aggregators. They could use one of the methods outlined above to strip the id or remap it internally.

## 6. LEAKAGE VIA NON-OSN SITES

Although we focus on OSNs in this study, it should be obvious that the manner of leakage could affect users who have accounts and PII on other sites. Sites related to e-commerce, travel, and news services, maintain information about registered users. Some of these sites do use transient session-specific identifiers, which are less prone to identifying an individual compared with persistent identifiers of OSNs. Yet, the sites may embed pieces of PII such as email addresses and location within cookies or Request-URIs.

We have carried out a *preliminary* examination of several popular commercial sites for which we have readily available access. These include books, newspaper, travel, micropayment, and e-commerce sites. We identified a news site that leaks user email addresses to at least three separate third-party aggregators. A travel site embeds a user's first name and default airport in its cookies, which is therefore leaked to any third-party server hiding within the domain name of the travel site. By and large we did *not* observe leakage of user's login identifier via the **Referer** header, the Cookie, or the Request-URI. It should be noted that even if the user's identifier had leaked, the associated profile information about the user will not be available to the aggregator without the corresponding password.

Our preliminary examination should not be taken as the final answer on this issue. A thorough understanding of the scope of the problem along with steps for preventing leakage in general remains a primary concern. Any protection technique must effectively ensure de-identification between a user's identity prior to any external communication on any site that requires logging in—OSN or otherwise.

## 7. CONCLUSION

The results of our study clearly show that the indirect leakage of PII via OSN identifiers to third-party aggregation servers is happening. OSNs in our study consistently demonstrate leakage of user identifier information to one or more third-parties via Request-URIs, **Referer** headers and cookies. In addition, two of the OSNs directly leak pieces of PII to third parties with one of the OSNs leaking zip code and email information about users that may not be even publicly available within the OSN itself. We also observe that this leakage extends to external OSN applications, which not only have access to user profile information, but leak a user's OSN identifier to other third parties. It should be noted that there may be private contractual agreements

between aggregators and OSNs that forbid aggregators from using any information they may receive as a result of user's interaction with an OSN.

OSNs are in the best position to prevent such leakage by eliminating OSN identifiers from the Request-URI and consequently the **Referer** header. This elimination can be done directly or by mapping an OSN identifier to a session-specific value. Users have some means for limiting PII leakage via what information they provide to the OSN or browser/proxy techniques to control use of the **Referer** header and cookies. However, these controls may break accesses to other sites or not completely eliminate PII leakage via OSNs.

A clear direction for future work is to understand the bigger picture of PII leakage to third parties. We have performed a preliminary examination of PII leakage for non-OSN sites and found a couple of instances where pieces of PII were leaked to third-parties. We plan to undertake a more extensive examination of this issue along with steps that can be taken to prevent leakage of private information.

## Acknowledgments

We would like to thank Steven Bellovin, Graham Cormode, Jeff Mogul, Raj Savoor, Josh Elman, and the anonymous reviewers for their comments.

## 8. REFERENCES

- [1] James Abbatiello. Refcontrol. Firefox Add-on. <https://addons.mozilla.org/en-US/firefox/addon/953>.
- [2] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext Transfer Protocol — HTTP/1.0. RFC 1945, IETF, May 1996. Defines current usage of HTTP/1.0. <http://www.rfc-editor.org/rfc/rfc1945.txt>.
- [3] The cafes: Privacy tip #3: Block referer headers in Firefox, October 2006. <http://cafe.elharo.com/privacy/privacy-tip-3-block-referer-headers-in-firefox/>.
- [4] Online profiling: A report to congress, July 2000. Federal Trade Commission. <http://www.ftc.gov/os/2000/07/onlineprofiling.htm>.
- [5] Clay Johnson III. Safeguarding against and responding to the breach of personally identifiable information, May 22 2007. Office of Management and Budget Memorandum. <http://www.whitehouse.gov/omb/memoranda/fy2007/m07-16.pdf>.
- [6] Balachander Krishnamurthy and Craig E. Wills. Characterizing privacy in online social networks. In *Proceedings of the Workshop on Online Social Networks*, pages 37–42, Seattle, WA USA, August 2008. ACM.
- [7] Balachander Krishnamurthy and Craig E. Wills. Privacy diffusion on the web: A longitudinal perspective. In *Procs World Wide Web Conference, Madrid, Spain*, April 2009. <http://www.research.att.com/~bala/papers/www09.pdf>.
- [8] Bradley Malin. Betrayed by my shadow: Learning data identify via trail matching. *Journal of Privacy Technology*, June 2005.
- [9] Erika McCallister, Tim Grance, and Karen Scanfone. Guide to protecting the confidentiality of personally identifiable information (PII) (draft), January 2009. NIST Special Publication 800-122. <http://csrc.nist.gov/publications/drafts/800-122/Draft-SP800-122.pdf>.
- [10] Privacy guard. Facebook Application. <http://apps.facebook.com/privacyguard/>.
- [11] Privacy protector. Facebook Application. <http://apps.facebook.com/privacyprotector/>.
- [12] Privoxy. <http://www.privoxy.org/>.
- [13] Proxify anonymous proxy. <http://proxify.com/>.
- [14] Daniel Savard. LiveHTTPHeaders. Firefox Add-on. <http://livehttpheaders.mozdev.org/>.

Network Working Group  
Request for Comments: 2616  
Obsoletes: 2068  
Category: Standards Track

R. Fielding  
UC Irvine  
J. Gettys  
Compaq/W3C  
J. C. Mogul  
Compaq  
H. Frystyk  
W3C/MIT  
L. Masinter  
Xerox  
P. Leach  
Microsoft  
T. Berners-Lee  
W3C/MIT  
June, 1999

# Hypertext Transfer Protocol -- HTTP/1.1

## Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

## Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers [47]. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. This specification defines the protocol referred to as "HTTP/1.1", and is an update to RFC 2068 [33].

# Table of Contents

<b>HYPertext TRANSFER PROTOCOL -- HTTP/1.1.....</b>	<b>1</b>
<b>Status of this Memo .....</b>	<b>1</b>
<b>Copyright Notice.....</b>	<b>1</b>
<b>Abstract .....</b>	<b>1</b>
<b>Table of Contents.....</b>	<b>2</b>
<b>1 Introduction .....</b>	<b>7</b>
1.1 Purpose .....	7
1.2 Requirements .....	7
1.3 Terminology .....	8
1.4 Overall Operation .....	10
<b>2 Notational Conventions and Generic Grammar .....</b>	<b>11</b>
2.1 Augmented BNF.....	11
2.2 Basic Rules .....	12
<b>3 Protocol Parameters .....</b>	<b>13</b>
3.1 HTTP Version .....	13
3.2 Uniform Resource Identifiers.....	14
3.2.1 General Syntax.....	14
3.2.2 http URL.....	14
3.2.3 URI Comparison .....	15
3.3 Date/Time Formats .....	15
3.3.1 Full Date .....	15
3.3.2 Delta Seconds .....	16
3.4 Character Sets .....	16
3.4.1 Missing Charset .....	16
3.5 Content Codings .....	16
3.6 Transfer Codings .....	17
3.6.1 Chunked Transfer Coding.....	18
3.7 Media Types .....	18
3.7.1 Canonicalization and Text Defaults .....	19
3.7.2 Multipart Types.....	19
3.8 Product Tokens .....	20
3.9 Quality Values .....	20
3.10 Language Tags.....	20
3.11 Entity Tags.....	20
3.12 Range Units .....	21
<b>4 HTTP Message.....</b>	<b>21</b>
4.1 Message Types.....	21
4.2 Message Headers .....	21
4.3 Message Body.....	22
4.4 Message Length .....	23
4.5 General Header Fields .....	23

<b>5</b>	<b>Request .....</b>	<b>24</b>
5.1	Request-Line .....	24
5.1.1	Method .....	24
5.1.2	Request-URI .....	24
5.2	The Resource Identified by a Request .....	25
5.3	Request Header Fields .....	26
<b>6</b>	<b>Response .....</b>	<b>26</b>
6.1	Status-Line .....	26
6.1.1	Status Code and Reason Phrase .....	26
6.2	Response Header Fields .....	28
<b>7</b>	<b>Entity .....</b>	<b>28</b>
7.1	Entity Header Fields .....	28
7.2	Entity Body .....	29
7.2.1	Type .....	29
7.2.2	Entity Length .....	29
<b>8</b>	<b>Connections .....</b>	<b>29</b>
8.1	Persistent Connections .....	29
8.1.1	Purpose .....	29
8.1.2	Overall Operation .....	30
8.1.3	Proxy Servers .....	31
8.1.4	Practical Considerations .....	31
8.2	Message Transmission Requirements .....	31
8.2.1	Persistent Connections and Flow Control .....	31
8.2.2	Monitoring Connections for Error Status Messages .....	31
8.2.3	Use of the 100 (Continue) Status .....	32
8.2.4	Client Behavior if Server Prematurely Closes Connection .....	33
<b>9</b>	<b>Method Definitions .....</b>	<b>33</b>
9.1	Safe and Idempotent Methods .....	33
9.1.1	Safe Methods .....	33
9.1.2	Idempotent Methods .....	34
9.2	OPTIONS .....	34
9.3	GET .....	35
9.4	HEAD .....	35
9.5	POST .....	35
9.6	PUT .....	36
9.7	DELETE .....	36
9.8	TRACE .....	37
9.9	CONNECT .....	37
<b>10</b>	<b>Status Code Definitions .....</b>	<b>37</b>
10.1	Informational 1xx .....	37
10.1.1	100 Continue .....	37
10.1.2	101 Switching Protocols .....	38
10.2	Successful 2xx .....	38
10.2.1	200 OK .....	38
10.2.2	201 Created .....	38
10.2.3	202 Accepted .....	38
10.2.4	203 Non-Authoritative Information .....	39
10.2.5	204 No Content .....	39
10.2.6	205 Reset Content .....	39
10.2.7	206 Partial Content .....	39



10.3	Redirection 3xx.....	40
10.3.1	300 Multiple Choices.....	40
10.3.2	301 Moved Permanently.....	40
10.3.3	302 Found.....	40
10.3.4	303 See Other.....	41
10.3.5	304 Not Modified.....	41
10.3.6	305 Use Proxy.....	41
10.3.7	306 (Unused).....	41
10.3.8	307 Temporary Redirect.....	42
10.4	Client Error 4xx.....	42
10.4.1	400 Bad Request.....	42
10.4.2	401 Unauthorized.....	42
10.4.3	402 Payment Required.....	42
10.4.4	403 Forbidden.....	42
10.4.5	404 Not Found.....	43
10.4.6	405 Method Not Allowed.....	43
10.4.7	406 Not Acceptable.....	43
10.4.8	407 Proxy Authentication Required.....	43
10.4.9	408 Request Timeout.....	43
10.4.10	409 Conflict.....	43
10.4.11	410 Gone.....	44
10.4.12	411 Length Required.....	44
10.4.13	412 Precondition Failed.....	44
10.4.14	413 Request Entity Too Large.....	44
10.4.15	414 Request-URI Too Long.....	44
10.4.16	415 Unsupported Media Type.....	44
10.4.17	416 Requested Range Not Satisfiable.....	44
10.4.18	417 Expectation Failed.....	45
10.5	Server Error 5xx.....	45
10.5.1	500 Internal Server Error.....	45
10.5.2	501 Not Implemented.....	45
10.5.3	502 Bad Gateway.....	45
10.5.4	503 Service Unavailable.....	45
10.5.5	504 Gateway Timeout.....	45
10.5.6	505 HTTP Version Not Supported.....	45
<b>11</b>	<b>Access Authentication.....</b>	<b>46</b>
<b>12</b>	<b>Content Negotiation.....</b>	<b>46</b>
12.1	Server-driven Negotiation.....	46
12.2	Agent-driven Negotiation.....	47
12.3	Transparent Negotiation.....	47
<b>13</b>	<b>Caching in HTTP.....</b>	<b>47</b>
13.1.1	Cache Correctness.....	48
13.1.2	Warnings.....	49
13.1.3	Cache-control Mechanisms.....	49
13.1.4	Explicit User Agent Warnings.....	49
13.1.5	Exceptions to the Rules and Warnings.....	50
13.1.6	Client-controlled Behavior.....	50
13.2	Expiration Model.....	50
13.2.1	Server-Specified Expiration.....	50
13.2.2	Heuristic Expiration.....	51
13.2.3	Age Calculations.....	51
13.2.4	Expiration Calculations.....	52

13.2.5	Disambiguating Expiration Values .....	53
13.2.6	Disambiguating Multiple Responses .....	53
13.3	Validation Model .....	53
13.3.1	Last-Modified Dates .....	54
13.3.2	Entity Tag Cache Validators .....	54
13.3.3	Weak and Strong Validators .....	54
13.3.4	Rules for When to Use Entity Tags and Last-Modified Dates .....	56
13.3.5	Non-validating Conditionals .....	57
13.4	Response Cacheability .....	57
13.5	Constructing Responses From Caches .....	57
13.5.1	End-to-end and Hop-by-hop Headers .....	58
13.5.2	Non-modifiable Headers .....	58
13.5.3	Combining Headers .....	59
13.5.4	Combining Byte Ranges .....	59
13.6	Caching Negotiated Responses .....	60
13.7	Shared and Non-Shared Caches .....	60
13.8	Errors or Incomplete Response Cache Behavior .....	61
13.9	Side Effects of GET and HEAD .....	61
13.10	Invalidation After Updates or Deletions .....	61
13.11	Write-Through Mandatory .....	61
13.12	Cache Replacement .....	62
13.13	History Lists .....	62
<b>14</b>	<b>Header Field Definitions .....</b>	<b>62</b>
14.1	Accept .....	62
14.2	Accept-Charset .....	64
14.3	Accept-Encoding .....	64
14.4	Accept-Language .....	65
14.5	Accept-Ranges .....	66
14.6	Age .....	66
14.7	Allow .....	66
14.8	Authorization .....	66
14.9	Cache-Control .....	67
14.9.1	What is Cacheable .....	68
14.9.2	What May be Stored by Caches .....	69
14.9.3	Modifications of the Basic Expiration Mechanism .....	69
14.9.4	Cache Revalidation and Reload Controls .....	70
14.9.5	No-Transform Directive .....	72
14.9.6	Cache Control Extensions .....	72
14.10	Connection .....	72
14.11	Content-Encoding .....	73
14.12	Content-Language .....	73
14.13	Content-Length .....	74
14.14	Content-Location .....	74
14.15	Content-MD5 .....	75
14.16	Content-Range .....	75
14.17	Content-Type .....	77
14.18	Date .....	77
14.18.1	Clockless Origin Server Operation .....	78
14.19	ETag .....	78
14.20	Expect .....	78
14.21	Expires .....	78
14.22	From .....	79
14.23	Host .....	79
14.24	If-Match .....	80

14.25	If-Modified-Since .....	80
14.26	If-None-Match .....	81
14.27	If-Range .....	82
14.28	If-Unmodified-Since .....	82
14.29	Last-Modified .....	83
14.30	Location .....	83
14.31	Max-Forwards.....	83
14.32	Pragma .....	84
14.33	Proxy-Authenticate .....	84
14.34	Proxy-Authorization .....	85
14.35	Range .....	85
14.35.1	Byte Ranges.....	85
14.35.2	Range Retrieval Requests .....	86
14.36	Referer .....	86
14.37	Retry-After.....	87
14.38	Server.....	87
14.39	TE .....	87
14.40	Trailer .....	88
14.41	Transfer-Encoding .....	88
14.42	Upgrade .....	88
14.43	User-Agent.....	89
14.44	Vary .....	89
14.45	Via .....	90
14.46	Warning .....	91
14.47	WWW-Authenticate .....	92
<b>15</b>	<b>Security Considerations .....</b>	<b>92</b>
15.1	Personal Information.....	92
15.1.1	Abuse of Server Log Information .....	93
15.1.2	Transfer of Sensitive Information .....	93
15.1.3	Encoding Sensitive Information in URI's .....	93
15.1.4	Privacy Issues Connected to Accept Headers .....	94
15.2	Attacks Based On File and Path Names.....	94
15.3	DNS Spoofing.....	94
15.4	Location Headers and Spoofing.....	95
15.5	Content-Disposition Issues .....	95
15.6	Authentication Credentials and Idle Clients.....	95
15.7	Proxies and Caching .....	95
15.7.1	Denial of Service Attacks on Proxies.....	96
<b>16</b>	<b>Acknowledgments .....</b>	<b>96</b>
<b>17</b>	<b>References.....</b>	<b>97</b>
<b>18</b>	<b>Authors' Addresses.....</b>	<b>99</b>
<b>19</b>	<b>Appendices.....</b>	<b>100</b>
19.1	Internet Media Type message/http and application/http .....	100
19.2	Internet Media Type multipart/byteranges.....	101
19.3	Tolerant Applications .....	102
19.4	Differences Between HTTP Entities and RFC 2045 Entities.....	102
19.4.1	MIME-Version.....	102
19.4.2	Conversion to Canonical Form .....	103
19.4.3	Conversion of Date Formats .....	103
19.4.4	Introduction of Content-Encoding .....	103

19.4.5	No Content-Transfer-Encoding.....	103
19.4.6	Introduction of Transfer-Encoding .....	103
19.4.7	MHTML and Line Length Limitations .....	104
19.5	Additional Features.....	104
19.5.1	Content-Disposition .....	104
19.6	Compatibility with Previous Versions .....	105
19.6.1	Changes from HTTP/1.0.....	105
19.6.2	Compatibility with HTTP/1.0 Persistent Connections .....	105
19.6.3	Changes from RFC 2068.....	106
<b>20</b>	<b>Full Copyright Statement.....</b>	<b>108</b>
20.1	Acknowledgement .....	108
<b>21</b>	<b>Index .....</b>	<b>109</b>

# 1 Introduction

## 1.1 Purpose

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. HTTP has been in use by the World-Wide Web global information initiative since 1990. The first version of HTTP, referred to as HTTP/0.9, was a simple protocol for raw data transfer across the Internet. HTTP/1.0, as defined by RFC 1945 [6], improved the protocol by allowing messages to be in the format of MIME-like messages, containing meta-information about the data transferred and modifiers on the request/response semantics. However, HTTP/1.0 does not sufficiently take into consideration the effects of hierarchical proxies, caching, the need for persistent connections, or virtual hosts. In addition, the proliferation of incompletely-implemented applications calling themselves “HTTP/1.0” has necessitated a protocol version change in order for two communicating applications to determine each other’s true capabilities.

This specification defines the protocol referred to as “HTTP/1.1”. This protocol includes more stringent requirements than HTTP/1.0 in order to ensure reliable implementation of its features.

Practical information systems require more functionality than simple retrieval, including search, front-end update, and annotation. HTTP allows an open-ended set of methods and headers that indicate the purpose of a request [47]. It builds on the discipline of reference provided by the Uniform Resource Identifier (URI) [3], as a location (URL) [4] or name (URN) [20], for indicating the resource to which a method is to be applied. Messages are passed in a format similar to that used by Internet mail [9] as defined by the Multipurpose Internet Mail Extensions (MIME) [7].

HTTP is also used as a generic protocol for communication between user agents and proxies/gateways to other Internet systems, including those supported by the SMTP [16], NNTP [13], FTP [18], Gopher [2], and WAIS [10] protocols. In this way, HTTP allows basic hypermedia access to resources available from diverse applications.

## 1.2 Requirements

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [34].

An implementation is not compliant if it fails to satisfy one or more of the MUST or REQUIRED level requirements for the protocols it implements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements for its protocols is said to be “unconditionally compliant”; one that satisfies all the MUST level requirements but not all the SHOULD level requirements for its protocols is said to be “conditionally compliant.”

## 1.3 Terminology

This specification uses a number of terms to refer to the roles played by participants in, and objects of, the HTTP communication.

### connection

A transport layer virtual circuit established between two programs for the purpose of communication.

### message

The basic unit of HTTP communication, consisting of a structured sequence of octets matching the syntax defined in section 4 and transmitted via the connection.

### request

An HTTP request message, as defined in section 5.

### response

An HTTP response message, as defined in section 6.

### resource

A network data object or service that can be identified by a URI, as defined in section 3.2. Resources may be available in multiple representations (e.g. multiple languages, data formats, size, and resolutions) or vary in other ways.

### entity

The information transferred as the payload of a request or response. An entity consists of meta-information in the form of entity-header fields and content in the form of an entity-body, as described in section 7.

### representation

An entity included with a response that is subject to content negotiation, as described in section 12. There may exist multiple representations associated with a particular response status.

### content negotiation

The mechanism for selecting the appropriate representation when servicing a request, as described in section 12. The representation of entities in any response can be negotiated (including error responses).

### variant

A resource may have one, or more than one, representation(s) associated with it at any given instant. Each of these representations is termed a 'variant.' Use of the term 'variant' does not necessarily imply that the resource is subject to content negotiation.

### client

A program that establishes connections for the purpose of sending requests.

### user agent

The client which initiates a request. These are often browsers, editors, spiders (web-traversing robots), or other end user tools.

### server

An application program that accepts connections in order to service requests by sending back responses. Any given program may be capable of being both a client and a server; our use of these terms refers only to the role being performed by the program for a particular connection, rather than to the program's capabilities in general. Likewise, any server may act as an origin server, proxy, gateway, or tunnel, switching behavior based on the nature of each request.

### origin server

The server on which a given resource resides or is to be created.

**proxy**

An intermediary program which acts as both a server and a client for the purpose of making requests on behalf of other clients. Requests are serviced internally or by passing them on, with possible translation, to other servers. A proxy **MUST** implement both the client and server requirements of this specification. A “transparent proxy” is a proxy that does not modify the request or response beyond what is required for proxy authentication and identification. A “non-transparent proxy” is a proxy that modifies the request or response in order to provide some added service to the user agent, such as group annotation services, media type transformation, protocol reduction, or anonymity filtering. Except where either transparent or non-transparent behavior is explicitly stated, the HTTP proxy requirements apply to both types of proxies.

**gateway**

A server which acts as an intermediary for some other server. Unlike a proxy, a gateway receives requests as if it were the origin server for the requested resource; the requesting client may not be aware that it is communicating with a gateway.

**tunnel**

An intermediary program which is acting as a blind relay between two connections. Once active, a tunnel is not considered a party to the HTTP communication, though the tunnel may have been initiated by an HTTP request. The tunnel ceases to exist when both ends of the relayed connections are closed.

**cache**

A program’s local store of response messages and the subsystem that controls its message storage, retrieval, and deletion. A cache stores cacheable responses in order to reduce the response time and network bandwidth consumption on future, equivalent requests. Any client or server may include a cache, though a cache cannot be used by a server that is acting as a tunnel.

**cacheable**

A response is cacheable if a cache is allowed to store a copy of the response message for use in answering subsequent requests. The rules for determining the cacheability of HTTP responses are defined in section 13. Even if a resource is cacheable, there may be additional constraints on whether a cache can use the cached copy for a particular request.

**first-hand**

A response is first-hand if it comes directly and without unnecessary delay from the origin server, perhaps via one or more proxies. A response is also first-hand if its validity has just been checked directly with the origin server.

**explicit expiration time**

The time at which the origin server intends that an entity should no longer be returned by a cache without further validation.

**heuristic expiration time**

An expiration time assigned by a cache when no explicit expiration time is available.

**age**

The age of a response is the time since it was sent by, or successfully validated with, the origin server.

**freshness lifetime**

The length of time between the generation of a response and its expiration time.

**fresh**

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

**stale**

A response is stale if its age has passed its freshness lifetime.

semantically transparent

A cache behaves in a “semantically transparent” manner, with respect to a particular response, when its use affects neither the requesting client nor the origin server, except to improve performance. When a cache is semantically transparent, the client receives exactly the same response (except for hop-by-hop headers) that it would have received had its request been handled directly by the origin server.

validator

A protocol element (e.g., an entity tag or a Last-Modified time) that is used to find out whether a cache entry is an equivalent copy of an entity.

upstream/downstream

Upstream and downstream describe the flow of a message: all messages flow from upstream to downstream.

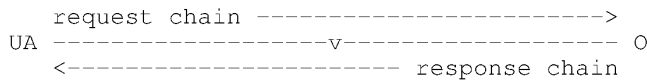
inbound/outbound

Inbound and outbound refer to the request and response paths for messages: “inbound” means “traveling toward the origin server”, and “outbound” means “traveling toward the user agent”

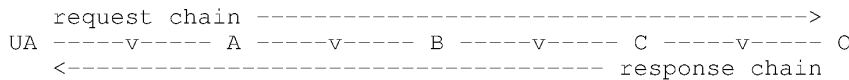
### 1.4 Overall Operation

The HTTP protocol is a request/response protocol. A client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message’s protocol version and a success or error code, followed by a MIME-like message containing server information, entity metainformation, and possible entity-body content. The relationship between HTTP and MIME is described in appendix 19.4.

Most HTTP communication is initiated by a user agent and consists of a request to be applied to a resource on some origin server. In the simplest case, this may be accomplished via a single connection (v) between the user agent (UA) and the origin server (O).

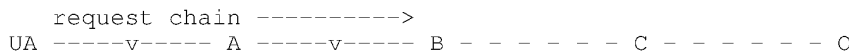


A more complicated situation occurs when one or more intermediaries are present in the request/response chain. There are three common forms of intermediary: proxy, gateway, and tunnel. A proxy is a forwarding agent, receiving requests for a URI in its absolute form, rewriting all or part of the message, and forwarding the reformatted request toward the server identified by the URI. A gateway is a receiving agent, acting as a layer above some other server(s) and, if necessary, translating the requests to the underlying server’s protocol. A tunnel acts as a relay point between two connections without changing the messages; tunnels are used when the communication needs to pass through an intermediary (such as a firewall) even when the intermediary cannot understand the contents of the messages.



The figure above shows three intermediaries (A, B, and C) between the user agent and origin server. A request or response message that travels the whole chain will pass through four separate connections. This distinction is important because some HTTP communication options may apply only to the connection with the nearest, non-tunnel neighbor, only to the end-points of the chain, or to all connections along the chain. Although the diagram is linear, each participant may be engaged in multiple, simultaneous communications. For example, B may be receiving requests from many clients other than A, and/or forwarding requests to servers other than C, at the same time that it is handling A’s request.

Any party to the communication which is not acting as a tunnel may employ an internal cache for handling requests. The effect of a cache is that the request/response chain is shortened if one of the participants along the chain has a cached response applicable to that request. The following illustrates the resulting chain if B has a cached copy of an earlier response from O (via C) for a request which has not been cached by UA or A.



<----- response chain

Not all responses are usefully cacheable, and some requests may contain modifiers which place special requirements on cache behavior. HTTP requirements for cache behavior and cacheable responses are defined in section 13.

In fact, there are a wide variety of architectures and configurations of caches and proxies currently being experimented with or deployed across the World Wide Web. These systems include national hierarchies of proxy caches to save transoceanic bandwidth, systems that broadcast or multicast cache entries, organizations that distribute subsets of cached data via CD-ROM, and so on. HTTP systems are used in corporate intranets over high-bandwidth links, and for access via PDAs with low-power radio links and intermittent connectivity. The goal of HTTP/1.1 is to support the wide diversity of configurations already deployed while introducing protocol constructs that meet the needs of those who build web applications that require high reliability and, failing that, at least reliable indications of failure.

HTTP communication usually takes place over TCP/IP connections. The default port is TCP 80 [19], but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used; the mapping of the HTTP/1.1 request and response structures onto the transport data units of the protocol in question is outside the scope of this specification.

In HTTP/1.0, most implementations used a new connection for each request/response exchange. In HTTP/1.1, a connection may be used for one or more request/response exchanges, although connections may be closed for a variety of reasons (see section 8.1).

## 2 Notational Conventions and Generic Grammar

### 2.1 Augmented BNF

All of the mechanisms specified in this document are described in both prose and an augmented Backus-Naur Form (BNF) similar to that used by RFC 822 [9]. Implementors will need to be familiar with the notation in order to understand this specification. The augmented BNF includes the following constructs:

`name = definition`

The name of a rule is simply the name itself (without any enclosing "<" and ">") and is separated from its definition by the equal "=" character. White space is only significant in that indentation of continuation lines is used to indicate a rule definition that spans more than one line. Certain basic rules are in uppercase, such as `SP`, `LWS`, `HT`, `CRLF`, `DIGIT`, `ALPHA`, etc. Angle brackets are used within definitions whenever their presence will facilitate discerning the use of rule names.

`"literal"`

Quotation marks surround literal text. Unless stated otherwise, the text is case-insensitive.

`rule1 | rule2`

Elements separated by a bar ("|") are alternatives, e.g., "yes | no" will accept yes or no.

`(rule1 rule2)`

Elements enclosed in parentheses are treated as a single element. Thus, "(elem (foo | bar) elem)" allows the token sequences "elem foo elem" and "elem bar elem".

`*rule`

The character "\*" preceding an element indicates repetition. The full form is "<n>\*<m>element" indicating at least <n> and at most <m> occurrences of element. Default values are 0 and infinity so that "(element)" allows any number, including zero; "1\*element" requires at least one; and "1\*2element" allows one or two.

`[rule]`

Square brackets enclose optional elements; "[foo bar]" is equivalent to "\*1(foo bar)".



**N rule**

Specific repetition: “<n> (element)” is equivalent to “<n>\*<n> (element)”; that is, exactly <n> occurrences of (element). Thus 2DIGIT is a 2-digit number, and 3ALPHA is a string of three alphabetic characters.

**#rule**

A construct “#” is defined, similar to “\*”, for defining lists of elements. The full form is “<n>#<m>element” indicating at least <n> and at most <m> elements, each separated by one or more commas (“,”) and OPTIONAL linear white space (LWS). This makes the usual form of lists very easy; a rule such as

```
( *LWS element *( *LWS "," *LWS element ) )
```

can be shown as

```
1#element
```

Wherever this construct is used, null elements are allowed, but do not contribute to the count of elements present. That is, “(element), , (element)” is permitted, but counts as only two elements. Therefore, where at least one element is required, at least one non-null element MUST be present. Default values are 0 and infinity so that “#element” allows any number, including zero; “1#element” requires at least one; and “1#2element” allows one or two.

**; comment**

A semi-colon, set off some distance to the right of rule text, starts a comment that continues to the end of line. This is a simple way of including useful notes in parallel with the specifications.

**implied \*LWS**

The grammar described by this specification is word-based. Except where noted otherwise, linear white space (LWS) can be included between any two adjacent words (token or quoted-string), and between adjacent words and separators, without changing the interpretation of a field. At least one delimiter (LWS and/or separators) MUST exist between any two tokens (for the definition of “token” below), since they would otherwise be interpreted as a single token.

## 2.2 Basic Rules

The following rules are used throughout this specification to describe basic parsing constructs. The US-ASCII coded character set is defined by ANSI X3.4-1986 [21].

```
OCTET      = <any 8-bit sequence of data>
CHAR       = <any US-ASCII character (octets 0 - 127)>
UPALPHA   = <any US-ASCII uppercase letter "A".."Z">
LOALPHA   = <any US-ASCII lowercase letter "a".."z">
ALPHA     = UPALPHA | LOALPHA
DIGIT     = <any US-ASCII digit "0".."9">
CTL       = <any US-ASCII control character
            (octets 0 - 31) and DEL (127)>
CR        = <US-ASCII CR, carriage return (13)>
LF        = <US-ASCII LF, linefeed (10)>
SP        = <US-ASCII SP, space (32)>
HT        = <US-ASCII HT, horizontal-tab (9)>
">"      = <US-ASCII double-quote mark (34)>
```

HTTP/1.1 defines the sequence CR LF as the end-of-line marker for all protocol elements except the entity-body (see appendix 19.3 for tolerant applications). The end-of-line marker within an entity-body is defined by its associated media type, as described in section 3.7.

```
CRLF      = CR LF
```

HTTP/1.1 header field values can be folded onto multiple lines if the continuation line begins with a space or horizontal tab. All linear white space, including folding, has the same semantics as SP. A recipient MAY replace any linear white space with a single SP before interpreting the field value or forwarding the message downstream.

```
LWS       = [CRLF] 1*( SP | HT )
```

The TEXT rule is only used for descriptive field contents and values that are not intended to be interpreted by the message parser. Words of \*TEXT MAY contain characters from character sets other than ISO-8859-1 [22] only when encoded according to the rules of RFC 2047 [14].

```
TEXT          = <any OCTET except CTLs,
                but including LWS>
```

A CRLF is allowed in the definition of TEXT only as part of a header field continuation. It is expected that the folding LWS will be replaced with a single SP before interpretation of the TEXT value.

Hexadecimal numeric characters are used in several protocol elements.

```
HEX          = "A" | "B" | "C" | "D" | "E" | "F"
              | "a" | "b" | "c" | "d" | "e" | "f" | DIGIT
```

Many HTTP/1.1 header field values consist of words separated by LWS or special characters. These special characters MUST be in a quoted string to be used within a parameter value (as defined in section 3.6).

```
token        = 1*<any CHAR except CTLs or separators>
separators   = "(" | ")" | "<" | ">" | "@"
              | "," | ";" | ":" | "\" | <">
              | "/" | "[" | "]" | "?" | "="
              | "{" | "}" | SP | HT
```

Comments can be included in some HTTP header fields by surrounding the comment text with parentheses. Comments are only allowed in fields containing "comment" as part of their field value definition. In all other fields, parentheses are considered part of the field value.

```
comment      = "(" *( ctext | quoted-pair | comment ) ")"
ctext        = <any TEXT excluding "(" and ">">
```

A string of text is parsed as a single word if it is quoted using double-quote marks.

```
quoted-string = ( <"> *( qdtext | quoted-pair ) <"> )
qdtext       = <any TEXT except <">>
```

The backslash character ("\") MAY be used as a single-character quoting mechanism only within quoted-string and comment constructs.

```
quoted-pair  = "\" CHAR
```

## 3 Protocol Parameters

### 3.1 HTTP Version

HTTP uses a "<major>.<minor>" numbering scheme to indicate versions of the protocol. The protocol versioning policy is intended to allow the sender to indicate the format of a message and its capacity for understanding further HTTP communication, rather than the features obtained via that communication. No change is made to the version number for the addition of message components which do not affect communication behavior or which only add to extensible field values. The <minor> number is incremented when the changes made to the protocol add features which do not change the general message parsing algorithm, but which may add to the message semantics and imply additional capabilities of the sender. The <major> number is incremented when the format of a message within the protocol is changed. See RFC 2145 [36] for a fuller explanation.

The version of an HTTP message is indicated by an HTTP-Version field in the first line of the message.

```
HTTP-Version = "HTTP" "/" 1*DIGIT "." 1*DIGIT
```

Note that the major and minor numbers MUST be treated as separate integers and that each MAY be incremented higher than a single digit. Thus, HTTP/2.4 is a lower version than HTTP/2.13, which in turn is lower than HTTP/12.3. Leading zeros MUST be ignored by recipients and MUST NOT be sent.

An application that sends a request or response message that includes HTTP-Version of "HTTP/1.1" MUST be at least conditionally compliant with this specification. Applications that are at least conditionally compliant with this specification SHOULD use an HTTP-Version of "HTTP/1.1" in their messages, and MUST do so for any message

that is not compatible with HTTP/1.0. For more details on when to send specific `HTTP-Version` values, see RFC 2145 [36].

The HTTP version of an application is the highest HTTP version for which the application is at least conditionally compliant.

Proxy and gateway applications need to be careful when forwarding messages in protocol versions different from that of the application. Since the protocol version indicates the protocol capability of the sender, a proxy/gateway **MUST NOT** send a message with a version indicator which is greater than its actual version. If a higher version request is received, the proxy/gateway **MUST** either downgrade the request version, or respond with an error, or switch to tunnel behavior.

Due to interoperability problems with HTTP/1.0 proxies discovered since the publication of RFC 2068[33], caching proxies **MUST**, gateways **MAY**, and tunnels **MUST NOT** upgrade the request to the highest version they support. The proxy/gateway's response to that request **MUST** be in the same major version as the request.

Note: Converting between versions of HTTP may involve modification of header fields required or forbidden by the versions involved.

## 3.2 Uniform Resource Identifiers

URIs have been known by many names: WWW addresses, Universal Document Identifiers, Universal Resource Identifiers [3], and finally the combination of Uniform Resource Locators (URL) [4] and Names (URN) [20]. As far as HTTP is concerned, Uniform Resource Identifiers are simply formatted strings which identify--via name, location, or any other characteristic--a resource.

### 3.2.1 General Syntax

URIs in HTTP can be represented in absolute form or relative to some known base URI [11], depending upon the context of their use. The two forms are differentiated by the fact that absolute URIs always begin with a scheme name followed by a colon. For definitive information on URL syntax and semantics, see "Uniform Resource Identifiers (URI): Generic Syntax and Semantics," RFC 2396 [42] (which replaces RFCs 1738 [4] and RFC 1808 [11]). This specification adopts the definitions of "URI-reference", "absoluteURI", "relativeURI", "port", "host", "abs\_path", "rel\_path", and "authority" from that specification.

The HTTP protocol does not place any *a priori* limit on the length of a URI. Servers **MUST** be able to handle the URI of any resource they serve, and **SHOULD** be able to handle URIs of unbounded length if they provide GET-based forms that could generate such URIs. A server **SHOULD** return 414 (Request-URI Too Long) status if a URI is longer than the server can handle (see section 10.4.15).

Note: Servers ought to be cautious about depending on URI lengths above 255 bytes, because some older client or proxy implementations might not properly support these lengths.

### 3.2.2 http URL

The "http" scheme is used to locate network resources via the HTTP protocol. This section defines the scheme-specific syntax and semantics for http URLs.

```
http_URL = "http:" "://" host [ ":" port ] [ abs_path [ "?" query ] ]
```

If the `port` is empty or not given, port 80 is assumed. The semantics are that the identified resource is located at the server listening for TCP connections on that `port` of that `host`, and the `Request-URI` for the resource is `abs_path` (section 5.1.2). The use of IP addresses in URLs **SHOULD** be avoided whenever possible (see RFC 1900 [24]). If the `abs_path` is not present in the URL, it **MUST** be given as "/" when used as a `Request-URI` for a resource (section 5.1.2). If a proxy receives a host name which is not a fully qualified domain name, it **MAY** add its domain to the host name it received. If a proxy receives a fully qualified domain name, the proxy **MUST NOT** change the host name.

### 3.2.3 URI Comparison

When comparing two URIs to decide if they match or not, a client **SHOULD** use a case-sensitive octet-by-octet comparison of the entire URIs, with these exceptions:

- A port that is empty or not given is equivalent to the default port for that `URI-reference`;
- Comparisons of host names **MUST** be case-insensitive;
- Comparisons of scheme names **MUST** be case-insensitive;
- An empty `abs_path` is equivalent to an `abs_path` of `"/"`.

Characters other than those in the "reserved" and "unsafe" sets (see RFC 2396 [42]) are equivalent to their "`%`" `HEX HEX`" encoding.

For example, the following three URIs are equivalent:

```
http://abc.com:80/~smith/home.html
http://ABC.com/%7Esmith/home.html
http://ABC.com:/%7esmith/home.html
```

## 3.3 Date/Time Formats

### 3.3.1 Full Date

HTTP applications have historically allowed three different formats for the representation of date/time stamps:

```
Sun, 06 Nov 1994 08:49:37 GMT ; RFC 822, updated by RFC 1123
Sunday, 06-Nov-94 08:49:37 GMT ; RFC 850, obsoleted by RFC 1036
Sun Nov 6 08:49:37 1994 ; ANSI C's asctime() format
```

The first format is preferred as an Internet standard and represents a fixed-length subset of that defined by RFC 1123 [8] (an update to RFC 822 [9]). The second format is in common use, but is based on the obsolete RFC 850 [12] date format and lacks a four-digit year. HTTP/1.1 clients and servers that parse the date value **MUST** accept all three formats (for compatibility with HTTP/1.0), though they **MUST** only generate the RFC 1123 format for representing HTTP-date values in header fields. See section 19.3 for further information.

Note: Recipients of date values are encouraged to be robust in accepting date values that may have been sent by non-HTTP applications, as is sometimes the case when retrieving or posting messages via proxies/gateways to SMTP or NNTP.

All HTTP date/time stamps **MUST** be represented in Greenwich Mean Time (GMT), without exception. For the purposes of HTTP, GMT is exactly equal to UTC (Coordinated Universal Time). This is indicated in the first two formats by the inclusion of "GMT" as the three-letter abbreviation for time zone, and **MUST** be assumed when reading the `asctime` format. HTTP-date is case sensitive and **MUST NOT** include additional LWS beyond that specifically included as SP in the grammar.

```
HTTP-date = rfc1123-date | rfc850-date | asctime-date
rfc1123-date = wkday "," SP date1 SP time SP "GMT"
rfc850-date = weekday "," SP date2 SP time SP "GMT"
asctime-date = wkday SP date3 SP time SP 4DIGIT
date1 = 2DIGIT SP month SP 4DIGIT
        ; day month year (e.g., 02 Jun 1982)
date2 = 2DIGIT "-" month "-" 2DIGIT
        ; day-month-year (e.g., 02-Jun-82)
date3 = month SP ( 2DIGIT | ( SP 1DIGIT ) )
        ; month day (e.g., Jun 2)
time = 2DIGIT ":" 2DIGIT ":" 2DIGIT
        ; 00:00:00 - 23:59:59
wkday = "Mon" | "Tue" | "Wed"
        | "Thu" | "Fri" | "Sat" | "Sun"
weekday = "Monday" | "Tuesday" | "Wednesday"
        | "Thursday" | "Friday" | "Saturday" | "Sunday"
month = "Jan" | "Feb" | "Mar" | "Apr"
        | "May" | "Jun" | "Jul" | "Aug"
```

| "Sep" | "Oct" | "Nov" | "Dec"

Note: HTTP requirements for the date/time stamp format apply only to their usage within the protocol stream. Clients and servers are not required to use these formats for user presentation, request logging, etc.

### 3.3.2 Delta Seconds

Some HTTP header fields allow a time value to be specified as an integer number of seconds, represented in decimal, after the time that the message was received.

`delta-seconds = 1 *DIGIT`

## 3.4 Character Sets

HTTP uses the same definition of the term “character set” as that described for MIME:

The term “character set” is used in this document to refer to a method used with one or more tables to convert a sequence of octets into a sequence of characters. Note that unconditional conversion in the other direction is not required, in that not all characters may be available in a given character set and a character set may provide more than one sequence of octets to represent a particular character. This definition is intended to allow various kinds of character encoding, from simple single-table mappings such as US-ASCII to complex table switching methods such as those that use ISO-2022’s techniques. However, the definition associated with a MIME character set name **MUST** fully specify the mapping to be performed from octets to characters. In particular, use of external profiling information to determine the exact mapping is not permitted.

Note: This use of the term “character set” is more commonly referred to as a “character encoding.” However, since HTTP and MIME share the same registry, it is important that the terminology also be shared.

HTTP character sets are identified by case-insensitive tokens. The complete set of tokens is defined by the IANA Character Set registry [19].

`charset = token`

Although HTTP allows an arbitrary token to be used as a charset value, any token that has a predefined value within the IANA Character Set registry [19] **MUST** represent the character set defined by that registry. Applications **SHOULD** limit their use of character sets to those defined by the IANA registry.

Implementors should be aware of IETF character set requirements [38] [41].

### 3.4.1 Missing Charset

Some HTTP/1.0 software has interpreted a `Content-Type` header without charset parameter incorrectly to mean “recipient should guess.” Senders wishing to defeat this behavior **MAY** include a charset parameter even when the charset is ISO-8859-1 and **SHOULD** do so when it is known that it will not confuse the recipient.

Unfortunately, some older HTTP/1.0 clients did not deal properly with an explicit charset parameter. HTTP/1.1 recipients **MUST** respect the charset label provided by the sender; and those user agents that have a provision to “guess” a charset **MUST** use the charset from the content-type field if they support that charset, rather than the recipient’s preference, when initially displaying a document. See section 3.7.1.

## 3.5 Content Codings

Content coding values indicate an encoding transformation that has been or can be applied to an entity. Content codings are primarily used to allow a document to be compressed or otherwise usefully transformed without losing the identity of its underlying media type and without loss of information. Frequently, the entity is stored in coded form, transmitted directly, and only decoded by the recipient.

`content-coding = token`

All content-coding values are case-insensitive. HTTP/1.1 uses content-coding values in the `Accept-Encoding` (section 14.3) and `Content-Encoding` (section 14.11) header fields. Although the value describes the content-

coding, what is more important is that it indicates what decoding mechanism will be required to remove the encoding.

The Internet Assigned Numbers Authority (IANA) acts as a registry for content-coding value tokens. Initially, the registry contains the following tokens:

`gzip` An encoding format produced by the file compression program “gzip” (GNU zip) as described in RFC 1952 [25]. This format is a Lempel-Ziv coding (LZ77) with a 32 bit CRC.

`compress`

The encoding format produced by the common UNIX file compression program “compress”. This format is an adaptive Lempel-Ziv-Welch coding (LZW).

Use of program names for the identification of encoding formats is not desirable and is discouraged for future encodings. Their use here is representative of historical practice, not good design. For compatibility with previous implementations of HTTP, applications SHOULD consider “x-gzip” and “x-compress” to be equivalent to “gzip” and “compress” respectively.

`deflate`

The “zlib” format defined in RFC 1950 [31] in combination with the “deflate” compression mechanism described in RFC 1951 [29].

`identity`

The default (identity) encoding; the use of no transformation whatsoever. This content-coding is used only in the `Accept-Encoding` header, and SHOULD NOT be used in the `Content-Encoding` header.

New content-coding value tokens SHOULD be registered; to allow interoperability between clients and servers, specifications of the content coding algorithms needed to implement a new value SHOULD be publicly available and adequate for independent implementation, and conform to the purpose of content coding defined in this section.

### 3.6 Transfer Codings

Transfer-coding values are used to indicate an encoding transformation that has been, can be, or may need to be applied to an entity-body in order to ensure “safe transport” through the network. This differs from a content coding in that the transfer-coding is a property of the message, not of the original entity.

```
transfer-coding      = "chunked" | transfer-extension
transfer-extension  = token *( ";" parameter )
```

Parameters are in the form of attribute/value pairs.

```
parameter           = attribute "=" value
attribute            = token
value                = token | quoted-string
```

All transfer-coding values are case-insensitive. HTTP/1.1 uses transfer-coding values in the `TE` header field (section 14.39) and in the `Transfer-Encoding` header field (section 14.41).

Whenever a transfer-coding is applied to a message-body, the set of transfer-codings MUST include “chunked”, unless the message is terminated by closing the connection. When the “chunked” transfer-coding is used, it MUST be the last transfer-coding applied to the message-body. The “chunked” transfer-coding MUST NOT be applied more than once to a message-body. These rules allow the recipient to determine the transfer-length of the message (section 4.4).

Transfer-codings are analogous to the `Content-Transfer-Encoding` values of MIME [7], which were designed to enable safe transport of binary data over a 7-bit transport service. However, safe transport has a different focus for an 8bit-clean transfer protocol. In HTTP, the only unsafe characteristic of message-bodies is the difficulty in determining the exact body length (section 7.2.2), or the desire to encrypt data over a shared transport.

The Internet Assigned Numbers Authority (IANA) acts as a registry for transfer-coding value tokens. Initially, the registry contains the following tokens: “chunked” (section 3.6.1), “identity” (section 3.6.2), “gzip” (section 3.5), “compress” (section 3.5), and “deflate” (section 3.5).

New transfer-coding value tokens SHOULD be registered in the same way as new content-coding value tokens (section 3.5).

A server which receives an entity-body with a transfer-coding it does not understand SHOULD return 501 (Unimplemented), and close the connection. A server MUST NOT send transfer-codings to an HTTP/1.0 client.

### 3.6.1 Chunked Transfer Coding

The chunked encoding modifies the body of a message in order to transfer it as a series of chunks, each with its own size indicator, followed by an OPTIONAL trailer containing entity-header fields. This allows dynamically produced content to be transferred along with the information necessary for the recipient to verify that it has received the full message.

```

Chunked-Body   = *chunk
                  last-chunk
                  trailer
                  CRLF
chunk          = chunk-size [ chunk-extension ] CRLF
                  chunk-data CRLF
chunk-size     = 1*HEX
last-chunk     = 1*("0") [ chunk-extension ] CRLF

chunk-extension= *( ";" chunk-ext-name [ "=" chunk-ext-val ] )
chunk-ext-name = token
chunk-ext-val  = token | quoted-string
chunk-data     = chunk-size(OCTET)
trailer        = *(entity-header CRLF)

```

The `chunk-size` field is a string of hex digits indicating the size of the chunk. The chunked encoding is ended by any chunk whose size is zero, followed by the trailer, which is terminated by an empty line.

The trailer allows the sender to include additional HTTP header fields at the end of the message. The `Trailer` header field can be used to indicate which header fields are included in a trailer (see section 14.40).

A server using chunked transfer-coding in a response MUST NOT use the trailer for any header fields unless at least one of the following is true:

- a) the request included a `TE` header field that indicates "trailers" is acceptable in the transfer-coding of the response, as described in section 14.39; or,
- b) the server is the origin server for the response, the trailer fields consist entirely of optional metadata, and the recipient could use the message (in a manner acceptable to the origin server) without receiving this metadata. In other words, the origin server is willing to accept the possibility that the trailer fields might be silently discarded along the path to the client.

This requirement prevents an interoperability failure when the message is being received by an HTTP/1.1 (or later) proxy and forwarded to an HTTP/1.0 recipient. It avoids a situation where compliance with the protocol would have necessitated a possibly infinite buffer on the proxy.

An example process for decoding a `Chunked-Body` is presented in appendix 19.4.6.

All HTTP/1.1 applications MUST be able to receive and decode the "chunked" transfer-coding, and MUST ignore `chunk-extension` extensions they do not understand.

## 3.7 Media Types

HTTP uses Internet Media Types [17] in the `Content-Type` (section 14.17) and `Accept` (section 14.1) header fields in order to provide open and extensible data typing and type negotiation.

```

media-type     = type "/" subtype *( ";" parameter )
type           = token
subtype        = token

```

Parameters MAY follow the type/subtype in the form of attribute/value pairs (as defined in section 3.6).

The type, subtype, and parameter attribute names are case-insensitive. Parameter values might or might not be case-sensitive, depending on the semantics of the parameter name. Linear white space (LWS) MUST NOT be used between the type and subtype, nor between an attribute and its value. The presence or absence of a parameter might be significant to the processing of a media-type, depending on its definition within the media type registry.

Note that some older HTTP applications do not recognize media type parameters. When sending data to older HTTP applications, implementations SHOULD only use media type parameters when they are required by that type/subtype definition.

Media-type values are registered with the Internet Assigned Number Authority (IANA [19]). The media type registration process is outlined in RFC 1590 [17]. Use of non-registered media types is discouraged.

### 3.7.1 Canonicalization and Text Defaults

Internet media types are registered with a canonical form. An entity-body transferred via HTTP messages MUST be represented in the appropriate canonical form prior to its transmission except for "text" types, as defined in the next paragraph.

When in canonical form, media subtypes of the "text" type use CRLF as the text line break. HTTP relaxes this requirement and allows the transport of text media with plain CR or LF alone representing a line break when it is done consistently for an entire entity-body. HTTP applications MUST accept CRLF, bare CR, and bare LF as being representative of a line break in text media received via HTTP. In addition, if the text is represented in a character set that does not use octets 13 and 10 for CR and LF respectively, as is the case for some multi-byte character sets, HTTP allows the use of whatever octet sequences are defined by that character set to represent the equivalent of CR and LF for line breaks. This flexibility regarding line breaks applies only to text media in the entity-body; a bare CR or LF MUST NOT be substituted for CRLF within any of the HTTP control structures (such as header fields and multipart boundaries).

If an entity-body is encoded with a content-coding, the underlying data MUST be in a form defined above prior to being encoded.

The "charset" parameter is used with some media types to define the character set (section 3.4) of the data. When no explicit charset parameter is provided by the sender, media subtypes of the "text" type are defined to have a default charset value of "ISO-8859-1" when received via HTTP. Data in character sets other than "ISO-8859-1" or its subsets MUST be labeled with an appropriate charset value. See section 3.4.1 for compatibility problems.

### 3.7.2 Multipart Types

MIME provides for a number of "multipart" types -- encapsulations of one or more entities within a single message-body. All multipart types share a common syntax, as defined in section 5.1.1 of RFC 2046 [40], and MUST include a boundary parameter as part of the media type value. The message body is itself a protocol element and MUST therefore use only CRLF to represent line breaks between body-parts. Unlike in RFC 2046, the epilogue of any multipart message MUST be empty; HTTP applications MUST NOT transmit the epilogue (even if the original multipart contains an epilogue). These restrictions exist in order to preserve the self-delimiting nature of a multipart message-body, wherein the "end" of the message-body is indicated by the ending multipart boundary.

In general, HTTP treats a multipart message-body no differently than any other media type: strictly as payload. The one exception is the "multipart/byteranges" type (appendix 19.2) when it appears in a 206 (Partial Content) response, which will be interpreted by some HTTP caching mechanisms as described in sections 13.5.4 and 14.16. In all other cases, an HTTP user agent SHOULD follow the same or similar behavior as a MIME user agent would upon receipt of a multipart type. The MIME header fields within each body-part of a multipart message-body do not have any significance to HTTP beyond that defined by their MIME semantics.

In general, an HTTP user agent SHOULD follow the same or similar behavior as a MIME user agent would upon receipt of a multipart type. If an application receives an unrecognized multipart subtype, the application MUST treat it as being equivalent to "multipart/mixed".



Note: The "multipart/form-data" type has been specifically defined for carrying form data suitable for processing via the POST request method, as described in RFC 1867 [15].

### 3.8 Product Tokens

Product tokens are used to allow communicating applications to identify themselves by software name and version. Most fields using product tokens also allow sub-products which form a significant part of the application to be listed, separated by white space. By convention, the products are listed in order of their significance for identifying the application.

```
product          = token [ "/" product-version ]
product-version = token
```

Examples:

```
User-Agent: CERN-LineMode/2.15 libwww/2.17b3
Server: Apache/0.8.4
```

Product tokens SHOULD be short and to the point. They MUST NOT be used for advertising or other non-essential information. Although any token character MAY appear in a `product-version`, this token SHOULD only be used for a version identifier (i.e., successive versions of the same product SHOULD only differ in the `product-version` portion of the `product` value).

### 3.9 Quality Values

HTTP content negotiation (section 12) uses short "floating point" numbers to indicate the relative importance ("weight") of various negotiable parameters. A weight is normalized to a real number in the range 0 through 1, where 0 is the minimum and 1 the maximum value. If a parameter has a quality value of 0, then content with this parameter is 'not acceptable' for the client. HTTP/1.1 applications MUST NOT generate more than three digits after the decimal point. User configuration of these values SHOULD also be limited in this fashion.

```
qvalue          = ( "0" [ "." 0*3DIGIT ] )
                | ( "1" [ "." 0*3("0") ] )
```

"Quality values" is a misnomer, since these values merely represent relative degradation in desired quality.

### 3.10 Language Tags

A language tag identifies a natural language spoken, written, or otherwise conveyed by human beings for communication of information to other human beings. Computer languages are explicitly excluded. HTTP uses language tags within the `Accept-Language` and `Content-Language` fields.

The syntax and registry of HTTP language tags is the same as that defined by RFC 1766 [1]. In summary, a language tag is composed of 1 or more parts: A primary language tag and a possibly empty series of subtags:

```
language-tag    = primary-tag *( "-" subtag )
primary-tag     = 1*8ALPHA
subtag          = 1*8ALPHA
```

White space is not allowed within the tag and all tags are case-insensitive. The name space of language tags is administered by the IANA. Example tags include:

```
en, en-US, en-cockney, i-cherokee, x-pig-latin
```

where any two-letter primary-tag is an ISO-639 language abbreviation and any two-letter initial subtag is an ISO-3166 country code. (The last three tags above are not registered tags; all but the last are examples of tags which could be registered in future.)

### 3.11 Entity Tags

Entity tags are used for comparing two or more entities from the same requested resource. HTTP/1.1 uses entity tags in the `ETag` (section 14.19), `If-Match` (section 14.24), `If-None-Match` (section 14.26), and `If-Range`

(section 14.27) header fields. The definition of how they are used and compared as cache validators is in section 13.3.3. An entity tag consists of an opaque quoted string, possibly prefixed by a weakness indicator.

```
entity-tag = [ weak ] opaque-tag
weak       = "W/"
opaque-tag = quoted-string
```

A “strong entity tag” MAY be shared by two entities of a resource only if they are equivalent by octet equality.

A “weak entity tag,” indicated by the “W/” prefix, MAY be shared by two entities of a resource only if the entities are equivalent and could be substituted for each other with no significant change in semantics. A weak entity tag can only be used for weak comparison.

An entity tag MUST be unique across all versions of all entities associated with a particular resource. A given entity tag value MAY be used for entities obtained by requests on different URIs. The use of the same entity tag value in conjunction with entities obtained by requests on different URIs does not imply the equivalence of those entities.

## 3.12 Range Units

HTTP/1.1 allows a client to request that only part (a range of) the response entity be included within the response. HTTP/1.1 uses range units in the `Range` (section 14.35) and `Content-Range` (section 14.16) header fields. An entity can be broken down into subranges according to various structural units.

```
range-unit      = bytes-unit | other-range-unit
bytes-unit      = "bytes"
other-range-unit = token
```

The only range unit defined by HTTP/1.1 is “bytes”. HTTP/1.1 implementations MAY ignore ranges specified using other units. HTTP/1.1 has been designed to allow implementations of applications that do not depend on knowledge of ranges.

# 4 HTTP Message

## 4.1 Message Types

HTTP messages consist of requests from client to server and responses from server to client.

```
HTTP-message = Request | Response ; HTTP/1.1 messages
```

`Request` (section 5) and `Response` (section 6) messages use the generic message format of RFC 822 [9] for transferring entities (the payload of the message). Both types of message consist of a start-line, zero or more header fields (also known as “headers”), an empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields, and possibly a message-body.

```
generic-message = start-line
                  * (message-header CRLF)
                  CRLF
                  [ message-body ]
start-line       = Request-Line | Status-Line
```

In the interest of robustness, servers SHOULD ignore any empty line(s) received where a `Request-Line` is expected. In other words, if the server is reading the protocol stream at the beginning of a message and receives a CRLF first, it should ignore the CRLF.

Certain buggy HTTP/1.0 client implementations generate extra CRLF’s after a POST request. To restate what is explicitly forbidden by the BNF, an HTTP/1.1 client MUST NOT preface or follow a request with an extra CRLF.

## 4.2 Message Headers

HTTP header fields, which include general-header (section 4.5), request-header (section 5.3), response-header (section 6.2), and entity-header (section 7.1) fields, follow the same generic format as that given in Section 3.1 of RFC 822 [9]. Each header field consists of a name followed by a colon (“:”) and the field value. Field names are

case-insensitive. The field value MAY be preceded by any amount of LWS, though a single SP is preferred. Header fields can be extended over multiple lines by preceding each extra line with at least one SP or HT. Applications ought to follow “common form”, where one is known or indicated, when generating HTTP constructs, since there might exist some implementations that fail to accept anything beyond the common forms.

```

message-header = field-name ":" [ field-value ]
field-name     = token
field-value    = *( field-content | LWS )
field-content  = <the OCTETs making up the field-value
                 and consisting of either *TEXT or combinations
                 of token, separators, and quoted-string>

```

The `field-content` does not include any leading or trailing LWS: linear white space occurring before the first non-whitespace character of the `field-value` or after the last non-whitespace character of the `field-value`. Such leading or trailing LWS MAY be removed without changing the semantics of the field value. Any LWS that occurs between `field-content` MAY be replaced with a single SP before interpreting the field value or forwarding the message downstream.

The order in which header fields with differing field names are received is not significant. However, it is “good practice” to send general-header fields first, followed by request-header or response-header fields, and ending with the entity-header fields.

Multiple message-header fields with the same `field-name` MAY be present in a message if and only if the entire `field-value` for that header field is defined as a comma-separated list [i.e., # (values)]. It MUST be possible to combine the multiple header fields into one “`field-name: field-value`” pair, without changing the semantics of the message, by appending each subsequent `field-value` to the first, each separated by a comma. The order in which header fields with the same `field-name` are received is therefore significant to the interpretation of the combined field value, and thus a proxy MUST NOT change the order of these field values when a message is forwarded.

### 4.3 Message Body

The message-body (if any) of an HTTP message is used to carry the entity-body associated with the request or response. The message-body differs from the entity-body only when a transfer-coding has been applied, as indicated by the `Transfer-Encoding` header field (section 14.41).

```

message-body = entity-body
              | <entity-body encoded as per Transfer-Encoding>

```

`Transfer-Encoding` MUST be used to indicate any transfer-codings applied by an application to ensure safe and proper transfer of the message. `Transfer-Encoding` is a property of the message, not of the entity, and thus MAY be added or removed by any application along the request/response chain. (However, section 3.6 places restrictions on when certain transfer-codings may be used.)

The rules for when a message-body is allowed in a message differ for requests and responses.

The presence of a message-body in a request is signaled by the inclusion of a `Content-Length` or `Transfer-Encoding` header field in the request’s message-headers. A message-body MUST NOT be included in a request if the specification of the request method (section 5.1.1) does not allow sending an entity-body in requests. A server SHOULD read and forward a message-body on any request; if the request method does not include defined semantics for an entity-body, then the message-body SHOULD be ignored when handling the request.

For response messages, whether or not a message-body is included with a message is dependent on both the request method and the response status code (section 6.1.1). All responses to the HEAD request method MUST NOT include a message-body, even though the presence of entity-header fields might lead one to believe they do. All 1xx (informational), 204 (no content), and 304 (not modified) responses MUST NOT include a message-body. All other responses do include a message-body, although it MAY be of zero length.

## 4.4 Message Length

The transfer-length of a message is the length of the message-body as it appears in the message; that is, after any transfer-codings have been applied. When a message-body is included with a message, the transfer-length of that body is determined by one of the following (in order of precedence):

1. Any response message which “MUST NOT” include a message-body (such as the 1xx, 204, and 304 responses and any response to a HEAD request) is always terminated by the first empty line after the header fields, regardless of the entity-header fields present in the message.
2. If a `Transfer-Encoding` header field (section 14.41) is present and has any value other than “identity”, then the transfer-length is defined by use of the “chunked” transfer-coding (section 3.6), unless the message is terminated by closing the connection.
3. If a `Content-Length` header field (section 14.13) is present, its decimal value in OCTETs represents both the entity-length and the transfer-length. The `Content-Length` header field MUST NOT be sent if these two lengths are different (i.e., if a `Transfer-Encoding` header field is present). If a message is received with both a `Transfer-Encoding` header field and a `Content-Length` header field, the latter MUST be ignored.
4. If the message uses the media type “multipart/byteranges”, and the transfer-length is not otherwise specified, then this self-delimiting media type defines the transfer-length. This media type MUST NOT be used unless the sender knows that the recipient can parse it; the presence in a request of a `Range` header with multiple byte-range specifiers from a 1.1 client implies that the client can parse multipart/byteranges responses.

A range header might be forwarded by a 1.0 proxy that does not understand multipart/byteranges; in this case the server MUST delimit the message using methods defined in items 1,3 or 5 of this section.

5. By the server closing the connection. (Closing the connection cannot be used to indicate the end of a request body, since that would leave no possibility for the server to send back a response.)

For compatibility with HTTP/1.0 applications, HTTP/1.1 requests containing a message-body MUST include a valid `Content-Length` header field unless the server is known to be HTTP/1.1 compliant. If a request contains a message-body and a `Content-Length` is not given, the server SHOULD respond with 400 (bad request) if it cannot determine the length of the message, or with 411 (length required) if it wishes to insist on receiving a valid `Content-Length`.

All HTTP/1.1 applications that receive entities MUST accept the “chunked” transfer-coding (section 3.6), thus allowing this mechanism to be used for messages when the message length cannot be determined in advance.

Messages MUST NOT include both a `Content-Length` header field and a non-identity transfer-coding. If the message does include a non-identity transfer-coding, the `Content-Length` MUST be ignored.

When a `Content-Length` is given in a message where a message-body is allowed, its field value MUST exactly match the number of OCTETs in the message-body. HTTP/1.1 user agents MUST notify the user when an invalid length is received and detected.

## 4.5 General Header Fields

There are a few header fields which have general applicability for both request and response messages, but which do not apply to the entity being transferred. These header fields apply only to the message being transmitted.

```

general-header = Cache-Control           ; Section 14.9
                | Connection            ; Section 14.10
                | Date                  ; Section 14.18
                | Pragma                ; Section 14.32
                | Trailer               ; Section 14.40
                | Transfer-Encoding     ; Section 14.41

```

```

| Upgrade                ; Section 14.42
| Via                    ; Section 14.45
| Warning                ; Section 14.46

```

General-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields may be given the semantics of general header fields if all parties in the communication recognize them to be general-header fields. Unrecognized header fields are treated as entity-header fields.

## 5 Request

A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.

```

Request = Request-Line           ; Section 5.1
        *(( general-header       ; Section 4.5
          | request-header       ; Section 5.3
          | entity-header ) CRLF) ; Section 7.1
        CRLF
        [ message-body ]        ; Section 4.3

```

### 5.1 Request-Line

The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

#### 5.1.1 Method

The Method token indicates the method to be performed on the resource identified by the Request-URI. The method is case-sensitive.

```

Method = "OPTIONS"           ; Section 9.2
       | "GET"               ; Section 9.3
       | "HEAD"              ; Section 9.4
       | "POST"              ; Section 9.5
       | "PUT"                ; Section 9.6
       | "DELETE"            ; Section 9.7
       | "TRACE"             ; Section 9.8
       | "CONNECT"           ; Section 9.9
       | extension-method
extension-method = token

```

The list of methods allowed by a resource can be specified in an Allow header field (section 14.7). The return code of the response always notifies the client whether a method is currently allowed on a resource, since the set of allowed methods can change dynamically. An origin server SHOULD return the status code 405 (Method Not Allowed) if the method is known by the origin server but not allowed for the requested resource, and 501 (Not Implemented) if the method is unrecognized or not implemented by the origin server. The methods GET and HEAD MUST be supported by all general-purpose servers. All other methods are OPTIONAL; however, if the above methods are implemented, they MUST be implemented with the same semantics as those specified in section 9.

#### 5.1.2 Request-URI

The Request-URI is a Uniform Resource Identifier (section 3.2) and identifies the resource upon which to apply the request.

```
Request-URI = "*" | absoluteURI | abs_path | authority
```

The four options for Request-URI are dependent on the nature of the request. The asterisk "\*" means that the request does not apply to a particular resource, but to the server itself, and is only allowed when the method used does not necessarily apply to a resource. One example would be

OPTIONS \* HTTP/1.1

The `absoluteURI` form is REQUIRED when the request is being made to a proxy. The proxy is requested to forward the request or service it from a valid cache, and return the response. Note that the proxy MAY forward the request on to another proxy or directly to the server specified by the `absoluteURI`. In order to avoid request loops, a proxy MUST be able to recognize all of its server names, including any aliases, local variations, and the numeric IP address. An example `Request-Line` would be:

```
GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1
```

To allow for transition to `absoluteURIs` in all requests in future versions of HTTP, all HTTP/1.1 servers MUST accept the `absoluteURI` form in requests, even though HTTP/1.1 clients will only generate them in requests to proxies.

The `authority` form is only used by the CONNECT method (section 9.9).

The most common form of `Request-URI` is that used to identify a resource on an origin server or gateway. In this case the absolute path of the URI MUST be transmitted (see section 3.2.1, `abs_path`) as the `Request-URI`, and the network location of the URI (`authority`) MUST be transmitted in a `Host` header field. For example, a client wishing to retrieve the resource above directly from the origin server would create a TCP connection to port 80 of the host "www.w3.org" and send the lines:

```
GET /pub/WWW/TheProject.html HTTP/1.1
Host: www.w3.org
```

followed by the remainder of the `Request`. Note that the absolute path cannot be empty; if none is present in the original URI, it MUST be given as "/" (the server root).

The `Request-URI` is transmitted in the format specified in section 3.2.1. If the `Request-URI` is encoded using the "% HEX HEX" encoding [42], the origin server MUST decode the `Request-URI` in order to properly interpret the request. Servers SHOULD respond to invalid `Request-URIs` with an appropriate status code.

A transparent proxy MUST NOT rewrite the "`abs_path`" part of the received `Request-URI` when forwarding it to the next inbound server, except as noted above to replace a null `abs_path` with "/".

Note: The "no rewrite" rule prevents the proxy from changing the meaning of the request when the origin server is improperly using a non-reserved URI character for a reserved purpose. Implementors should be aware that some pre-HTTP/1.1 proxies have been known to rewrite the `Request-URI`.

## 5.2 The Resource Identified by a Request

The exact resource identified by an Internet request is determined by examining both the `Request-URI` and the `Host` header field.

An origin server that does not allow resources to differ by the requested host MAY ignore the `Host` header field value when determining the resource identified by an HTTP/1.1 request. (But see section 19.6.1.1 for other requirements on `Host` support in HTTP/1.1.)

An origin server that does differentiate resources based on the host requested (sometimes referred to as virtual hosts or vanity host names) MUST use the following rules for determining the requested resource on an HTTP/1.1 request:

1. If `Request-URI` is an `absoluteURI`, the host is part of the `Request-URI`. Any `Host` header field value in the request MUST be ignored.
2. If the `Request-URI` is not an `absoluteURI`, and the request includes a `Host` header field, the host is determined by the `Host` header field value.
3. If the host as determined by rule 1 or 2 is not a valid host on the server, the response MUST be a 400 (Bad Request) error message.

Recipients of an HTTP/1.0 request that lacks a `Host` header field MAY attempt to use heuristics (e.g., examination of the URI path for something unique to a particular host) in order to determine what exact resource is being requested.

### 5.3 Request Header Fields

The request-header fields allow the client to pass additional information about the request, and about the client itself, to the server. These fields act as request modifiers, with semantics equivalent to the parameters on a programming language method invocation.

```

request-header = Accept                ; Section 14.1
                | Accept-Charset      ; Section 14.2
                | Accept-Encoding     ; Section 14.3
                | Accept-Language     ; Section 14.4
                | Authorization       ; Section 14.8
                | Expect              ; Section 14.20
                | From                ; Section 14.22
                | Host                ; Section 14.23
                | If-Match            ; Section 14.24
                | If-Modified-Since  ; Section 14.25
                | If-None-Match      ; Section 14.26
                | If-Range            ; Section 14.27
                | If-Unmodified-Since ; Section 14.28
                | Max-Forwards        ; Section 14.31
                | Proxy-Authorization ; Section 14.34
                | Range               ; Section 14.35
                | Referer             ; Section 14.36
                | TE                  ; Section 14.39
                | User-Agent          ; Section 14.43

```

Request-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields MAY be given the semantics of request-header fields if all parties in the communication recognize them to be request-header fields. Unrecognized header fields are treated as entity-header fields.

## 6 Response

After receiving and interpreting a request message, a server responds with an HTTP response message.

```

Response      = Status-Line          ; Section 6.1
                * ( ( general-header  ; Section 4.5
                    | response-header ; Section 6.2
                    | entity-header ) ; Section 7.1
                  CRLF
                [ message-body ]     ; Section 7.2

```

### 6.1 Status-Line

The first line of a Response message is the `Status-Line`, consisting of the protocol version followed by a numeric status code and its associated textual phrase, with each element separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF
```

#### 6.1.1 Status Code and Reason Phrase

The `Status-Code` element is a 3-digit integer result code of the attempt to understand and satisfy the request. These codes are fully defined in section 10. The `Reason-Phrase` is intended to give a short textual description of the `Status-Code`. The `Status-Code` is intended for use by automata and the `Reason-Phrase` is intended for the human user. The client is not required to examine or display the `Reason-Phrase`.

The first digit of the `Status-Code` defines the class of response. The last two digits do not have any categorization role. There are 5 values for the first digit:

- 1xx: Informational - Request received, continuing process
- 2xx: Success - The action was successfully received, understood, and accepted
- 3xx: Redirection - Further action must be taken in order to complete the request
- 4xx: Client Error - The request contains bad syntax or cannot be fulfilled
- 5xx: Server Error - The server failed to fulfill an apparently valid request

The individual values of the numeric status codes defined for HTTP/1.1, and an example set of corresponding `Reason-Phrase`'s, are presented below. The reason phrases listed here are only recommendations -- they MAY be replaced by local equivalents without affecting the protocol.

```

Status-Code      =
    "100" ; Section 10.1.1: Continue
    | "101" ; Section 10.1.2: Switching Protocols
    | "200" ; Section 10.2.1: OK
    | "201" ; Section 10.2.2: Created
    | "202" ; Section 10.2.3: Accepted
    | "203" ; Section 10.2.4: Non-Authoritative Information
    | "204" ; Section 10.2.5: No Content
    | "205" ; Section 10.2.6: Reset Content
    | "206" ; Section 10.2.7: Partial Content
    | "300" ; Section 10.3.1: Multiple Choices
    | "301" ; Section 10.3.2: Moved Permanently
    | "302" ; Section 10.3.3: Found
    | "303" ; Section 10.3.4: See Other
    | "304" ; Section 10.3.5: Not Modified
    | "305" ; Section 10.3.6: Use Proxy
    | "307" ; Section 10.3.8: Temporary Redirect
    | "400" ; Section 10.4.1: Bad Request
    | "401" ; Section 10.4.2: Unauthorized
    | "402" ; Section 10.4.3: Payment Required
    | "403" ; Section 10.4.4: Forbidden
    | "404" ; Section 10.4.5: Not Found
    | "405" ; Section 10.4.6: Method Not Allowed
    | "406" ; Section 10.4.7: Not Acceptable
    | "407" ; Section 10.4.8: Proxy Authentication Required
    | "408" ; Section 10.4.9: Request Time-out
    | "409" ; Section 10.4.10: Conflict
    | "410" ; Section 10.4.11: Gone
    | "411" ; Section 10.4.12: Length Required
    | "412" ; Section 10.4.13: Precondition Failed
    | "413" ; Section 10.4.14: Request Entity Too Large
    | "414" ; Section 10.4.15: Request-URI Too Large
    | "415" ; Section 10.4.16: Unsupported Media Type
    | "416" ; Section 10.4.17: Requested range not satisfiable
    | "417" ; Section 10.4.18: Expectation Failed
    | "500" ; Section 10.5.1: Internal Server Error
    | "501" ; Section 10.5.2: Not Implemented
    | "502" ; Section 10.5.3: Bad Gateway
    | "503" ; Section 10.5.4: Service Unavailable
    | "504" ; Section 10.5.5: Gateway Time-out
    | "505" ; Section 10.5.6: HTTP Version not supported
    | extension-code
extension-code = 3DIGIT
Reason-Phrase = *<TEXT, excluding CR, LF>

```



HTTP status codes are extensible. HTTP applications are not required to understand the meaning of all registered status codes, though such understanding is obviously desirable. However, applications **MUST** understand the class of any status code, as indicated by the first digit, and treat any unrecognized response as being equivalent to the x00 status code of that class, with the exception that an unrecognized response **MUST NOT** be cached. For example, if an unrecognized status code of 431 is received by the client, it can safely assume that there was something wrong with its request and treat the response as if it had received a 400 status code. In such cases, user agents **SHOULD** present to the user the entity returned with the response, since that entity is likely to include human-readable information which will explain the unusual status.

## 6.2 Response Header Fields

The response-header fields allow the server to pass additional information about the response which cannot be placed in the *Status-Line*. These header fields give information about the server and about further access to the resource identified by the *Request-URL*.

```

response-header = Accept-Ranges           ; Section 14.5
                  | Age                   ; Section 14.6
                  | ETag                   ; Section 14.19
                  | Location               ; Section 14.30
                  | Proxy-Authenticate    ; Section 14.33
                  | Retry-After           ; Section 14.37
                  | Server                 ; Section 14.38
                  | Vary                   ; Section 14.44
                  | WWW-Authenticate      ; Section 14.47

```

Response-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields **MAY** be given the semantics of response-header fields if all parties in the communication recognize them to be response-header fields. Unrecognized header fields are treated as entity-header fields.

## 7 Entity

*Request* and *Response* messages **MAY** transfer an entity if not otherwise restricted by the request method or response status code. An entity consists of entity-header fields and an entity-body, although some responses will only include the entity-headers.

In this section, both *sender* and *recipient* refer to either the client or the server, depending on who sends and who receives the entity.

### 7.1 Entity Header Fields

Entity-header fields define meta-information about the entity-body or, if no body is present, about the resource identified by the request. Some of this meta-information is **OPTIONAL**; some might be **REQUIRED** by portions of this specification.

```

entity-header    = Allow                   ; Section 14.7
                  | Content-Encoding       ; Section 14.11
                  | Content-Language       ; Section 14.12
                  | Content-Length         ; Section 14.13
                  | Content-Location       ; Section 14.14
                  | Content-MD5            ; Section 14.15
                  | Content-Range          ; Section 14.16
                  | Content-Type           ; Section 14.17
                  | Expires                 ; Section 14.21
                  | Last-Modified          ; Section 14.29
                  | extension-header
extension-header = message-header

```

The extension-header mechanism allows additional entity-header fields to be defined without changing the protocol, but these fields cannot be assumed to be recognizable by the recipient. Unrecognized header fields SHOULD be ignored by the recipient and MUST be forwarded by transparent proxies.

## 7.2 Entity Body

The entity-body (if any) sent with an HTTP request or response is in a format and encoding defined by the entity-header fields.

```
entity-body = *OCTET
```

An entity-body is only present in a message when a message-body is present, as described in section 4.3. The entity-body is obtained from the message-body by decoding any `Transfer-Encoding` that might have been applied to ensure safe and proper transfer of the message.

### 7.2.1 Type

When an entity-body is included with a message, the data type of that body is determined via the header fields `Content-Type` and `Content-Encoding`. These define a two-layer, ordered encoding model:

```
entity-body := Content-Encoding( Content-Type( data ) )
```

`Content-Type` specifies the media type of the underlying data. `Content-Encoding` may be used to indicate any additional content codings applied to the data, usually for the purpose of data compression, that are a property of the requested resource. There is no default encoding.

Any HTTP/1.1 message containing an entity-body SHOULD include a `Content-Type` header field defining the media type of that body. If and **only if** the media type is not given by a `Content-Type` field, the recipient MAY attempt to guess the media type via inspection of its content and/or the name extension(s) of the URI used to identify the resource. If the media type remains unknown, the recipient SHOULD treat it as type "application/octet-stream".

### 7.2.2 Entity Length

The entity-length of a message is the length of the message-body before any transfer-codings have been applied. Section 4.4 defines how the transfer-length of a message-body is determined.

# 8 Connections

## 8.1 Persistent Connections

### 8.1.1 Purpose

Prior to persistent connections, a separate TCP connection was established to fetch each URL, increasing the load on HTTP servers and causing congestion on the Internet. The use of inline images and other associated data often require a client to make multiple requests of the same server in a short amount of time. Analysis of these performance problems and results from a prototype implementation are available [26] [30]. Implementation experience and measurements of actual HTTP/1.1 (RFC 2068) implementations show good results [39]. Alternatives have also been explored, for example, T/TCP [27].

Persistent HTTP connections have a number of advantages:

- By opening and closing fewer TCP connections, CPU time is saved in routers and hosts (clients, servers, proxies, gateways, tunnels, or caches), and memory used for TCP protocol control blocks can be saved in hosts.
- HTTP requests and responses can be pipelined on a connection. Pipelining allows a client to make multiple requests without waiting for each response, allowing a single TCP connection to be used much more efficiently, with much lower elapsed time.

- Network congestion is reduced by reducing the number of packets caused by TCP opens, and by allowing TCP sufficient time to determine the congestion state of the network.
- Latency on subsequent requests is reduced since there is no time spent in TCP's connection opening handshake.
- HTTP can evolve more gracefully, since errors can be reported without the penalty of closing the TCP connection. Clients using future versions of HTTP might optimistically try a new feature, but if communicating with an older server, retry with old semantics after an error is reported.

HTTP implementations SHOULD implement persistent connections.

### 8.1.2 Overall Operation

A significant difference between HTTP/1.1 and earlier versions of HTTP is that persistent connections are the default behavior of any HTTP connection. That is, unless otherwise indicated, the client SHOULD assume that the server will maintain a persistent connection, even after error responses from the server.

Persistent connections provide a mechanism by which a client and a server can signal the close of a TCP connection. This signaling takes place using the `Connection` header field (section 14.10). Once a close has been signaled, the client MUST NOT send any more requests on that connection.

#### 8.1.2.1 Negotiation

An HTTP/1.1 server MAY assume that a HTTP/1.1 client intends to maintain a persistent connection unless a `Connection` header including the connection-token "close" was sent in the request. If the server chooses to close the connection immediately after sending the response, it SHOULD send a `Connection` header including the connection-token `close`.

An HTTP/1.1 client MAY expect a connection to remain open, but would decide to keep it open based on whether the response from a server contains a `Connection` header with the connection-token `close`. In case the client does not want to maintain a connection for more than that request, it SHOULD send a `Connection` header including the connection-token `close`.

If either the client or the server sends the `close` token in the `Connection` header, that request becomes the last one for the connection.

Clients and servers SHOULD NOT assume that a persistent connection is maintained for HTTP versions less than 1.1 unless it is explicitly signaled. See section 19.6.2 for more information on backward compatibility with HTTP/1.0 clients.

In order to remain persistent, all messages on the connection MUST have a self-defined message length (i.e., one not defined by closure of the connection), as described in section 4.4.

#### 8.1.2.2 Pipelining

A client that supports persistent connections MAY "pipeline" its requests (i.e., send multiple requests without waiting for each response). A server MUST send its responses to those requests in the same order that the requests were received.

Clients which assume persistent connections and pipeline immediately after connection establishment SHOULD be prepared to retry their connection if the first pipelined attempt fails. If a client does such a retry, it MUST NOT pipeline before it knows the connection is persistent. Clients MUST also be prepared to resend their requests if the server closes the connection before sending all of the corresponding responses.

Clients SHOULD NOT pipeline requests using non-idempotent methods or non-idempotent sequences of methods (see section 9.1.2). Otherwise, a premature termination of the transport connection could lead to indeterminate results. A client wishing to send a non-idempotent request SHOULD wait to send that request until it has received the response status for the previous request.

### 8.1.3 Proxy Servers

It is especially important that proxies correctly implement the properties of the `Connection` header field as specified in section 14.10.

The proxy server **MUST** signal persistent connections separately with its clients and the origin servers (or other proxy servers) that it connects to. Each persistent connection applies to only one transport link.

A proxy server **MUST NOT** establish a HTTP/1.1 persistent connection with an HTTP/1.0 client (but see RFC 2068 [33] for information and discussion of the problems with the `Keep-Alive` header implemented by many HTTP/1.0 clients).

### 8.1.4 Practical Considerations

Servers will usually have some time-out value beyond which they will no longer maintain an inactive connection. Proxy servers might make this a higher value since it is likely that the client will be making more connections through the same server. The use of persistent connections places no requirements on the length (or existence) of this time-out for either the client or the server.

When a client or server wishes to time-out it **SHOULD** issue a graceful close on the transport connection. Clients and servers **SHOULD** both constantly watch for the other side of the transport close, and respond to it as appropriate. If a client or server does not detect the other side's close promptly it could cause unnecessary resource drain on the network.

A client, server, or proxy **MAY** close the transport connection at any time. For example, a client might have started to send a new request at the same time that the server has decided to close the "idle" connection. From the server's point of view, the connection is being closed while it was idle, but from the client's point of view, a request is in progress.

This means that clients, servers, and proxies **MUST** be able to recover from asynchronous close events. Client software **SHOULD** reopen the transport connection and retransmit the aborted sequence of requests without user interaction so long as the request sequence is idempotent (see section 9.1.2). Non-idempotent methods or sequences **MUST NOT** be automatically retried, although user agents **MAY** offer a human operator the choice of retrying the request(s). Confirmation by user-agent software with semantic understanding of the application **MAY** substitute for user confirmation. The automatic retry **SHOULD NOT** be repeated if the second sequence of requests fails.

Servers **SHOULD** always respond to at least one request per connection, if at all possible. Servers **SHOULD NOT** close a connection in the middle of transmitting a response, unless a network or client failure is suspected.

Clients that use persistent connections **SHOULD** limit the number of simultaneous connections that they maintain to a given server. A single-user client **SHOULD NOT** maintain more than 2 connections with any server or proxy. A proxy **SHOULD** use up to  $2 * N$  connections to another server or proxy, where  $N$  is the number of simultaneously active users. These guidelines are intended to improve HTTP response times and avoid congestion.

## 8.2 Message Transmission Requirements

### 8.2.1 Persistent Connections and Flow Control

HTTP/1.1 servers **SHOULD** maintain persistent connections and use TCP's flow control mechanisms to resolve temporary overloads, rather than terminating connections with the expectation that clients will retry. The latter technique can exacerbate network congestion.

### 8.2.2 Monitoring Connections for Error Status Messages

An HTTP/1.1 (or later) client sending a message-body **SHOULD** monitor the network connection for an error status while it is transmitting the request. If the client sees an error status, it **SHOULD** immediately cease transmitting the body. If the body is being sent using a "chunked" encoding (section 3.6), a zero length chunk and empty trailer **MAY**

be used to prematurely mark the end of the message. If the body was preceded by a `Content-Length` header, the client **MUST** close the connection.

### 8.2.3 Use of the 100 (Continue) Status

The purpose of the 100 (Continue) status (see section 10.1.1) is to allow a client that is sending a request message with a request body to determine if the origin server is willing to accept the request (based on the request headers) before the client sends the request body. In some cases, it might either be inappropriate or highly inefficient for the client to send the body if the server will reject the message without looking at the body.

Requirements for HTTP/1.1 clients:

- If a client will wait for a 100 (Continue) response before sending the request body, it **MUST** send an `Expect` request-header field (section 14.20) with the “100-continue” expectation.
- A client **MUST NOT** send an `Expect` request-header field (section 14.20) with the “100-continue” expectation if it does not intend to send a request body.

Because of the presence of older implementations, the protocol allows ambiguous situations in which a client may send “Expect: 100-continue” without receiving either a 417 (Expectation Failed) status or a 100 (Continue) status. Therefore, when a client sends this header field to an origin server (possibly via a proxy) from which it has never seen a 100 (Continue) status, the client **SHOULD NOT** wait for an indefinite period before sending the request body.

Requirements for HTTP/1.1 origin servers:

- Upon receiving a request which includes an `Expect` request-header field with the “100-continue” expectation, an origin server **MUST** either respond with 100 (Continue) status and continue to read from the input stream, or respond with a final status code. The origin server **MUST NOT** wait for the request body before sending the 100 (Continue) response. If it responds with a final status code, it **MAY** close the transport connection or it **MAY** continue to read and discard the rest of the request. It **MUST NOT** perform the requested method if it returns a final status code.
- An origin server **SHOULD NOT** send a 100 (Continue) response if the request message does not include an `Expect` request-header field with the “100-continue” expectation, and **MUST NOT** send a 100 (Continue) response if such a request comes from an HTTP/1.0 (or earlier) client. There is an exception to this rule: for compatibility with RFC 2068, a server **MAY** send a 100 (Continue) status in response to an HTTP/1.1 PUT or POST request that does not include an `Expect` request-header field with the “100-continue” expectation. This exception, the purpose of which is to minimize any client processing delays associated with an undeclared wait for 100 (Continue) status, applies only to HTTP/1.1 requests, and not to requests with any other HTTP-version value.
- An origin server **MAY** omit a 100 (Continue) response if it has already received some or all of the request body for the corresponding request.
- An origin server that sends a 100 (Continue) response **MUST** ultimately send a final status code, once the request body is received and processed, unless it terminates the transport connection prematurely.
- If an origin server receives a request that does not include an `Expect` request-header field with the “100-continue” expectation, the request includes a request body, and the server responds with a final status code before reading the entire request body from the transport connection, then the server **SHOULD NOT** close the transport connection until it has read the entire request, or until the client closes the connection. Otherwise, the client might not reliably receive the response message. However, this requirement is not construed as preventing a server from defending itself against denial-of-service attacks, or from badly broken client implementations.

Requirements for HTTP/1.1 proxies:

- If a proxy receives a request that includes an `Expect` request-header field with the “100-continue” expectation, and the proxy either knows that the next-hop server complies with HTTP/1.1 or higher, or does

- not know the HTTP version of the next-hop server, it MUST forward the request, including the `Expect` header field.
- If the proxy knows that the version of the next-hop server is HTTP/1.0 or lower, it MUST NOT forward the request, and it MUST respond with a 417 (Expectation Failed) status.
  - Proxies SHOULD maintain a cache recording the HTTP version numbers received from recently-referenced next-hop servers.
  - A proxy MUST NOT forward a 100 (Continue) response if the request message was received from an HTTP/1.0 (or earlier) client and did not include an `Expect` request-header field with the “100-continue” expectation. This requirement overrides the general rule for forwarding of 1xx responses (see section 10.1).

#### 8.2.4 Client Behavior if Server Prematurely Closes Connection

If an HTTP/1.1 client sends a request which includes a request body, but which does not include an `Expect` request-header field with the “100-continue” expectation, and if the client is not directly connected to an HTTP/1.1 origin server, and if the client sees the connection close before receiving any status from the server, the client SHOULD retry the request. If the client does retry this request, it MAY use the following “binary exponential backoff” algorithm to be assured of obtaining a reliable response:

1. Initiate a new connection to the server
2. Transmit the request-headers
3. Initialize a variable R to the estimated round-trip time to the server (e.g., based on the time it took to establish the connection), or to a constant value of 5 seconds if the round-trip time is not available.
4. Compute  $T = R * (2^{**}N)$ , where N is the number of previous retries of this request.
5. Wait either for an error response from the server, or for T seconds (whichever comes first)
6. If no error response is received, after T seconds transmit the body of the request.
7. If client sees that the connection is closed prematurely, repeat from step 1 until the request is accepted, an error response is received, or the user becomes impatient and terminates the retry process.

If at any point an error status is received, the client

- SHOULD NOT continue and
- SHOULD close the connection if it has not completed sending the request message.

## 9 Method Definitions

The set of common methods for HTTP/1.1 is defined below. Although this set can be expanded, additional methods cannot be assumed to share the same semantics for separately extended clients and servers.

The `Host` request-header field (section 14.23) MUST accompany all HTTP/1.1 requests.

### 9.1 Safe and Idempotent Methods

#### 9.1.1 Safe Methods

Implementors should be aware that the software represents the user in their interactions over the Internet, and should be careful to allow the user to be aware of any actions they might take which may have an unexpected significance to themselves or others.

In particular, the convention has been established that the `GET` and `HEAD` methods **SHOULD NOT** have the significance of taking an action other than retrieval. These methods ought to be considered “safe”. This allows user agents to represent other methods, such as `POST`, `PUT` and `DELETE`, in a special way, so that the user is made aware of the fact that a possibly unsafe action is being requested.

Naturally, it is not possible to ensure that the server does not generate side-effects as a result of performing a `GET` request; in fact, some dynamic resources consider that a feature. The important distinction here is that the user did not request the side-effects, so therefore cannot be held accountable for them.

### 9.1.2 Idempotent Methods

Methods can also have the property of “idempotence” in that (aside from error or expiration issues) the side-effects of  $N > 0$  identical requests is the same as for a single request. The methods `GET`, `HEAD`, `PUT` and `DELETE` share this property. Also, the methods `OPTIONS` and `TRACE` **SHOULD NOT** have side effects, and so are inherently idempotent.

However, it is possible that a sequence of several requests is non-idempotent, even if all of the methods executed in that sequence are idempotent. (A sequence is idempotent if a single execution of the entire sequence always yields a result that is not changed by a reexecution of all, or part, of that sequence.) For example, a sequence is non-idempotent if its result depends on a value that is later modified in the same sequence.

A sequence that never has side effects is idempotent, by definition (provided that no concurrent operations are being executed on the same set of resources).

## 9.2 OPTIONS

The `OPTIONS` method represents a request for information about the communication options available on the request/response chain identified by the `Request-URI`. This method allows the client to determine the options and/or requirements associated with a resource, or the capabilities of a server, without implying a resource action or initiating a resource retrieval.

Responses to this method are not cacheable.

If the `OPTIONS` request includes an entity-body (as indicated by the presence of `Content-Length` or `Transfer-Encoding`), then the media type **MUST** be indicated by a `Content-Type` field. Although this specification does not define any use for such a body, future extensions to HTTP might use the `OPTIONS` body to make more detailed queries on the server. A server that does not support such an extension **MAY** discard the request body.

If the `Request-URI` is an asterisk (“\*”), the `OPTIONS` request is intended to apply to the server in general rather than to a specific resource. Since a server’s communication options typically depend on the resource, the “\*” request is only useful as a “ping” or “no-op” type of method; it does nothing beyond allowing the client to test the capabilities of the server. For example, this can be used to test a proxy for HTTP/1.1 compliance (or lack thereof).

If the `Request-URI` is not an asterisk, the `OPTIONS` request applies only to the options that are available when communicating with that resource.

A 200 response **SHOULD** include any header fields that indicate optional features implemented by the server and applicable to that resource (e.g., `Allow`), possibly including extensions not defined by this specification. The response body, if any, **SHOULD** also include information about the communication options. The format for such a body is not defined by this specification, but might be defined by future extensions to HTTP. Content negotiation **MAY** be used to select the appropriate response format. If no response body is included, the response **MUST** include a `Content-Length` field with a field-value of “0”.

The `Max-Forwards` request-header field **MAY** be used to target a specific proxy in the request chain. When a proxy receives an `OPTIONS` request on an absoluteURI for which request forwarding is permitted, the proxy **MUST** check for a `Max-Forwards` field. If the `Max-Forwards` field-value is zero (“0”), the proxy **MUST NOT** forward the message; instead, the proxy **SHOULD** respond with its own communication options. If the `Max-Forwards` field-value is an integer greater than zero, the proxy **MUST** decrement the field-value when it forwards

the request. If no `Max-Forwards` field is present in the request, then the forwarded request **MUST NOT** include a `Max-Forwards` field.

### 9.3 GET

The `GET` method means retrieve whatever information (in the form of an entity) is identified by the `Request-URI`. If the `Request-URI` refers to a data-producing process, it is the produced data which shall be returned as the entity in the response and not the source text of the process, unless that text happens to be the output of the process.

The semantics of the `GET` method change to a “conditional `GET`” if the request message includes an `If-Modified-Since`, `If-Unmodified-Since`, `If-Match`, `If-None-Match`, or `If-Range` header field. A conditional `GET` method requests that the entity be transferred only under the circumstances described by the conditional header field(s). The conditional `GET` method is intended to reduce unnecessary network usage by allowing cached entities to be refreshed without requiring multiple requests or transferring data already held by the client.

The semantics of the `GET` method change to a “partial `GET`” if the request message includes a `Range` header field. A partial `GET` requests that only part of the entity be transferred, as described in section 14.35. The partial `GET` method is intended to reduce unnecessary network usage by allowing partially-retrieved entities to be completed without transferring data already held by the client.

The response to a `GET` request is cacheable if and only if it meets the requirements for HTTP caching described in section 13.

See section 15.1.3 for security considerations when used for forms.

### 9.4 HEAD

The `HEAD` method is identical to `GET` except that the server **MUST NOT** return a message-body in the response. The metainformation contained in the HTTP headers in response to a `HEAD` request **SHOULD** be identical to the information sent in response to a `GET` request. This method can be used for obtaining metainformation about the entity implied by the request without transferring the entity-body itself. This method is often used for testing hypertext links for validity, accessibility, and recent modification.

The response to a `HEAD` request **MAY** be cacheable in the sense that the information contained in the response **MAY** be used to update a previously cached entity from that resource. If the new field values indicate that the cached entity differs from the current entity (as would be indicated by a change in `Content-Length`, `Content-MD5`, `ETag` or `Last-Modified`), then the cache **MUST** treat the cache entry as stale.

### 9.5 POST

The `POST` method is used to request that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the `Request-URI` in the `Request-Line`. `POST` is designed to allow a uniform method to cover the following functions:

- Annotation of existing resources;
- Posting a message to a bulletin board, newsgroup, mailing list, or similar group of articles;
- Providing a block of data, such as the result of submitting a form, to a data-handling process;
- Extending a database through an append operation.

The actual function performed by the `POST` method is determined by the server and is usually dependent on the `Request-URI`. The posted entity is subordinate to that URI in the same way that a file is subordinate to a directory containing it, a news article is subordinate to a newsgroup to which it is posted, or a record is subordinate to a database.



The action performed by the `POST` method might not result in a resource that can be identified by a URI. In this case, either 200 (OK) or 204 (No Content) is the appropriate response status, depending on whether or not the response includes an entity that describes the result.

If a resource has been created on the origin server, the response SHOULD be 201 (Created) and contain an entity which describes the status of the request and refers to the new resource, and a `Location` header (see section 14.30).

Responses to this method are not cacheable, unless the response includes appropriate `Cache-Control` or `Expires` header fields. However, the 303 (See Other) response can be used to direct the user agent to retrieve a cacheable resource.

`POST` requests MUST obey the message transmission requirements set out in section 8.2.

See section 15.1.3 for security considerations.

## 9.6 PUT

The `PUT` method requests that the enclosed entity be stored under the supplied `Request-URI`. If the `Request-URI` refers to an already existing resource, the enclosed entity SHOULD be considered as a modified version of the one residing on the origin server. If the `Request-URI` does not point to an existing resource, and that URI is capable of being defined as a new resource by the requesting user agent, the origin server can create the resource with that URI. If a new resource is created, the origin server MUST inform the user agent via the 201 (Created) response. If an existing resource is modified, either the 200 (OK) or 204 (No Content) response codes SHOULD be sent to indicate successful completion of the request. If the resource could not be created or modified with the `Request-URI`, an appropriate error response SHOULD be given that reflects the nature of the problem. The recipient of the entity MUST NOT ignore any `Content-*` (e.g. `Content-Range`) headers that it does not understand or implement and MUST return a 501 (Not Implemented) response in such cases.

If the request passes through a cache and the `Request-URI` identifies one or more currently cached entities, those entries SHOULD be treated as stale. Responses to this method are not cacheable.

The fundamental difference between the `POST` and `PUT` requests is reflected in the different meaning of the `Request-URI`. The URI in a `POST` request identifies the resource that will handle the enclosed entity. That resource might be a data-accepting process, a gateway to some other protocol, or a separate entity that accepts annotations. In contrast, the URI in a `PUT` request identifies the entity enclosed with the request -- the user agent knows what URI is intended and the server MUST NOT attempt to apply the request to some other resource. If the server desires that the request be applied to a different URI, it MUST send a 301 (Moved Permanently) response; the user agent MAY then make its own decision regarding whether or not to redirect the request.

A single resource MAY be identified by many different URIs. For example, an article might have a URI for identifying "the current version" which is separate from the URI identifying each particular version. In this case, a `PUT` request on a general URI might result in several other URIs being defined by the origin server.

HTTP/1.1 does not define how a `PUT` method affects the state of an origin server.

`PUT` requests MUST obey the message transmission requirements set out in section 8.2.

Unless otherwise specified for a particular entity-header, the entity-headers in the `PUT` request SHOULD be applied to the resource created or modified by the `PUT`.

## 9.7 DELETE

The `DELETE` method requests that the origin server delete the resource identified by the `Request-URI`. This method MAY be overridden by human intervention (or other means) on the origin server. The client cannot be guaranteed that the operation has been carried out, even if the status code returned from the origin server indicates that the action has been completed successfully. However, the server SHOULD NOT indicate success unless, at the time the response is given, it intends to delete the resource or move it to an inaccessible location.

A successful response SHOULD be 200 (OK) if the response includes an entity describing the status, 202 (Accepted) if the action has not yet been enacted, or 204 (No Content) if the action has been enacted but the response does not include an entity.

If the request passes through a cache and the `Request-URI` identifies one or more currently cached entities, those entries SHOULD be treated as stale. Responses to this method are not cacheable.

## 9.8 TRACE

The TRACE method is used to invoke a remote, application-layer loop-back of the request message. The final recipient of the request SHOULD reflect the message received back to the client as the entity-body of a 200 (OK) response. The final recipient is either the origin server or the first proxy or gateway to receive a `Max-Forwards` value of zero (0) in the request (see section 14.31). A TRACE request MUST NOT include an entity.

TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information. The value of the `Via` header field (section 14.45) is of particular interest, since it acts as a trace of the request chain. Use of the `Max-Forwards` header field allows the client to limit the length of the request chain, which is useful for testing a chain of proxies forwarding messages in an infinite loop.

If the request is valid, the response SHOULD contain the entire request message in the entity-body, with a `Content-Type` of "message/http". Responses to this method MUST NOT be cached.

## 9.9 CONNECT

This specification reserves the method name CONNECT for use with a proxy that can dynamically switch to being a tunnel (e.g. SSL tunneling [44]).

# 10 Status Code Definitions

Each `Status-Code` is described below, including a description of which method(s) it can follow and any metainformation required in the response.

## 10.1 Informational 1xx

This class of status code indicates a provisional response, consisting only of the `Status-Line` and optional headers, and is terminated by an empty line. There are no required headers for this class of status code. Since HTTP/1.0 did not define any 1xx status codes, servers MUST NOT send a 1xx response to an HTTP/1.0 client except under experimental conditions.

A client MUST be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message. Unexpected 1xx status responses MAY be ignored by a user agent.

Proxies MUST forward 1xx responses, unless the connection between the proxy and its client has been closed, or unless the proxy itself requested the generation of the 1xx response. (For example, if a proxy adds a "Expect: 100-continue" field when it forwards a request, then it need not forward the corresponding 100 (Continue) response(s).)

### 10.1.1 100 Continue

The client SHOULD continue with its request. This interim response is used to inform the client that the initial part of the request has been received and has not yet been rejected by the server. The client SHOULD continue by sending the remainder of the request or, if the request has already been completed, ignore this response. The server MUST send a final response after the request has been completed. See section 8.2.3 for detailed discussion of the use and handling of this status code.

### 10.1.2 101 Switching Protocols

The server understands and is willing to comply with the client's request, via the `Upgrade` message header field (section 14.42), for a change in the application protocol being used on this connection. The server will switch protocols to those defined by the response's `Upgrade` header field immediately after the empty line which terminates the 101 response.

The protocol **SHOULD** be switched only when it is advantageous to do so. For example, switching to a newer version of HTTP is advantageous over older versions, and switching to a real-time, synchronous protocol might be advantageous when delivering resources that use such features.

## 10.2 Successful 2xx

This class of status code indicates that the client's request was successfully received, understood, and accepted.

### 10.2.1 200 OK

The request has succeeded. The information returned with the response is dependent on the method used in the request, for example:

`GET` an entity corresponding to the requested resource is sent in the response;

`HEAD` the entity-header fields corresponding to the requested resource are sent in the response without any message-body;

`POST` an entity describing or containing the result of the action;

`TRACE` an entity containing the request message as received by the end server.

### 10.2.2 201 Created

The request has been fulfilled and resulted in a new resource being created. The newly created resource can be referenced by the URI(s) returned in the entity of the response, with the most specific URI for the resource given by a `Location` header field. The response **SHOULD** include an entity containing a list of resource characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the `Content-Type` header field. The origin server **MUST** create the resource before returning the 201 status code. If the action cannot be carried out immediately, the server **SHOULD** respond with 202 (Accepted) response instead.

A 201 response **MAY** contain an `ETag` response header field indicating the current value of the entity tag for the requested variant just created, see section 14.19.

### 10.2.3 202 Accepted

The request has been accepted for processing, but the processing has not been completed. The request might or might not eventually be acted upon, as it might be disallowed when processing actually takes place. There is no facility for re-sending a status code from an asynchronous operation such as this.

The 202 response is intentionally non-committal. Its purpose is to allow a server to accept a request for some other process (perhaps a batch-oriented process that is only run once per day) without requiring that the user agent's connection to the server persist until the process is completed. The entity returned with this response **SHOULD** include an indication of the request's current status and either a pointer to a status monitor or some estimate of when the user can expect the request to be fulfilled.

#### 10.2.4 203 Non-Authoritative Information

The returned metainformation in the entity-header is not the definitive set as available from the origin server, but is gathered from a local or a third-party copy. The set presented MAY be a subset or superset of the original version. For example, including local annotation information about the resource might result in a superset of the metainformation known by the origin server. Use of this response code is not required and is only appropriate when the response would otherwise be 200 (OK).

#### 10.2.5 204 No Content

The server has fulfilled the request but does not need to return an entity-body, and might want to return updated metainformation. The response MAY include new or updated metainformation in the form of entity-headers, which if present SHOULD be associated with the requested variant.

If the client is a user agent, it SHOULD NOT change its document view from that which caused the request to be sent. This response is primarily intended to allow input for actions to take place without causing a change to the user agent's active document view, although any new or updated metainformation SHOULD be applied to the document currently in the user agent's active view.

The 204 response MUST NOT include a message-body, and thus is always terminated by the first empty line after the header fields.

#### 10.2.6 205 Reset Content

The server has fulfilled the request and the user agent SHOULD reset the document view which caused the request to be sent. This response is primarily intended to allow input for actions to take place via user input, followed by a clearing of the form in which the input is given so that the user can easily initiate another input action. The response MUST NOT include an entity.

#### 10.2.7 206 Partial Content

The server has fulfilled the partial GET request for the resource. The request MUST have included a Range header field (section 14.35) indicating the desired range, and MAY have included an If-Range header field (section 14.27) to make the request conditional.

The response MUST include the following header fields:

- Either a Content-Range header field (section 14.16) indicating the range included with this response, or a multipart/byteranges Content-Type including Content-Range fields for each part. If a Content-Length header field is present in the response, its value MUST match the actual number of OCTETs transmitted in the message-body.
- Date
- ETag and/or Content-Location, if the header would have been sent in a 200 response to the same request
- Expires, Cache-Control, and/or Vary, if the field-value might differ from that sent in any previous response for the same variant

If the 206 response is the result of an If-Range request that used a strong cache validator (see section 13.3.3), the response SHOULD NOT include other entity-headers. If the response is the result of an If-Range request that used a weak validator, the response MUST NOT include other entity-headers; this prevents inconsistencies between cached entity-bodies and updated headers. Otherwise, the response MUST include all of the entity-headers that would have been returned with a 200 (OK) response to the same request.

A cache MUST NOT combine a 206 response with other previously cached content if the ETag or Last-Modified headers do not match exactly, see 13.5.4.

A cache that does not support the `Range` and `Content-Range` headers **MUST NOT** cache 206 (Partial) responses.

### 10.3 Redirection 3xx

This class of status code indicates that further action needs to be taken by the user agent in order to fulfill the request. The action required **MAY** be carried out by the user agent without interaction with the user if and only if the method used in the second request is `GET` or `HEAD`. A client **SHOULD** detect infinite redirection loops, since such loops generate network traffic for each redirection.

Note: previous versions of this specification recommended a maximum of five redirections. Content developers should be aware that there might be clients that implement such a fixed limitation.

#### 10.3.1 300 Multiple Choices

The requested resource corresponds to any one of a set of representations, each with its own specific location, and agent-driven negotiation information (section 12) is being provided so that the user (or user agent) can select a preferred representation and redirect its request to that location.

Unless it was a `HEAD` request, the response **SHOULD** include an entity containing a list of resource characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the `Content-Type` header field. Depending upon the format and the capabilities of the user agent, selection of the most appropriate choice **MAY** be performed automatically. However, this specification does not define any standard for such automatic selection.

If the server has a preferred choice of representation, it **SHOULD** include the specific URI for that representation in the `Location` field; user agents **MAY** use the `Location` field value for automatic redirection. This response is cacheable unless indicated otherwise.

#### 10.3.2 301 Moved Permanently

The requested resource has been assigned a new permanent URI and any future references to this resource **SHOULD** use one of the returned URIs. Clients with link editing capabilities ought to automatically re-link references to the `Request-URI` to one or more of the new references returned by the server, where possible. This response is cacheable unless indicated otherwise.

The new permanent URI **SHOULD** be given by the `Location` field in the response. Unless the request method was `HEAD`, the entity of the response **SHOULD** contain a short hypertext note with a hyperlink to the new URI(s).

If the 301 status code is received in response to a request other than `GET` or `HEAD`, the user agent **MUST NOT** automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

Note: When automatically redirecting a `POST` request after receiving a 301 status code, some existing HTTP/1.0 user agents will erroneously change it into a `GET` request.

#### 10.3.3 302 Found

The requested resource resides temporarily under a different URI. Since the redirection might be altered on occasion, the client **SHOULD** continue to use the `Request-URI` for future requests. This response is only cacheable if indicated by a `Cache-Control` or `Expires` header field.

The temporary URI **SHOULD** be given by the `Location` field in the response. Unless the request method was `HEAD`, the entity of the response **SHOULD** contain a short hypertext note with a hyperlink to the new URI(s).

If the 302 status code is received in response to a request other than `GET` or `HEAD`, the user agent **MUST NOT** automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

Note: RFC 1945 and RFC 2068 specify that the client is not allowed to change the method on the redirected request. However, most existing user agent implementations treat 302 as if it were a 303 response, performing a GET on the `Location` field-value regardless of the original request method. The status codes 303 and 307 have been added for servers that wish to make unambiguously clear which kind of reaction is expected of the client.

#### 10.3.4 303 See Other

The response to the request can be found under a different URI and SHOULD be retrieved using a GET method on that resource. This method exists primarily to allow the output of a POST-activated script to redirect the user agent to a selected resource. The new URI is not a substitute reference for the originally requested resource. The 303 response MUST NOT be cached, but the response to the second (redirected) request might be cacheable.

The different URI SHOULD be given by the `Location` field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).

Note: Many pre-HTTP/1.1 user agents do not understand the 303 status. When interoperability with such clients is a concern, the 302 status code may be used instead, since most user agents react to a 302 response as described here for 303.

#### 10.3.5 304 Not Modified

If the client has performed a conditional GET request and access is allowed, but the document has not been modified, the server SHOULD respond with this status code. The 304 response MUST NOT contain a message-body, and thus is always terminated by the first empty line after the header fields.

The response MUST include the following header fields:

- `Date`, unless its omission is required by section 14.18.1  
If a clockless origin server obeys these rules, and proxies and clients add their own `Date` to any response received without one (as already specified by [RFC 2068], section 14.19), caches will operate correctly.
- `ETag` and/or `Content-Location`, if the header would have been sent in a 200 response to the same request
- `Expires`, `Cache-Control`, and/or `Vary`, if the field-value might differ from that sent in any previous response for the same variant

If the conditional GET used a strong cache validator (see section 13.3.3), the response SHOULD NOT include other entity-headers. Otherwise (i.e., the conditional GET used a weak validator), the response MUST NOT include other entity-headers; this prevents inconsistencies between cached entity-bodies and updated headers.

If a 304 response indicates an entity not currently cached, then the cache MUST disregard the response and repeat the request without the conditional.

If a cache uses a received 304 response to update a cache entry, the cache MUST update the entry to reflect any new field values given in the response.

#### 10.3.6 305 Use Proxy

The requested resource MUST be accessed through the proxy given by the `Location` field. The `Location` field gives the URI of the proxy. The recipient is expected to repeat this single request via the proxy. 305 responses MUST only be generated by origin servers.

Note: RFC 2068 was not clear that 305 was intended to redirect a single request, and to be generated by origin servers only. Not observing these limitations has significant security consequences.

#### 10.3.7 306 (Unused)

The 306 status code was used in a previous version of the specification, is no longer used, and the code is reserved.

### 10.3.8 307 Temporary Redirect

The requested resource resides temporarily under a different URI. Since the redirection MAY be altered on occasion, the client SHOULD continue to use the `Request-URI` for future requests. This response is only cacheable if indicated by a `Cache-Control` or `Expires` header field.

The temporary URI SHOULD be given by the `Location` field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s), since many pre-HTTP/1.1 user agents do not understand the 307 status. Therefore, the note SHOULD contain the information necessary for a user to repeat the original request on the new URI.

If the 307 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

## 10.4 Client Error 4xx

The 4xx class of status code is intended for cases in which the client seems to have erred. Except when responding to a HEAD request, the server SHOULD include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. These status codes are applicable to any request method. User agents SHOULD display any included entity to the user.

If the client is sending data, a server implementation using TCP SHOULD be careful to ensure that the client acknowledges receipt of the packet(s) containing the response, before the server closes the input connection. If the client continues sending data to the server after the close, the server's TCP stack will send a reset packet to the client, which may erase the client's unacknowledged input buffers before they can be read and interpreted by the HTTP application.

### 10.4.1 400 Bad Request

The request could not be understood by the server due to malformed syntax. The client SHOULD NOT repeat the request without modifications.

### 10.4.2 401 Unauthorized

The request requires user authentication. The response MUST include a `WWW-Authenticate` header field (section 14.47) containing a challenge applicable to the requested resource. The client MAY repeat the request with a suitable `Authorization` header field (section 14.8). If the request already included `Authorization` credentials, then the 401 response indicates that authorization has been refused for those credentials. If the 401 response contains the same challenge as the prior response, and the user agent has already attempted authentication at least once, then the user SHOULD be presented the entity that was given in the response, since that entity might include relevant diagnostic information. HTTP access authentication is explained in "HTTP Authentication: Basic and Digest Access Authentication" [43].

### 10.4.3 402 Payment Required

This code is reserved for future use.

### 10.4.4 403 Forbidden

The server understood the request, but is refusing to fulfill it. Authorization will not help and the request SHOULD NOT be repeated. If the request method was not HEAD and the server wishes to make public why the request has not been fulfilled, it SHOULD describe the reason for the refusal in the entity. If the server does not wish to make this information available to the client, the status code 404 (Not Found) can be used instead.

#### 10.4.5 404 Not Found

The server has not found anything matching the `Request-URI`. No indication is given of whether the condition is temporary or permanent. The 410 (Gone) status code **SHOULD** be used if the server knows, through some internally configurable mechanism, that an old resource is permanently unavailable and has no forwarding address. This status code is commonly used when the server does not wish to reveal exactly why the request has been refused, or when no other response is applicable.

#### 10.4.6 405 Method Not Allowed

The method specified in the `Request-Line` is not allowed for the resource identified by the `Request-URI`. The response **MUST** include an `Allow` header containing a list of valid methods for the requested resource.

#### 10.4.7 406 Not Acceptable

The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request.

Unless it was a HEAD request, the response **SHOULD** include an entity containing a list of available entity characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the `Content-Type` header field. Depending upon the format and the capabilities of the user agent, selection of the most appropriate choice **MAY** be performed automatically. However, this specification does not define any standard for such automatic selection.

Note: HTTP/1.1 servers are allowed to return responses which are not acceptable according to the accept headers sent in the request. In some cases, this may even be preferable to sending a 406 response. User agents are encouraged to inspect the headers of an incoming response to determine if it is acceptable.

If the response could be unacceptable, a user agent **SHOULD** temporarily stop receipt of more data and query the user for a decision on further actions.

#### 10.4.8 407 Proxy Authentication Required

This code is similar to 401 (Unauthorized), but indicates that the client must first authenticate itself with the proxy. The proxy **MUST** return a `Proxy-Authenticate` header field (section 14.33) containing a challenge applicable to the proxy for the requested resource. The client **MAY** repeat the request with a suitable `Proxy-Authorization` header field (section 14.34). HTTP access authentication is explained in "HTTP Authentication: Basic and Digest Access Authentication" [43].

#### 10.4.9 408 Request Timeout

The client did not produce a request within the time that the server was prepared to wait. The client **MAY** repeat the request without modifications at any later time.

#### 10.4.10 409 Conflict

The request could not be completed due to a conflict with the current state of the resource. This code is only allowed in situations where it is expected that the user might be able to resolve the conflict and resubmit the request. The response body **SHOULD** include enough information for the user to recognize the source of the conflict. Ideally, the response entity would include enough information for the user or user agent to fix the problem; however, that might not be possible and is not required.

Conflicts are most likely to occur in response to a PUT request. For example, if versioning were being used and the entity being PUT included changes to a resource which conflict with those made by an earlier (third-party) request, the server might use the 409 response to indicate that it can't complete the request. In this case, the response entity would likely contain a list of the differences between the two versions in a format defined by the response `Content-Type`.



#### 10.4.11 410 Gone

The requested resource is no longer available at the server and no forwarding address is known. This condition is expected to be considered permanent. Clients with link editing capabilities SHOULD delete references to the `Request-URI` after user approval. If the server does not know, or has no facility to determine, whether or not the condition is permanent, the status code 404 (Not Found) SHOULD be used instead. This response is cacheable unless indicated otherwise.

The 410 response is primarily intended to assist the task of web maintenance by notifying the recipient that the resource is intentionally unavailable and that the server owners desire that remote links to that resource be removed. Such an event is common for limited-time, promotional services and for resources belonging to individuals no longer working at the server's site. It is not necessary to mark all permanently unavailable resources as "gone" or to keep the mark for any length of time -- that is left to the discretion of the server owner.

#### 10.4.12 411 Length Required

The server refuses to accept the request without a defined `Content-Length`. The client MAY repeat the request if it adds a valid `Content-Length` header field containing the length of the message-body in the request message.

#### 10.4.13 412 Precondition Failed

The precondition given in one or more of the request-header fields evaluated to false when it was tested on the server. This response code allows the client to place preconditions on the current resource meta-information (header field data) and thus prevent the requested method from being applied to a resource other than the one intended.

#### 10.4.14 413 Request Entity Too Large

The server is refusing to process a request because the request entity is larger than the server is willing or able to process. The server MAY close the connection to prevent the client from continuing the request.

If the condition is temporary, the server SHOULD include a `Retry-After` header field to indicate that it is temporary and after what time the client MAY try again.

#### 10.4.15 414 Request-URI Too Long

The server is refusing to service the request because the `Request-URI` is longer than the server is willing to interpret. This rare condition is only likely to occur when a client has improperly converted a POST request to a GET request with long query information, when the client has descended into a URI "black hole" of redirection (e.g., a redirected URI prefix that points to a suffix of itself), or when the server is under attack by a client attempting to exploit security holes present in some servers using fixed-length buffers for reading or manipulating the `Request-URI`.

#### 10.4.16 415 Unsupported Media Type

The server is refusing to service the request because the entity of the request is in a format not supported by the requested resource for the requested method.

#### 10.4.17 416 Requested Range Not Satisfiable

A server SHOULD return a response with this status code if a request included a `Range` request-header field (section 14.35), and none of the range-specifier values in this field overlap the current extent of the selected resource, and the request did not include an `If-Range` request-header field. (For byte-ranges, this means that the first-byte-pos of all of the byte-range-spec values were greater than the current length of the selected resource.)

When this status code is returned for a byte-range request, the response **SHOULD** include a `Content-Range` entity-header field specifying the current length of the selected resource (see section 14.16). This response **MUST NOT** use the `multipart/byteranges` content-type.

#### 10.4.18 417 Expectation Failed

The expectation given in an `Expect` request-header field (see section 14.20) could not be met by this server, or, if the server is a proxy, the server has unambiguous evidence that the request could not be met by the next-hop server.

## 10.5 Server Error 5xx

Response status codes beginning with the digit “5” indicate cases in which the server is aware that it has erred or is incapable of performing the request. Except when responding to a HEAD request, the server **SHOULD** include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. User agents **SHOULD** display any included entity to the user. These response codes are applicable to any request method.

### 10.5.1 500 Internal Server Error

The server encountered an unexpected condition which prevented it from fulfilling the request.

### 10.5.2 501 Not Implemented

The server does not support the functionality required to fulfill the request. This is the appropriate response when the server does not recognize the request method and is not capable of supporting it for any resource.

### 10.5.3 502 Bad Gateway

The server, while acting as a gateway or proxy, received an invalid response from the upstream server it accessed in attempting to fulfill the request.

### 10.5.4 503 Service Unavailable

The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. The implication is that this is a temporary condition which will be alleviated after some delay. If known, the length of the delay **MAY** be indicated in a `Retry-After` header. If no `Retry-After` is given, the client **SHOULD** handle the response as it would for a 500 response.

Note: The existence of the 503 status code does not imply that a server must use it when becoming overloaded. Some servers may wish to simply refuse the connection.

### 10.5.5 504 Gateway Timeout

The server, while acting as a gateway or proxy, did not receive a timely response from the upstream server specified by the URI (e.g. HTTP, FTP, LDAP) or some other auxiliary server (e.g. DNS) it needed to access in attempting to complete the request.

Note: Note to implementors: some deployed proxies are known to return 400 or 500 when DNS lookups time out.

### 10.5.6 505 HTTP Version Not Supported

The server does not support, or refuses to support, the HTTP protocol version that was used in the request message. The server is indicating that it is unable or unwilling to complete the request using the same major version as the client, as described in section 3.1, other than with this error message. The response **SHOULD** contain an entity describing why that version is not supported and what other protocols are supported by that server.

## 11 Access Authentication

HTTP provides several OPTIONAL challenge-response authentication mechanisms which can be used by a server to challenge a client request and by a client to provide authentication information. The general framework for access authentication, and the specification of “basic” and “digest” authentication, are specified in “HTTP Authentication: Basic and Digest Access Authentication” [43]. This specification adopts the definitions of “challenge” and “credentials” from that specification.

## 12 Content Negotiation

Most HTTP responses include an entity which contains information for interpretation by a human user. Naturally, it is desirable to supply the user with the “best available” entity corresponding to the request. Unfortunately for servers and caches, not all users have the same preferences for what is “best,” and not all user agents are equally capable of rendering all entity types. For that reason, HTTP has provisions for several mechanisms for “content negotiation” -- the process of selecting the best representation for a given response when there are multiple representations available.

Note: This is not called “format negotiation” because the alternate representations may be of the same media type, but use different capabilities of that type, be in different languages, etc.

Any response containing an entity-body MAY be subject to negotiation, including error responses.

There are two kinds of content negotiation which are possible in HTTP: server-driven and agent-driven negotiation. These two kinds of negotiation are orthogonal and thus may be used separately or in combination. One method of combination, referred to as transparent negotiation, occurs when a cache uses the agent-driven negotiation information provided by the origin server in order to provide server-driven negotiation for subsequent requests.

### 12.1 Server-driven Negotiation

If the selection of the best representation for a response is made by an algorithm located at the server, it is called server-driven negotiation. Selection is based on the available representations of the response (the dimensions over which it can vary; e.g. language, content-coding, etc.) and the contents of particular header fields in the request message or on other information pertaining to the request (such as the network address of the client).

Server-driven negotiation is advantageous when the algorithm for selecting from among the available representations is difficult to describe to the user agent, or when the server desires to send its “best guess” to the client along with the first response (hoping to avoid the round-trip delay of a subsequent request if the “best guess” is good enough for the user). In order to improve the server's guess, the user agent MAY include request header fields (`Accept`, `Accept-Language`, `Accept-Encoding`, etc.) which describe its preferences for such a response.

Server-driven negotiation has disadvantages:

1. It is impossible for the server to accurately determine what might be “best” for any given user, since that would require complete knowledge of both the capabilities of the user agent and the intended use for the response (e.g., does the user want to view it on screen or print it on paper?).
2. Having the user agent describe its capabilities in every request can be both very inefficient (given that only a small percentage of responses have multiple representations) and a potential violation of the user's privacy.
3. It complicates the implementation of an origin server and the algorithms for generating responses to a request.
4. It may limit a public cache's ability to use the same response for multiple user's requests.

HTTP/1.1 includes the following request-header fields for enabling server-driven negotiation through description of user agent capabilities and user preferences: `Accept` (section 14.1), `Accept-Charset` (section 14.2), `Accept-`

Encoding (section 14.3), Accept-Language (section 14.4), and User-Agent (section 14.43). However, an origin server is not limited to these dimensions and MAY vary the response based on any aspect of the request, including information outside the request-header fields or within extension header fields not defined by this specification.

The Vary header field can be used to express the parameters the server uses to select a representation that is subject to server-driven negotiation. See section 13.6 for use of the Vary header field by caches and section 14.44 for use of the Vary header field by servers.

## 12.2 Agent-driven Negotiation

With agent-driven negotiation, selection of the best representation for a response is performed by the user agent after receiving an initial response from the origin server. Selection is based on a list of the available representations of the response included within the header fields or entity-body of the initial response, with each representation identified by its own URI. Selection from among the representations may be performed automatically (if the user agent is capable of doing so) or manually by the user selecting from a generated (possibly hypertext) menu.

Agent-driven negotiation is advantageous when the response would vary over commonly-used dimensions (such as type, language, or encoding), when the origin server is unable to determine a user agent's capabilities from examining the request, and generally when public caches are used to distribute server load and reduce network usage.

Agent-driven negotiation suffers from the disadvantage of needing a second request to obtain the best alternate representation. This second request is only efficient when caching is used. In addition, this specification does not define any mechanism for supporting automatic selection, though it also does not prevent any such mechanism from being developed as an extension and used within HTTP/1.1.

HTTP/1.1 defines the 300 (Multiple Choices) and 406 (Not Acceptable) status codes for enabling agent-driven negotiation when the server is unwilling or unable to provide a varying response using server-driven negotiation.

## 12.3 Transparent Negotiation

Transparent negotiation is a combination of both server-driven and agent-driven negotiation. When a cache is supplied with a form of the list of available representations of the response (as in agent-driven negotiation) and the dimensions of variance are completely understood by the cache, then the cache becomes capable of performing server-driven negotiation on behalf of the origin server for subsequent requests on that resource.

Transparent negotiation has the advantage of distributing the negotiation work that would otherwise be required of the origin server and also removing the second request delay of agent-driven negotiation when the cache is able to correctly guess the right response.

This specification does not define any mechanism for transparent negotiation, though it also does not prevent any such mechanism from being developed as an extension that could be used within HTTP/1.1.

## 13 Caching in HTTP

HTTP is typically used for distributed information systems, where performance can be improved by the use of response caches. The HTTP/1.1 protocol includes a number of elements intended to make caching work as well as possible. Because these elements are inextricable from other aspects of the protocol, and because they interact with each other, it is useful to describe the basic caching design of HTTP separately from the detailed descriptions of methods, headers, response codes, etc.

Caching would be useless if it did not significantly improve performance. The goal of caching in HTTP/1.1 is to eliminate the need to send requests in many cases, and to eliminate the need to send full responses in many other cases. The former reduces the number of network round-trips required for many operations; we use an "expiration" mechanism for this purpose (see section 13.2). The latter reduces network bandwidth requirements; we use a "validation" mechanism for this purpose (see section 13.3).

Requirements for performance, availability, and disconnected operation require us to be able to relax the goal of semantic transparency. The HTTP/1.1 protocol allows origin servers, caches, and clients to explicitly reduce transparency when necessary. However, because non-transparent operation may confuse non-expert users, and might be incompatible with certain server applications (such as those for ordering merchandise), the protocol requires that transparency be relaxed

- only by an explicit protocol-level request when relaxed by client or origin server
- only with an explicit warning to the end user when relaxed by cache or client

Therefore, the HTTP/1.1 protocol provides these important elements:

1. Protocol features that provide full semantic transparency when this is required by all parties.
2. Protocol features that allow an origin server or user agent to explicitly request and control non-transparent operation.
3. Protocol features that allow a cache to attach warnings to responses that do not preserve the requested approximation of semantic transparency.

A basic principle is that it must be possible for the clients to detect any potential relaxation of semantic transparency.

Note: The server, cache, or client implementor might be faced with design decisions not explicitly discussed in this specification. If a decision might affect semantic transparency, the implementor ought to err on the side of maintaining transparency unless a careful and complete analysis shows significant benefits in breaking transparency.

### 13.1.1 Cache Correctness

A correct cache **MUST** respond to a request with the most up-to-date response held by the cache that is appropriate to the request (see sections 13.2.5, 13.2.6, and 13.12) which meets one of the following conditions:

1. It has been checked for equivalence with what the origin server would have returned by revalidating the response with the origin server (section 13.3);
2. It is “fresh enough” (see section 13.2). In the default case, this means it meets the least restrictive freshness requirement of the client, origin server, and cache (see section 14.9); if the origin server so specifies, it is the freshness requirement of the origin server alone.

If a stored response is not “fresh enough” by the most restrictive freshness requirement of both the client and the origin server, in carefully considered circumstances the cache **MAY** still return the response with the appropriate `Warning` header (see section 13.1.5 and 14.46), unless such a response is prohibited (e.g., by a “no-store” cache-directive, or by a “no-cache” cache-request-directive; see section 14.9).

3. It is an appropriate 304 (Not Modified), 305 (Proxy Redirect), or error (4xx or 5xx) response message.

If the cache can not communicate with the origin server, then a correct cache **SHOULD** respond as above if the response can be correctly served from the cache; if not it **MUST** return an error or warning indicating that there was a communication failure.

If a cache receives a response (either an entire response, or a 304 (Not Modified) response) that it would normally forward to the requesting client, and the received response is no longer fresh, the cache **SHOULD** forward it to the requesting client without adding a new `Warning` (but without removing any existing `Warning` headers). A cache **SHOULD NOT** attempt to revalidate a response simply because that response became stale in transit; this might lead to an infinite loop. A user agent that receives a stale response without a `Warning` **MAY** display a warning indication to the user.

### 13.1.2 Warnings

Whenever a cache returns a response that is neither first-hand nor “fresh enough” (in the sense of condition 2 in section 13.1.1), it **MUST** attach a warning to that effect, using a `Warning` general-header. The `Warning` header and the currently defined warnings are described in section 14.46. The warning allows clients to take appropriate action.

Warnings **MAY** be used for other purposes, both cache-related and otherwise. The use of a warning, rather than an error status code, distinguish these responses from true failures.

Warnings are assigned three digit `warn-codes`. The first digit indicates whether the `Warning` **MUST** or **MUST NOT** be deleted from a stored cache entry after a successful revalidation:

1xx Warnings that describe the freshness or revalidation status of the response, and so **MUST** be deleted after a successful revalidation. 1XX `warn-codes` **MAY** be generated by a cache only when validating a cached entry. It **MUST NOT** be generated by clients.

2xx Warnings that describe some aspect of the entity body or entity headers that is not rectified by a revalidation (for example, a lossy compression of the entity bodies) and which **MUST NOT** be deleted after a successful revalidation.

See section 14.46 for the definitions of the codes themselves.

HTTP/1.0 caches will cache all `Warnings` in responses, without deleting the ones in the first category. Warnings in responses that are passed to HTTP/1.0 caches carry an extra `warning-date` field, which prevents a future HTTP/1.1 recipient from believing an erroneously cached `Warning`.

Warnings also carry a warning text. The text **MAY** be in any appropriate natural language (perhaps based on the client's `Accept` headers), and include an **OPTIONAL** indication of what character set is used.

Multiple warnings **MAY** be attached to a response (either by the origin server or by a cache), including multiple warnings with the same code number. For example, a server might provide the same warning with texts in both English and Basque.

When multiple warnings are attached to a response, it might not be practical or reasonable to display all of them to the user. This version of HTTP does not specify strict priority rules for deciding which warnings to display and in what order, but does suggest some heuristics.

### 13.1.3 Cache-control Mechanisms

The basic cache mechanisms in HTTP/1.1 (server-specified expiration times and validators) are implicit directives to caches. In some cases, a server or client might need to provide explicit directives to the HTTP caches. We use the `Cache-Control` header for this purpose.

The `Cache-Control` header allows a client or server to transmit a variety of directives in either requests or responses. These directives typically override the default caching algorithms. As a general rule, if there is any apparent conflict between header values, the most restrictive interpretation is applied (that is, the one that is most likely to preserve semantic transparency). However, in some cases, cache-control directives are explicitly specified as weakening the approximation of semantic transparency (for example, “`max-stale`” or “`public`”).

The cache-control directives are described in detail in section 14.9.

### 13.1.4 Explicit User Agent Warnings

Many user agents make it possible for users to override the basic caching mechanisms. For example, the user agent might allow the user to specify that cached entities (even explicitly stale ones) are never validated. Or the user agent might habitually add “`Cache-Control: max-stale=3600`” to every request. The user agent **SHOULD NOT** default to either non-transparent behavior, or behavior that results in abnormally ineffective caching, but **MAY** be explicitly configured to do so by an explicit action of the user.

If the user has overridden the basic caching mechanisms, the user agent SHOULD explicitly indicate to the user whenever this results in the display of information that might not meet the server's transparency requirements (in particular, if the displayed entity is known to be stale). Since the protocol normally allows the user agent to determine if responses are stale or not, this indication need only be displayed when this actually happens. The indication need not be a dialog box; it could be an icon (for example, a picture of a rotting fish) or some other indicator.

If the user has overridden the caching mechanisms in a way that would abnormally reduce the effectiveness of caches, the user agent SHOULD continually indicate this state to the user (for example, by a display of a picture of currency in flames) so that the user does not inadvertently consume excess resources or suffer from excessive latency.

### 13.1.5 Exceptions to the Rules and Warnings

In some cases, the operator of a cache MAY choose to return stale responses even when not requested by clients. This decision ought not be made lightly, but may be necessary for reasons of availability or performance, especially when the cache is poorly connected to the origin server. Whenever a cache returns a stale response, it MUST mark it as such (using a `Warning` header) enabling the client software to alert the user that there might be a potential problem.

It also allows the user agent to take steps to obtain a first-hand or fresh response. For this reason, a cache SHOULD NOT return a stale response if the client explicitly requests a first-hand or fresh one, unless it is impossible to comply for technical or policy reasons.

### 13.1.6 Client-controlled Behavior

While the origin server (and to a lesser extent, intermediate caches, by their contribution to the age of a response) are the primary source of expiration information, in some cases the client might need to control a cache's decision about whether to return a cached response without validating it. Clients do this using several directives of the `Cache-Control` header.

A client's request MAY specify the maximum age it is willing to accept of an unvalidated response; specifying a value of zero forces the cache(s) to revalidate all responses. A client MAY also specify the minimum time remaining before a response expires. Both of these options increase constraints on the behavior of caches, and so cannot further relax the cache's approximation of semantic transparency.

A client MAY also specify that it will accept stale responses, up to some maximum amount of staleness. This loosens the constraints on the caches, and so might violate the origin server's specified constraints on semantic transparency, but might be necessary to support disconnected operation, or high availability in the face of poor connectivity.

## 13.2 Expiration Model

### 13.2.1 Server-Specified Expiration

HTTP caching works best when caches can entirely avoid making requests to the origin server. The primary mechanism for avoiding requests is for an origin server to provide an explicit expiration time in the future, indicating that a response MAY be used to satisfy subsequent requests. In other words, a cache can return a fresh response without first contacting the server.

Our expectation is that servers will assign future explicit expiration times to responses in the belief that the entity is not likely to change, in a semantically significant way, before the expiration time is reached. This normally preserves semantic transparency, as long as the server's expiration times are carefully chosen.

The expiration mechanism applies only to responses taken from a cache and not to first-hand responses forwarded immediately to the requesting client.

If an origin server wishes to force a semantically transparent cache to validate every request, it MAY assign an explicit expiration time in the past. This means that the response is always stale, and so the cache SHOULD validate it before using it for subsequent requests. See section 14.9.4 for a more restrictive way to force revalidation.

If an origin server wishes to force any HTTP/1.1 cache, no matter how it is configured, to validate every request, it SHOULD use the “`must-revalidate`” cache-control directive (see section 14.9).

Servers specify explicit expiration times using either the `Expires` header, or the `max-age` directive of the `Cache-Control` header.

An expiration time cannot be used to force a user agent to refresh its display or reload a resource; its semantics apply only to caching mechanisms, and such mechanisms need only check a resource’s expiration status when a new request for that resource is initiated. See section 13.13 for an explanation of the difference between caches and history mechanisms.

### 13.2.2 Heuristic Expiration

Since origin servers do not always provide explicit expiration times, HTTP caches typically assign heuristic expiration times, employing algorithms that use other header values (such as the `Last-Modified` time) to estimate a plausible expiration time. The HTTP/1.1 specification does not provide specific algorithms, but does impose worst-case constraints on their results. Since heuristic expiration times might compromise semantic transparency, they ought to be used cautiously, and we encourage origin servers to provide explicit expiration times as much as possible.

### 13.2.3 Age Calculations

In order to know if a cached entry is fresh, a cache needs to know if its age exceeds its freshness lifetime. We discuss how to calculate the latter in section 13.2.4; this section describes how to calculate the age of a response or cache entry.

In this discussion, we use the term “now” to mean “the current value of the clock at the host performing the calculation.” Hosts that use HTTP, but especially hosts running origin servers and caches, SHOULD use NTP [28] or some similar protocol to synchronize their clocks to a globally accurate time standard.

HTTP/1.1 requires origin servers to send a `Date` header, if possible, with every response, giving the time at which the response was generated (see section 14.18). We use the term “`date_value`” to denote the value of the `Date` header, in a form appropriate for arithmetic operations.

HTTP/1.1 uses the `Age` response-header to convey the estimated age of the response message when obtained from a cache. The `Age` field value is the cache’s estimate of the amount of time since the response was generated or revalidated by the origin server.

In essence, the `Age` value is the sum of the time that the response has been resident in each of the caches along the path from the origin server, plus the amount of time it has been in transit along network paths.

We use the term “`age_value`” to denote the value of the `Age` header, in a form appropriate for arithmetic operations.

A response’s age can be calculated in two entirely independent ways:

1. `now` minus `date_value`, if the local clock is reasonably well synchronized to the origin server’s clock. If the result is negative, the result is replaced by zero.
2. `age_value`, if all of the caches along the response path implement HTTP/1.1.

Given that we have two independent ways to compute the age of a response when it is received, we can combine these as

$$\text{corrected\_received\_age} = \max(\text{now} - \text{date\_value}, \text{age\_value})$$

and as long as we have either nearly synchronized clocks or all-HTTP/1.1 paths, one gets a reliable (conservative) result.



Because of network-imposed delays, some significant interval might pass between the time that a server generates a response and the time it is received at the next outbound cache or client. If uncorrected, this delay could result in improperly low ages.

Because the request that resulted in the returned Age value must have been initiated prior to that Age value's generation, we can correct for delays imposed by the network by recording the time at which the request was initiated. Then, when an Age value is received, it MUST be interpreted relative to the time the request was initiated, not the time that the response was received. This algorithm results in conservative behavior no matter how much delay is experienced. So, we compute:

$$\text{corrected\_initial\_age} = \text{corrected\_received\_age} + (\text{now} - \text{request\_time})$$

where "request\_time" is the time (according to the local clock) when the request that elicited this response was sent.

Summary of age calculation algorithm, when a cache receives a response:

```

/*
 * age_value
 *   is the value of Age: header received by the cache with
 *   this response.
 * date_value
 *   is the value of the origin server's Date: header
 * request_time
 *   is the (local) time when the cache made the request
 *   that resulted in this cached response
 * response_time
 *   is the (local) time when the cache received the
 *   response
 * now
 *   is the current (local) time
 */
apparent_age = max(0, response_time - date_value);
corrected_received_age = max(apparent_age, age_value);
response_delay = response_time - request_time;
corrected_initial_age = corrected_received_age + response_delay;
resident_time = now - response_time;
current_age = corrected_initial_age + resident_time;

```

The `current_age` of a cache entry is calculated by adding the amount of time (in seconds) since the cache entry was last validated by the origin server to the `corrected_initial_age`. When a response is generated from a cache entry, the cache MUST include a single Age header field in the response with a value equal to the cache entry's `current_age`.

The presence of an Age header field in a response implies that a response is not first-hand. However, the converse is not true, since the lack of an Age header field in a response does not imply that the response is first-hand unless all caches along the request path are compliant with HTTP/1.1 (i.e., older HTTP caches did not implement the Age header field).

### 13.2.4 Expiration Calculations

In order to decide whether a response is fresh or stale, we need to compare its freshness lifetime to its age. The age is calculated as described in section 13.2.3; this section describes how to calculate the freshness lifetime, and to determine if a response has expired. In the discussion below, the values can be represented in any form appropriate for arithmetic operations.

We use the term "expires\_value" to denote the value of the Expires header. We use the term "max\_age\_value" to denote an appropriate value of the number of seconds carried by the "max-age" directive of the Cache-Control header in a response (see section 14.9.3).

The max-age directive takes priority over Expires, so if max-age is present in a response, the calculation is simply:

```
freshness_lifetime = max_age_value
```

Otherwise, if Expires is present in the response, the calculation is:

```
freshness_lifetime = expires_value - date_value
```

Note that neither of these calculations is vulnerable to clock skew, since all of the information comes from the origin server.

If none of Expires, Cache-Control: max-age, or Cache-Control: s-maxage (see section 14.9.3) appears in the response, and the response does not include other restrictions on caching, the cache MAY compute a freshness lifetime using a heuristic. The cache MUST attach Warning 113 to any response whose age is more than 24 hours if such warning has not already been added.

Also, if the response does have a Last-Modified time, the heuristic expiration value SHOULD be no more than some fraction of the interval since that time. A typical setting of this fraction might be 10%.

The calculation to determine if a response has expired is quite simple:

```
response_is_fresh = (freshness_lifetime > current_age)
```

### 13.2.5 Disambiguating Expiration Values

Because expiration values are assigned optimistically, it is possible for two caches to contain fresh values for the same resource that are different.

If a client performing a retrieval receives a non-first-hand response for a request that was already fresh in its own cache, and the Date header in its existing cache entry is newer than the Date on the new response, then the client MAY ignore the response. If so, it MAY retry the request with a "Cache-Control: max-age=0" directive (see section 14.9), to force a check with the origin server.

If a cache has two fresh responses for the same representation with different validators, it MUST use the one with the more recent Date header. This situation might arise because the cache is pooling responses from other caches, or because a client has asked for a reload or a revalidation of an apparently fresh cache entry.

### 13.2.6 Disambiguating Multiple Responses

Because a client might be receiving responses via multiple paths, so that some responses flow through one set of caches and other responses flow through a different set of caches, a client might receive responses in an order different from that in which the origin server sent them. We would like the client to use the most recently generated response, even if older responses are still apparently fresh.

Neither the entity tag nor the expiration value can impose an ordering on responses, since it is possible that a later response intentionally carries an earlier expiration time. The Date values are ordered to a granularity of one second.

When a client tries to revalidate a cache entry, and the response it receives contains a Date header that appears to be older than the one for the existing entry, then the client SHOULD repeat the request unconditionally, and include

```
Cache-Control: max-age=0
```

to force any intermediate caches to validate their copies directly with the origin server, or

```
Cache-Control: no-cache
```

to force any intermediate caches to obtain a new copy from the origin server.

If the Date values are equal, then the client MAY use either response (or MAY, if it is being extremely prudent, request a new response). Servers MUST NOT depend on clients being able to choose deterministically between responses generated during the same second, if their expiration times overlap.

## 13.3 Validation Model

When a cache has a stale entry that it would like to use as a response to a client's request, it first has to check with the origin server (or possibly an intermediate cache with a fresh response) to see if its cached entry is still usable. We call this "validating" the cache entry. Since we do not want to have to pay the overhead of retransmitting the full

response if the cached entry is good, and we do not want to pay the overhead of an extra round trip if the cached entry is invalid, the HTTP/1.1 protocol supports the use of conditional methods.

The key protocol features for supporting conditional methods are those concerned with “cache validators.” When an origin server generates a full response, it attaches some sort of validator to it, which is kept with the cache entry. When a client (user agent or proxy cache) makes a conditional request for a resource for which it has a cache entry, it includes the associated validator in the request.

The server then checks that validator against the current validator for the entity, and, if they match (see section 13.3.3), it responds with a special status code (usually, 304 (Not Modified)) and no entity-body. Otherwise, it returns a full response (including entity-body). Thus, we avoid transmitting the full response if the validator matches, and we avoid an extra round trip if it does not match.

In HTTP/1.1, a conditional request looks exactly the same as a normal request for the same resource, except that it carries a special header (which includes the validator) that implicitly turns the method (usually, GET) into a conditional.

The protocol includes both positive and negative senses of cache-validating conditions. That is, it is possible to request either that a method be performed if and only if a validator matches or if and only if no validators match.

Note: a response that lacks a validator may still be cached, and served from cache until it expires, unless this is explicitly prohibited by a cache-control directive. However, a cache cannot do a conditional retrieval if it does not have a validator for the entity, which means it will not be refreshable after it expires.

### 13.3.1 Last-Modified Dates

The `Last-Modified` entity-header field value is often used as a cache validator. In simple terms, a cache entry is considered to be valid if the entity has not been modified since the `Last-Modified` value.

### 13.3.2 Entity Tag Cache Validators

The `ETag` response-header field value, an entity tag, provides for an “opaque” cache validator. This might allow more reliable validation in situations where it is inconvenient to store modification dates, where the one-second resolution of HTTP date values is not sufficient, or where the origin server wishes to avoid certain paradoxes that might arise from the use of modification dates.

Entity Tags are described in section 3.11. The headers used with entity tags are described in sections 14.19, 14.24, 14.26 and 14.44.

### 13.3.3 Weak and Strong Validators

Since both origin servers and caches will compare two validators to decide if they represent the same or different entities, one normally would expect that if the entity (the entity-body or any entity-headers) changes in any way, then the associated validator would change as well. If this is true, then we call this validator a “strong validator.”

However, there might be cases when a server prefers to change the validator only on semantically significant changes, and not when insignificant aspects of the entity change. A validator that does not always change when the resource changes is a “weak validator.”

Entity tags are normally “strong validators,” but the protocol provides a mechanism to tag an entity tag as “weak.” One can think of a strong validator as one that changes whenever the bits of an entity changes, while a weak value changes whenever the meaning of an entity changes. Alternatively, one can think of a strong validator as part of an identifier for a specific entity, while a weak validator is part of an identifier for a set of semantically equivalent entities.

Note: One example of a strong validator is an integer that is incremented in stable storage every time an entity is changed.

An entity's modification time, if represented with one-second resolution, could be a weak validator, since it is possible that the resource might be modified twice during a single second.

Support for weak validators is optional. However, weak validators allow for more efficient caching of equivalent objects; for example, a hit counter on a site is probably good enough if it is updated every few days or weeks, and any value during that period is likely "good enough" to be equivalent.

A "use" of a validator is either when a client generates a request and includes the validator in a validating header field, or when a server compares two validators.

Strong validators are usable in any context. Weak validators are only usable in contexts that do not depend on exact equality of an entity. For example, either kind is usable for a conditional GET of a full entity. However, only a strong validator is usable for a sub-range retrieval, since otherwise the client might end up with an internally inconsistent entity.

Clients MAY issue simple (non-subrange) GET requests with either weak validators or strong validators. Clients MUST NOT use weak validators in other forms of request.

The only function that the HTTP/1.1 protocol defines on validators is comparison. There are two validator comparison functions, depending on whether the comparison context allows the use of weak validators or not:

- The strong comparison function: in order to be considered equal, both validators MUST be identical in every way, and both MUST NOT be weak.
- The weak comparison function: in order to be considered equal, both validators MUST be identical in every way, but either or both of them MAY be tagged as "weak" without affecting the result.

An entity tag is strong unless it is explicitly tagged as weak. Section 3.11 gives the syntax for entity tags.

A Last-Modified time, when used as a validator in a request, is implicitly weak unless it is possible to deduce that it is strong, using the following rules:

- The validator is being compared by an origin server to the actual current validator for the entity and,
- That origin server reliably knows that the associated entity did not change twice during the second covered by the presented validator.

or

- The validator is about to be used by a client in an If-Modified-Since or If-Unmodified-Since header, because the client has a cache entry for the associated entity, and
- That cache entry includes a Date value, which gives the time when the origin server sent the original response, and
- The presented Last-Modified time is at least 60 seconds before the Date value.

or

- The validator is being compared by an intermediate cache to the validator stored in its cache entry for the entity, and
- That cache entry includes a Date value, which gives the time when the origin server sent the original response, and
- The presented Last-Modified time is at least 60 seconds before the Date value.

This method relies on the fact that if two different responses were sent by the origin server during the same second, but both had the same Last-Modified time, then at least one of those responses would have a Date value equal to its Last-Modified time. The arbitrary 60-second limit guards against the possibility that the Date and Last-Modified values are generated from different clocks, or at somewhat different times during the preparation of the response. An implementation MAY use a value larger than 60 seconds, if it is believed that 60 seconds is too short.

If a client wishes to perform a sub-range retrieval on a value for which it has only a `Last-Modified` time and no opaque validator, it **MAY** do this only if the `Last-Modified` time is strong in the sense described here.

A cache or origin server receiving a conditional request, other than a full-body GET request, **MUST** use the strong comparison function to evaluate the condition.

These rules allow HTTP/1.1 caches and clients to safely perform sub-range retrievals on values that have been obtained from HTTP/1.0 servers.

#### 13.3.4 Rules for When to Use Entity Tags and Last-Modified Dates

We adopt a set of rules and recommendations for origin servers, clients, and caches regarding when various validator types ought to be used, and for what purposes.

HTTP/1.1 origin servers:

- **SHOULD** send an entity tag validator unless it is not feasible to generate one.
- **MAY** send a weak entity tag instead of a strong entity tag, if performance considerations support the use of weak entity tags, or if it is unfeasible to send a strong entity tag.
- **SHOULD** send a `Last-Modified` value if it is feasible to send one, unless the risk of a breakdown in semantic transparency that could result from using this date in an `If-Modified-Since` header would lead to serious problems.

In other words, the preferred behavior for an HTTP/1.1 origin server is to send both a strong entity tag and a `Last-Modified` value.

In order to be legal, a strong entity tag **MUST** change whenever the associated entity value changes in any way. A weak entity tag **SHOULD** change whenever the associated entity changes in a semantically significant way.

Note: in order to provide semantically transparent caching, an origin server must avoid reusing a specific strong entity tag value for two different entities, or reusing a specific weak entity tag value for two semantically different entities. Cache entries might persist for arbitrarily long periods, regardless of expiration times, so it might be inappropriate to expect that a cache will never again attempt to validate an entry using a validator that it obtained at some point in the past.

HTTP/1.1 clients:

- If an entity tag has been provided by the origin server, **MUST** use that entity tag in any cache-conditional request (using `If-Match` or `If-None-Match`).
- If only a `Last-Modified` value has been provided by the origin server, **SHOULD** use that value in non-subrange cache-conditional requests (using `If-Modified-Since`).
- If only a `Last-Modified` value has been provided by an HTTP/1.0 origin server, **MAY** use that value in subrange cache-conditional requests (using `If-Unmodified-Since`). The user agent **SHOULD** provide a way to disable this, in case of difficulty.
- If both an entity tag and a `Last-Modified` value have been provided by the origin server, **SHOULD** use both validators in cache-conditional requests. This allows both HTTP/1.0 and HTTP/1.1 caches to respond appropriately.

An HTTP/1.1 origin server, upon receiving a conditional request that includes both a `Last-Modified` date (e.g., in an `If-Modified-Since` or `If-Unmodified-Since` header field) and one or more entity tags (e.g., in an `If-Match`, `If-None-Match`, or `If-Range` header field) as cache validators, **MUST NOT** return a response status of 304 (Not Modified) unless doing so is consistent with all of the conditional header fields in the request.

An HTTP/1.1 caching proxy, upon receiving a conditional request that includes both a `Last-Modified` date and one or more entity tags as cache validators, **MUST NOT** return a locally cached response to the client unless that cached response is consistent with all of the conditional header fields in the request.

Note: The general principle behind these rules is that HTTP/1.1 servers and clients should transmit as much non-redundant information as is available in their responses and requests. HTTP/1.1 systems receiving this information will make the most conservative assumptions about the validators they receive.

HTTP/1.0 clients and caches will ignore entity tags. Generally, last-modified values received or used by these systems will support transparent and efficient caching, and so HTTP/1.1 origin servers should provide `Last-Modified` values. In those rare cases where the use of a `Last-Modified` value as a validator by an HTTP/1.0 system could result in a serious problem, then HTTP/1.1 origin servers should not provide one.

### 13.3.5 Non-validating Conditionals

The principle behind entity tags is that only the service author knows the semantics of a resource well enough to select an appropriate cache validation mechanism, and the specification of any validator comparison function more complex than byte-equality would open up a can of worms. Thus, comparisons of any other headers (except `Last-Modified`, for compatibility with HTTP/1.0) are never used for purposes of validating a cache entry.

## 13.4 Response Cacheability

Unless specifically constrained by a cache-control (section 14.9) directive, a caching system *MAY* always store a successful response (see section 13.8) as a cache entry, *MAY* return it without validation if it is fresh, and *MAY* return it after successful validation. If there is neither a cache validator nor an explicit expiration time associated with a response, we do not expect it to be cached, but certain caches *MAY* violate this expectation (for example, when little or no network connectivity is available). A client can usually detect that such a response was taken from a cache by comparing the `Date` header to the current time.

Note: some HTTP/1.0 caches are known to violate this expectation without providing any `Warning`.

However, in some cases it might be inappropriate for a cache to retain an entity, or to return it in response to a subsequent request. This might be because absolute semantic transparency is deemed necessary by the service author, or because of security or privacy considerations. Certain cache-control directives are therefore provided so that the server can indicate that certain resource entities, or portions thereof, are not to be cached regardless of other considerations.

Note that section 14.8 normally prevents a shared cache from saving and returning a response to a previous request if that request included an `Authorization` header.

A response received with a status code of 200, 203, 206, 300, 301 or 410 *MAY* be stored by a cache and used in reply to a subsequent request, subject to the expiration mechanism, unless a cache-control directive prohibits caching. However, a cache that does not support the `Range` and `Content-Range` headers **MUST NOT** cache 206 (Partial Content) responses.

A response received with any other status code (e.g. status codes 302 and 307) **MUST NOT** be returned in a reply to a subsequent request unless there are cache-control directives or another header(s) that explicitly allow it. For example, these include the following: an `Expires` header (section 14.21); a "max-age", "s-maxage", "must-revalidate", "proxy-revalidate", "public" or "private" cache-control directive (section 14.9).

## 13.5 Constructing Responses From Caches

The purpose of an HTTP cache is to store information received in response to requests for use in responding to future requests. In many cases, a cache simply returns the appropriate parts of a response to the requester. However, if the cache holds a cache entry based on a previous response, it might have to combine parts of a new response with what is held in the cache entry.

### 13.5.1 End-to-end and Hop-by-hop Headers

For the purpose of defining the behavior of caches and non-caching proxies, we divide HTTP headers into two categories:

- End-to-end headers, which are transmitted to the ultimate recipient of a request or response. End-to-end headers in responses **MUST** be stored as part of a cache entry and **MUST** be transmitted in any response formed from a cache entry.
- Hop-by-hop headers, which are meaningful only for a single transport-level connection, and are not stored by caches or forwarded by proxies.

The following HTTP/1.1 headers are hop-by-hop headers:

- `Connection`
- `Keep-Alive`
- `Proxy-Authenticate`
- `Proxy-Authorization`
- `TE`
- `Trailers`
- `Transfer-Encoding`
- `Upgrade`

All other headers defined by HTTP/1.1 are end-to-end headers.

Other hop-by-hop headers **MUST** be listed in a `Connection` header, (section 14.10) to be introduced into HTTP/1.1 (or later).

### 13.5.2 Non-modifiable Headers

Some features of the HTTP/1.1 protocol, such as Digest Authentication, depend on the value of certain end-to-end headers. A transparent proxy **SHOULD NOT** modify an end-to-end header unless the definition of that header requires or specifically allows that.

A transparent proxy **MUST NOT** modify any of the following fields in a request or response, and it **MUST NOT** add any of these fields if not already present:

- `Content-Location`
- `Content-MD5`
- `ETag`
- `Last-Modified`

A transparent proxy **MUST NOT** modify any of the following fields in a response:

- `Expires`

but it **MAY** add any of these fields if not already present. If an `Expires` header is added, it **MUST** be given a field-value identical to that of the `Date` header in that response.

A proxy **MUST NOT** modify or add any of the following fields in a message that contains the `no-transform` cache-control directive, or in any request:

- `Content-Encoding`
- `Content-Range`
- `Content-Type`

A non-transparent proxy MAY modify or add these fields to a message that does not include `no-transform`, but if it does so, it MUST add a Warning 214 (Transformation applied) if one does not already appear in the message (see section 14.46).

Warning: unnecessary modification of end-to-end headers might cause authentication failures if stronger authentication mechanisms are introduced in later versions of HTTP. Such authentication mechanisms MAY rely on the values of header fields not listed here.

The `Content-Length` field of a request or response is added or deleted according to the rules in section 4.4. A transparent proxy MUST preserve the entity-length (section 7.2.2) of the entity-body, although it MAY change the transfer-length (section 4.4).

### 13.5.3 Combining Headers

When a cache makes a validating request to a server, and the server provides a 304 (Not Modified) response or a 206 (Partial Content) response, the cache then constructs a response to send to the requesting client.

If the status code is 304 (Not Modified), the cache uses the entity-body stored in the cache entry as the entity-body of this outgoing response. If the status code is 206 (Partial Content) and the `ETag` or `Last-Modified` headers match exactly, the cache MAY combine the contents stored in the cache entry with the new contents received in the response and use the result as the entity-body of this outgoing response, (see 13.5.4).

The end-to-end headers stored in the cache entry are used for the constructed response, except that

- any stored `Warning` headers with warn-code 1xx (see section 14.46) MUST be deleted from the cache entry and the forwarded response.
- any stored `Warning` headers with warn-code 2xx MUST be retained in the cache entry and the forwarded response.
- any end-to-end headers provided in the 304 or 206 response MUST replace the corresponding headers from the cache entry.

Unless the cache decides to remove the cache entry, it MUST also replace the end-to-end headers stored with the cache entry with corresponding headers received in the incoming response, except for `Warning` headers as described immediately above. If a header field-name in the incoming response matches more than one header in the cache entry, all such old headers MUST be replaced.

In other words, the set of end-to-end headers received in the incoming response overrides all corresponding end-to-end headers stored with the cache entry (except for stored `Warning` headers with warn-code 1xx, which are deleted even if not overridden).

Note: this rule allows an origin server to use a 304 (Not Modified) or a 206 (Partial Content) response to update any header associated with a previous response for the same entity or sub-ranges thereof, although it might not always be meaningful or correct to do so. This rule does not allow an origin server to use a 304 (Not Modified) or a 206 (Partial Content) response to entirely delete a header that it had provided with a previous response.

### 13.5.4 Combining Byte Ranges

A response might transfer only a subrange of the bytes of an entity-body, either because the request included one or more `Range` specifications, or because a connection was broken prematurely. After several such transfers, a cache might have received several ranges of the same entity-body.

If a cache has a stored non-empty set of subranges for an entity, and an incoming response transfers another subrange, the cache MAY combine the new subrange with the existing set if both the following conditions are met:

- Both the incoming response and the cache entry have a cache validator.
- The two cache validators match using the strong comparison function (see section 13.3.3).



If either requirement is not met, the cache **MUST** use only the most recent partial response (based on the `Date` values transmitted with every response, and using the incoming response if these values are equal or missing), and **MUST** discard the other partial information.

## 13.6 Caching Negotiated Responses

Use of server-driven content negotiation (section 12.1), as indicated by the presence of a `Vary` header field in a response, alters the conditions and procedure by which a cache can use the response for subsequent requests. See section 14.44 for use of the `Vary` header field by servers.

A server **SHOULD** use the `Vary` header field to inform a cache of what request-header fields were used to select among multiple representations of a cacheable response subject to server-driven negotiation. The set of header fields named by the `Vary` field value is known as the “selecting” request-headers.

When the cache receives a subsequent request whose `Request-URI` specifies one or more cache entries including a `Vary` header field, the cache **MUST NOT** use such a cache entry to construct a response to the new request unless all of the selecting request-headers present in the new request match the corresponding stored request-headers in the original request.

The selecting request-headers from two requests are defined to match if and only if the selecting request-headers in the first request can be transformed to the selecting request-headers in the second request by adding or removing linear white space (LWS) at places where this is allowed by the corresponding BNF, and/or combining multiple message-header fields with the same field name following the rules about message headers in section 4.2.

A `Vary` header field-value of “\*” always fails to match and subsequent requests on that resource can only be properly interpreted by the origin server.

If the selecting request header fields for the cached entry do not match the selecting request header fields of the new request, then the cache **MUST NOT** use a cached entry to satisfy the request unless it first relays the new request to the origin server in a conditional request and the server responds with 304 (Not Modified), including an entity tag or `Content-Location` that indicates the entity to be used.

If an entity tag was assigned to a cached representation, the forwarded request **SHOULD** be conditional and include the entity tags in an `If-None-Match` header field from all its cache entries for the resource. This conveys to the server the set of entities currently held by the cache, so that if any one of these entities matches the requested entity, the server can use the `ETag` header field in its 304 (Not Modified) response to tell the cache which entry is appropriate. If the entity-tag of the new response matches that of an existing entry, the new response **SHOULD** be used to update the header fields of the existing entry, and the result **MUST** be returned to the client.

If any of the existing cache entries contains only partial content for the associated entity, its entity-tag **SHOULD NOT** be included in the `If-None-Match` header field unless the request is for a range that would be fully satisfied by that entry.

If a cache receives a successful response whose `Content-Location` field matches that of an existing cache entry for the same `Request-URI`, whose entity-tag differs from that of the existing entry, and whose `Date` is more recent than that of the existing entry, the existing entry **SHOULD NOT** be returned in response to future requests and **SHOULD** be deleted from the cache.

## 13.7 Shared and Non-Shared Caches

For reasons of security and privacy, it is necessary to make a distinction between “shared” and “non-shared” caches. A non-shared cache is one that is accessible only to a single user. Accessibility in this case **SHOULD** be enforced by appropriate security mechanisms. All other caches are considered to be “shared.” Other sections of this specification place certain constraints on the operation of shared caches in order to prevent loss of privacy or failure of access controls.

## 13.8 Errors or Incomplete Response Cache Behavior

A cache that receives an incomplete response (for example, with fewer bytes of data than specified in a `Content-Length` header) MAY store the response. However, the cache MUST treat this as a partial response. Partial responses MAY be combined as described in section 13.5.4; the result might be a full response or might still be partial. A cache MUST NOT return a partial response to a client without explicitly marking it as such, using the 206 (Partial Content) status code. A cache MUST NOT return a partial response using a status code of 200 (OK).

If a cache receives a 5xx response while attempting to revalidate an entry, it MAY either forward this response to the requesting client, or act as if the server failed to respond. In the latter case, it MAY return a previously received response unless the cached entry includes the “`must-revalidate`” cache-control directive (see section 14.9).

## 13.9 Side Effects of GET and HEAD

Unless the origin server explicitly prohibits the caching of their responses, the application of GET and HEAD methods to any resources SHOULD NOT have side effects that would lead to erroneous behavior if these responses are taken from a cache. They MAY still have side effects, but a cache is not required to consider such side effects in its caching decisions. Caches are always expected to observe an origin server’s explicit restrictions on caching.

We note one exception to this rule: since some applications have traditionally used GETs and HEADs with query URLs (those containing a “?” in the `rel_path` part) to perform operations with significant side effects, caches MUST NOT treat responses to such URIs as fresh unless the server provides an explicit expiration time. This specifically means that responses from HTTP/1.0 servers for such URIs SHOULD NOT be taken from a cache. See section 9.1.1 for related information.

## 13.10 Invalidation After Updates or Deletions

The effect of certain methods performed on a resource at the origin server might cause one or more existing cache entries to become non-transparently invalid. That is, although they might continue to be “fresh,” they do not accurately reflect what the origin server would return for a new request on that resource.

There is no way for the HTTP protocol to guarantee that all such cache entries are marked invalid. For example, the request that caused the change at the origin server might not have gone through the proxy where a cache entry is stored. However, several rules help reduce the likelihood of erroneous behavior.

In this section, the phrase “invalidate an entity” means that the cache will either remove all instances of that entity from its storage, or will mark these as “invalid” and in need of a mandatory revalidation before they can be returned in response to a subsequent request.

Some HTTP methods MUST cause a cache to invalidate an entity. This is either the entity referred to by the `Request-URI`, or by the `Location` or `Content-Location` headers (if present). These methods are:

- PUT
- DELETE
- POST

In order to prevent denial of service attacks, an invalidation based on the URI in a `Location` or `Content-Location` header MUST only be performed if the host part is the same as in the `Request-URI`.

A cache that passes through requests for methods it does not understand SHOULD invalidate any entities referred to by the `Request-URI`.

## 13.11 Write-Through Mandatory

All methods that might be expected to cause modifications to the origin server’s resources MUST be written through to the origin server. This currently includes all methods except for GET and HEAD. A cache MUST NOT reply to such a request from a client before having transmitted the request to the inbound server, and having received a

corresponding response from the inbound server. This does not prevent a proxy cache from sending a 100 (Continue) response before the inbound server has sent its final reply.

The alternative (known as “write-back” or “copy-back” caching) is not allowed in HTTP/1.1, due to the difficulty of providing consistent updates and the problems arising from server, cache, or network failure prior to write-back.

## 13.12 Cache Replacement

If a new cacheable (see sections 14.9.2, 13.2.5, 13.2.6 and 13.8) response is received from a resource while any existing responses for the same resource are cached, the cache **SHOULD** use the new response to reply to the current request. It **MAY** insert it into cache storage and **MAY**, if it meets all other requirements, use it to respond to any future requests that would previously have caused the old response to be returned. If it inserts the new response into cache storage the rules in section 13.5.3 apply.

Note: a new response that has an older `Date` header value than existing cached responses is not cacheable.

## 13.13 History Lists

User agents often have history mechanisms, such as “Back” buttons and history lists, which can be used to redisplay an entity retrieved earlier in a session.

History mechanisms and caches are different. In particular history mechanisms **SHOULD NOT** try to show a semantically transparent view of the current state of a resource. Rather, a history mechanism is meant to show exactly what the user saw at the time when the resource was retrieved.

By default, an expiration time does not apply to history mechanisms. If the entity is still in storage, a history mechanism **SHOULD** display it even if the entity has expired, unless the user has specifically configured the agent to refresh expired history documents.

This is not to be construed to prohibit the history mechanism from telling the user that a view might be stale.

Note: if history list mechanisms unnecessarily prevent users from viewing stale resources, this will tend to force service authors to avoid using HTTP expiration controls and cache controls when they would otherwise like to. Service authors may consider it important that users not be presented with error messages or warning messages when they use navigation controls (such as BACK) to view previously fetched resources. Even though sometimes such resources ought not to be cached, or ought to expire quickly, user interface considerations may force service authors to resort to other means of preventing caching (e.g. “once-only” URLs) in order not to suffer the effects of improperly functioning history mechanisms.

# 14 Header Field Definitions

This section defines the syntax and semantics of all standard HTTP/1.1 header fields. For entity-header fields, both *sender* and *recipient* refer to either the client or the server, depending on who sends and who receives the entity.

## 14.1 Accept

The `Accept` request-header field can be used to specify certain media types which are acceptable for the response. `Accept` headers can be used to indicate that the request is specifically limited to a small set of desired types, as in the case of a request for an in-line image.

```
Accept          = "Accept" ":"
                  #( media-range [ accept-params ] )

media-range     = ( "*"/*"
                  | ( type "/" "*" )
                  | ( type "/" subtype )
                  ) *( ";" parameter )

accept-params   = ";" "q" "=" qvalue *( accept-extension )
accept-extension = ";" token [ "=" ( token | quoted-string ) ]
```

The asterisk "\*" character is used to group media types into ranges, with "\*/\*" indicating all media types and "type/\*" indicating all subtypes of that type. The `media-range` MAY include media type parameters that are applicable to that range.

Each `media-range` MAY be followed by one or more `accept-params`, beginning with the "q" parameter for indicating a relative quality factor. The first "q" parameter (if any) separates the `media-range` parameter(s) from the `accept-params`. Quality factors allow the user or user agent to indicate the relative degree of preference for that `media-range`, using the `qvalue` scale from 0 to 1 (section 3.9). The default value is `q=1`.

Note: Use of the "q" parameter name to separate media type parameters from `Accept` extension parameters is due to historical practice. Although this prevents any media type parameter named "q" from being used with a media range, such an event is believed to be unlikely given the lack of any "q" parameters in the IANA media type registry and the rare usage of any media type parameters in `Accept`. Future media types are discouraged from registering any parameter named "q".

The example

```
Accept: audio/*; q=0.2, audio/basic
```

SHOULD be interpreted as "I prefer audio/basic, but send me any audio type if it is the best available after an 80% mark-down in quality."

If no `Accept` header field is present, then it is assumed that the client accepts all media types. If an `Accept` header field is present, and if the server cannot send a response which is acceptable according to the combined `Accept` field value, then the server SHOULD send a 406 (not acceptable) response.

A more elaborate example is

```
Accept: text/plain; q=0.5, text/html,
       text/x-dvi; q=0.8, text/x-c
```

Verbally, this would be interpreted as "text/html and text/x-c are the preferred media types, but if they do not exist, then send the text/x-dvi entity, and if that does not exist, send the text/plain entity."

Media ranges can be overridden by more specific media ranges or specific media types. If more than one media range applies to a given type, the most specific reference has precedence. For example,

```
Accept: text/*, text/html, text/html;level=1, */*
```

have the following precedence:

- 1) text/html;level=1
- 2) text/html
- 3) text/\*
- 4) \*/\*

The media type quality factor associated with a given type is determined by finding the media range with the highest precedence which matches that type. For example,

```
Accept: text/*;q=0.3, text/html;q=0.7, text/html;level=1,
       text/html;level=2;q=0.4, */*;q=0.5
```

would cause the following values to be associated:

```
text/html;level=1      = 1
text/html              = 0.7
text/plain             = 0.3
image/jpeg             = 0.5
text/html;level=2     = 0.4
text/html;level=3     = 0.7
```

Note: A user agent might be provided with a default set of quality values for certain media ranges.

However, unless the user agent is a closed system which cannot interact with other rendering agents, this default set ought to be configurable by the user.

## 14.2 Accept-Charset

The `Accept-Charset` request-header field can be used to indicate what character sets are acceptable for the response. This field allows clients capable of understanding more comprehensive or special-purpose character sets to signal that capability to a server which is capable of representing documents in those character sets.

```
Accept-Charset = "Accept-Charset" ":"
                1#( ( charset | "*" ) [ ";" "q" "=" qvalue ] )
```

Character set values are described in section 3.4. Each charset MAY be given an associated quality value which represents the user's preference for that charset. The default value is `q=1`. An example is

```
Accept-Charset: iso-8859-5, unicode-1-1;q=0.8
```

The special value `"*"`, if present in the `Accept-Charset` field, matches every character set (including ISO-8859-1) which is not mentioned elsewhere in the `Accept-Charset` field. If no `"*"` is present in an `Accept-Charset` field, then all character sets not explicitly mentioned get a quality value of 0, except for ISO-8859-1, which gets a quality value of 1 if not explicitly mentioned.

If no `Accept-Charset` header is present, the default is that any character set is acceptable. If an `Accept-Charset` header is present, and if the server cannot send a response which is acceptable according to the `Accept-Charset` header, then the server SHOULD send an error response with the 406 (not acceptable) status code, though the sending of an unacceptable response is also allowed.

## 14.3 Accept-Encoding

The `Accept-Encoding` request-header field is similar to `Accept`, but restricts the content-codings (section 3.5) that are acceptable in the response.

```
Accept-Encoding = "Accept-Encoding" ":"
                1#( codings [ ";" "q" "=" qvalue ] )
codings         = ( content-coding | "*" )
```

Examples of its use are:

```
Accept-Encoding: compress, gzip
Accept-Encoding:
Accept-Encoding: *
Accept-Encoding: compress;q=0.5, gzip;q=1.0
Accept-Encoding: gzip;q=1.0, identity; q=0.5, *;q=0
```

A server tests whether a content-coding is acceptable, according to an `Accept-Encoding` field, using these rules:

1. If the content-coding is one of the content-codings listed in the `Accept-Encoding` field, then it is acceptable, unless it is accompanied by a `qvalue` of 0. (As defined in section 3.9, a `qvalue` of 0 means "not acceptable.")
2. The special `"*"` symbol in an `Accept-Encoding` field matches any available content-coding not explicitly listed in the header field.
3. If multiple content-codings are acceptable, then the acceptable content-coding with the highest non-zero `qvalue` is preferred.
4. The `"identity"` content-coding is always acceptable, unless specifically refused because the `Accept-Encoding` field includes `"identity;q=0"`, or because the field includes `"*;q=0"` and does not explicitly include the `"identity"` content-coding. If the `Accept-Encoding` field-value is empty, then only the `"identity"` encoding is acceptable.

If an `Accept-Encoding` field is present in a request, and if the server cannot send a response which is acceptable according to the `Accept-Encoding` header, then the server SHOULD send an error response with the 406 (Not Acceptable) status code.

If no `Accept-Encoding` field is present in a request, the server MAY assume that the client will accept any content coding. In this case, if “identity” is one of the available content-codings, then the server SHOULD use the “identity” content-coding, unless it has additional information that a different content-coding is meaningful to the client.

Note: If the request does not include an `Accept-Encoding` field, and if the “identity” content-coding is unavailable, then content-codings commonly understood by HTTP/1.0 clients (i.e., “gzip” and “compress”) are preferred; some older clients improperly display messages sent with other content-codings. The server might also make this decision based on information about the particular user-agent or client.

Note: Most HTTP/1.0 applications do not recognize or obey qvalues associated with content-codings. This means that qvalues will not work and are not permitted with `x-gzip` or `x-compress`.

## 14.4 Accept-Language

The `Accept-Language` request-header field is similar to `Accept`, but restricts the set of natural languages that are preferred as a response to the request. Language tags are defined in section 3.10.

```
Accept-Language = "Accept-Language" ":"
                  1#( language-range [ ";" "q" "=" qvalue ] )
language-range = ( ( 1*8ALPHA *( "-" 1*8ALPHA ) ) | "*" )
```

Each language-range MAY be given an associated quality value which represents an estimate of the user’s preference for the languages specified by that range. The quality value defaults to “q=1”. For example,

```
Accept-Language: da, en-gb;q=0.8, en;q=0.7
```

would mean: “I prefer Danish, but will accept British English and other types of English.” A language-range matches a language-tag if it exactly equals the tag, or if it exactly equals a prefix of the tag such that the first tag character following the prefix is “-”. The special range “\*”, if present in the `Accept-Language` field, matches every tag not matched by any other range present in the `Accept-Language` field.

Note: This use of a prefix matching rule does not imply that language tags are assigned to languages in such a way that it is always true that if a user understands a language with a certain tag, then this user will also understand all languages with tags for which this tag is a prefix. The prefix rule simply allows the use of prefix tags if this is the case.

The language quality factor assigned to a language-tag by the `Accept-Language` field is the quality value of the longest language-range in the field that matches the language-tag. If no language-range in the field matches the tag, the language quality factor assigned is 0. If no `Accept-Language` header is present in the request, the server SHOULD assume that all languages are equally acceptable. If an `Accept-Language` header is present, then all languages which are assigned a quality factor greater than 0 are acceptable.

It might be contrary to the privacy expectations of the user to send an `Accept-Language` header with the complete linguistic preferences of the user in every request. For a discussion of this issue, see section 15.1.4.

As intelligibility is highly dependent on the individual user, it is recommended that client applications make the choice of linguistic preference available to the user. If the choice is not made available, then the `Accept-Language` header field MUST NOT be given in the request.

Note: When making the choice of linguistic preference available to the user, we remind implementors of the fact that users are not familiar with the details of language matching as described above, and should provide appropriate guidance. As an example, users might assume that on selecting “en-gb”, they will be served any kind of English document if British English is not available. A user agent might suggest in such a case to add “en” to get the best matching behavior.

## 14.5 Accept-Ranges

The `Accept-Range` response-header field allows the server to indicate its acceptance of range requests for a resource:

```
Accept-Range      = "Accept-Range" ":" acceptable-ranges
acceptable-ranges = 1#range-unit | "none"
```

Origin servers that accept byte-range requests **MAY** send

```
Accept-Range: bytes
```

but are not required to do so. Clients **MAY** generate byte-range requests without having received this header for the resource involved. Range units are defined in section 3.12.

Servers that do not accept any kind of range request for a resource **MAY** send

```
Accept-Range: none
```

to advise the client not to attempt a range request.

## 14.6 Age

The `Age` response-header field conveys the sender's estimate of the amount of time since the response (or its revalidation) was generated at the origin server. A cached response is "fresh" if its age does not exceed its freshness lifetime. Age values are calculated as specified in section 13.2.3.

```
Age = "Age" ":" age-value
age-value = delta-seconds
```

Age values are non-negative decimal integers, representing time in seconds.

If a cache receives a value larger than the largest positive integer it can represent, or if any of its age calculations overflows, it **MUST** transmit an `Age` header with a value of 2147483648 ( $2^{31}$ ). An HTTP/1.1 server that includes a cache **MUST** include an `Age` header field in every response generated from its own cache. Caches **SHOULD** use an arithmetic type of at least 31 bits of range.

## 14.7 Allow

The `Allow` entity-header field lists the set of methods supported by the resource identified by the `Request-URI`. The purpose of this field is strictly to inform the recipient of valid methods associated with the resource. An `Allow` header field **MUST** be present in a 405 (Method Not Allowed) response.

```
Allow = "Allow" ":" #Method
```

Example of use:

```
Allow: GET, HEAD, PUT
```

This field cannot prevent a client from trying other methods. However, the indications given by the `Allow` header field value **SHOULD** be followed. The actual set of allowed methods is defined by the origin server at the time of each request.

The `Allow` header field **MAY** be provided with a `PUT` request to recommend the methods to be supported by the new or modified resource. The server is not required to support these methods and **SHOULD** include an `Allow` header in the response giving the actual supported methods.

A proxy **MUST NOT** modify the `Allow` header field even if it does not understand all the methods specified, since the user agent might have other means of communicating with the origin server.

## 14.8 Authorization

A user agent that wishes to authenticate itself with a server--usually, but not necessarily, after receiving a 401 response--does so by including an `Authorization` request-header field with the request. The `Authorization` field value consists of `credentials` containing the authentication information of the user agent for the realm of the resource being requested.

`Authorization` = "Authorization" ":" credentials

HTTP access authentication is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. If a request is authenticated and a `realm` specified, the same `credentials` SHOULD be valid for all other requests within this `realm` (assuming that the authentication scheme itself does not require otherwise, such as credentials that vary according to a challenge value or using synchronized clocks).

When a shared cache (see section 13.7) receives a request containing an `Authorization` field, it MUST NOT return the corresponding response as a reply to any other request, unless one of the following specific exceptions holds:

1. If the response includes the "s-maxage" cache-control directive, the cache MAY use that response in replying to a subsequent request. But (if the specified maximum age has passed) a proxy cache MUST first revalidate it with the origin server, using the request-headers from the new request to allow the origin server to authenticate the new request. (This is the defined behavior for s-maxage.) If the response includes "s-maxage=0", the proxy MUST always revalidate it before re-using it.
2. If the response includes the "must-revalidate" cache-control directive, the cache MAY use that response in replying to a subsequent request. But if the response is stale, all caches MUST first revalidate it with the origin server, using the request-headers from the new request to allow the origin server to authenticate the new request.
3. If the response includes the "public" cache-control directive, it MAY be returned in reply to any subsequent request.

## 14.9 Cache-Control

The `Cache-Control` general-header field is used to specify directives that MUST be obeyed by all caching mechanisms along the request/response chain. The directives specify behavior intended to prevent caches from adversely interfering with the request or response. These directives typically override the default caching algorithms. Cache directives are unidirectional in that the presence of a directive in a request does not imply that the same directive is to be given in the response.

Note that HTTP/1.0 caches might not implement `Cache-Control` and might only implement `Pragma: no-cache` (see section 14.32).

Cache directives MUST be passed through by a proxy or gateway application, regardless of their significance to that application, since the directives might be applicable to all recipients along the request/response chain. It is not possible to specify a cache-directive for a specific cache.

```
Cache-Control = "Cache-Control" ":" 1#cache-directive
cache-directive = cache-request-directive
                  | cache-response-directive
cache-request-directive =
    "no-cache"                ; Section 14.9.1
    | "no-store"              ; Section 14.9.2
    | "max-age" "=" delta-seconds ; Section 14.9.3, 14.9.4
    | "max-stale" [ "=" delta-seconds ] ; Section 14.9.3
    | "min-fresh" "=" delta-seconds ; Section 14.9.3
    | "no-transform"          ; Section 14.9.5
    | "only-if-cached"        ; Section 14.9.4
    | cache-extension         ; Section 14.9.6
cache-response-directive =
    "public"                  ; Section 14.9.1
    | "private" [ "=" <"> 1#field-name <"> ] ; Section 14.9.1
    | "no-cache" [ "=" <"> 1#field-name <"> ] ; Section 14.9.1
    | "no-store"              ; Section 14.9.2
    | "no-transform"          ; Section 14.9.5
    | "must-revalidate"       ; Section 14.9.4
    | "proxy-revalidate"      ; Section 14.9.4
```



```

    | "max-age" "=" delta-seconds           ; Section 14.9.3
    | "s-maxage" "=" delta-seconds          ; Section 14.9.3
    | cache-extension                       ; Section 14.9.6
cache-extension = token [ "=" ( token | quoted-string ) ]

```

When a directive appears without any `1#field-name` parameter, the directive applies to the entire request or response. When such a directive appears with a `1#field-name` parameter, it applies only to the named field or fields, and not to the rest of the request or response. This mechanism supports extensibility; implementations of future versions of the HTTP protocol might apply these directives to header fields not defined in HTTP/1.1.

The cache-control directives can be broken down into these general categories:

- Restrictions on what are cacheable; these may only be imposed by the origin server.
- Restrictions on what may be stored by a cache; these may be imposed by either the origin server or the user agent.
- Modifications of the basic expiration mechanism; these may be imposed by either the origin server or the user agent.
- Controls over cache revalidation and reload; these may only be imposed by a user agent.
- Control over transformation of entities.
- Extensions to the caching system.

#### 14.9.1 What is Cacheable

By default, a response is cacheable if the requirements of the request method, request header fields, and the response status indicate that it is cacheable. Section 13.4 summarizes these defaults for cacheability. The following `Cache-Control` response directives allow an origin server to override the default cacheability of a response:

`public`

Indicates that the response **MAY** be cached by any cache, even if it would normally be non-cacheable or cacheable only within a non-shared cache. (See also `Authorization`, section 14.8, for additional details.)

`private`

Indicates that all or part of the response message is intended for a single user and **MUST NOT** be cached by a shared cache. This allows an origin server to state that the specified parts of the response are intended for only one user and are not a valid response for requests by other users. A private (non-shared) cache **MAY** cache the response.

Note: This usage of the word `private` only controls where the response may be cached, and cannot ensure the privacy of the message content.

`no-cache`

If the `no-cache` directive does not specify a field-name, then a cache **MUST NOT** use the response to satisfy a subsequent request without successful revalidation with the origin server. This allows an origin server to prevent caching even by caches that have been configured to return stale responses to client requests.

If the `no-cache` directive does specify one or more field-names, then a cache **MAY** use the response to satisfy a subsequent request, subject to any other restrictions on caching. However, the specified field-name(s) **MUST NOT** be sent in the response to a subsequent request without successful revalidation with the origin server. This allows an origin server to prevent the re-use of certain header fields in a response, while still allowing caching of the rest of the response.

Note: Most HTTP/1.0 caches will not recognize or obey this directive.

### 14.9.2 What May be Stored by Caches

#### `no-store`

The purpose of the `no-store` directive is to prevent the inadvertent release or retention of sensitive information (for example, on backup tapes). The `no-store` directive applies to the entire message, and MAY be sent either in a response or in a request. If sent in a request, a cache MUST NOT store any part of either this request or any response to it. If sent in a response, a cache MUST NOT store any part of either this response or the request that elicited it. This directive applies to both non-shared and shared caches. "MUST NOT store" in this context means that the cache MUST NOT intentionally store the information in non-volatile storage, and MUST make a best-effort attempt to remove the information from volatile storage as promptly as possible after forwarding it.

Even when this directive is associated with a response, users might explicitly store such a response outside of the caching system (e.g., with a "Save As" dialog). History buffers MAY store such responses as part of their normal operation.

The purpose of this directive is to meet the stated requirements of certain users and service authors who are concerned about accidental releases of information via unanticipated accesses to cache data structures. While the use of this directive might improve privacy in some cases, we caution that it is NOT in any way a reliable or sufficient mechanism for ensuring privacy. In particular, malicious or compromised caches might not recognize or obey this directive, and communications networks might be vulnerable to eavesdropping.

### 14.9.3 Modifications of the Basic Expiration Mechanism

The expiration time of an entity MAY be specified by the origin server using the `Expires` header (see section 14.21). Alternatively, it MAY be specified using the `max-age` directive in a response. When the `max-age` cache-control directive is present in a cached response, the response is stale if its current age is greater than the age value given (in seconds) at the time of a new request for that resource. The `max-age` directive on a response implies that the response is cacheable (i.e., "public") unless some other, more restrictive cache directive is also present.

If a response includes both an `Expires` header and a `max-age` directive, the `max-age` directive overrides the `Expires` header, even if the `Expires` header is more restrictive. This rule allows an origin server to provide, for a given response, a longer expiration time to an HTTP/1.1 (or later) cache than to an HTTP/1.0 cache. This might be useful if certain HTTP/1.0 caches improperly calculate ages or expiration times, perhaps due to desynchronized clocks.

Many HTTP/1.0 cache implementations will treat an `Expires` value that is less than or equal to the response `Date` value as being equivalent to the `Cache-Control` response directive "no-cache". If an HTTP/1.1 cache receives such a response, and the response does not include a `Cache-Control` header field, it SHOULD consider the response to be non-cacheable in order to retain compatibility with HTTP/1.0 servers.

Note: An origin server might wish to use a relatively new HTTP cache control feature, such as the "private" directive, on a network including older caches that do not understand that feature. The origin server will need to combine the new feature with an `Expires` field whose value is less than or equal to the `Date` value. This will prevent older caches from improperly caching the response.

#### `s-maxage`

If a response includes an `s-maxage` directive, then for a shared cache (but not for a private cache), the maximum age specified by this directive overrides the maximum age specified by either the `max-age` directive or the `Expires` header. The `s-maxage` directive also implies the semantics of the `proxy-revalidate` directive (see section 14.9.4), i.e., that the shared cache must not use the entry after it becomes stale to respond to a subsequent request without first revalidating it with the origin server. The `s-maxage` directive is always ignored by a private cache.

Note that most older caches, not compliant with this specification, do not implement any cache-control directives. An origin server wishing to use a cache-control directive that restricts, but does not prevent, caching by an HTTP/1.1-

compliant cache MAY exploit the requirement that the `max-age` directive overrides the `Expires` header, and the fact that pre-HTTP/1.1-compliant caches do not observe the `max-age` directive.

Other directives allow a user agent to modify the basic expiration mechanism. These directives MAY be specified on a request:

#### `max-age`

Indicates that the client is willing to accept a response whose age is no greater than the specified time in seconds. Unless `max-stale` directive is also included, the client is not willing to accept a stale response.

#### `min-fresh`

Indicates that the client is willing to accept a response whose freshness lifetime is no less than its current age plus the specified time in seconds. That is, the client wants a response that will still be fresh for at least the specified number of seconds.

#### `max-stale`

Indicates that the client is willing to accept a response that has exceeded its expiration time. If `max-stale` is assigned a value, then the client is willing to accept a response that has exceeded its expiration time by no more than the specified number of seconds. If no value is assigned to `max-stale`, then the client is willing to accept a stale response of any age.

If a cache returns a stale response, either because of a `max-stale` directive on a request, or because the cache is configured to override the expiration time of a response, the cache MUST attach a `Warning` header to the stale response, using Warning 110 (Response is stale).

A cache MAY be configured to return stale responses without validation, but only if this does not conflict with any "MUST"-level requirements concerning cache validation (e.g., a "`must-revalidate`" `cache-control` directive).

If both the new request and the cached entry include "`max-age`" directives, then the lesser of the two values is used for determining the freshness of the cached entry for that request.

### 14.9.4 Cache Revalidation and Reload Controls

Sometimes a user agent might want or need to insist that a cache revalidate its cache entry with the origin server (and not just with the next cache along the path to the origin server), or to reload its cache entry from the origin server. End-to-end revalidation might be necessary if either the cache or the origin server has overestimated the expiration time of the cached response. End-to-end reload may be necessary if the cache entry has become corrupted for some reason.

End-to-end revalidation may be requested either when the client does not have its own local cached copy, in which case we call it "unspecified end-to-end revalidation", or when the client does have a local cached copy, in which case we call it "specific end-to-end revalidation."

The client can specify these three kinds of action using `Cache-Control` request directives:

#### End-to-end reload

The request includes a "`no-cache`" `cache-control` directive or, for compatibility with HTTP/1.0 clients, "`Pragma: no-cache`". Field names MUST NOT be included with the `no-cache` directive in a request. The server MUST NOT use a cached copy when responding to such a request.

#### Specific end-to-end revalidation

The request includes a "`max-age=0`" `cache-control` directive, which forces each cache along the path to the origin server to revalidate its own entry, if any, with the next cache or server. The initial request includes a cache-validating conditional with the client's current validator.

#### Unspecified end-to-end revalidation

The request includes "`max-age=0`" `cache-control` directive, which forces each cache along the path to the origin server to revalidate its own entry, if any, with the next cache or server. The initial request does not include

a cache-validating conditional; the first cache along the path (if any) that holds a cache entry for this resource includes a cache-validating conditional with its current validator.

#### max-age

When an intermediate cache is forced, by means of a `max-age=0` directive, to revalidate its own cache entry, and the client has supplied its own validator in the request, the supplied validator might differ from the validator currently stored with the cache entry. In this case, the cache *MAY* use either validator in making its own request without affecting semantic transparency.

However, the choice of validator might affect performance. The best approach is for the intermediate cache to use its own validator when making its request. If the server replies with 304 (Not Modified), then the cache can return its now validated copy to the client with a 200 (OK) response. If the server replies with a new entity and cache validator, however, the intermediate cache can compare the returned validator with the one provided in the client's request, using the strong comparison function. If the client's validator is equal to the origin server's, then the intermediate cache simply returns 304 (Not Modified). Otherwise, it returns the new entity with a 200 (OK) response.

If a request includes the `no-cache` directive, it *SHOULD NOT* include `min-fresh`, `max-stale`, or `max-age`.

#### only-if-cached

In some cases, such as times of extremely poor network connectivity, a client may want a cache to return only those responses that it currently has stored, and not to reload or revalidate with the origin server. To do this, the client may include the `only-if-cached` directive in a request. If it receives this directive, a cache *SHOULD* either respond using a cached entry that is consistent with the other constraints of the request, or respond with a 504 (Gateway Timeout) status. However, if a group of caches is being operated as a unified system with good internal connectivity, such a request *MAY* be forwarded within that group of caches.

#### must-revalidate

Because a cache *MAY* be configured to ignore a server's specified expiration time, and because a client request *MAY* include a `max-stale` directive (which has a similar effect), the protocol also includes a mechanism for the origin server to require revalidation of a cache entry on any subsequent use. When the `must-revalidate` directive is present in a response received by a cache, that cache *MUST NOT* use the entry after it becomes stale to respond to a subsequent request without first revalidating it with the origin server. (I.e., the cache *MUST* do an end-to-end revalidation every time, if, based solely on the origin server's `Expires` or `max-age` value, the cached response is stale.)

The `must-revalidate` directive is necessary to support reliable operation for certain protocol features. In all circumstances an HTTP/1.1 cache *MUST* obey the `must-revalidate` directive; in particular, if the cache cannot reach the origin server for any reason, it *MUST* generate a 504 (Gateway Timeout) response.

Servers *SHOULD* send the `must-revalidate` directive if and only if failure to revalidate a request on the entity could result in incorrect operation, such as a silently unexecuted financial transaction. Recipients *MUST NOT* take any automated action that violates this directive, and *MUST NOT* automatically provide an unvalidated copy of the entity if revalidation fails.

Although this is not recommended, user agents operating under severe connectivity constraints *MAY* violate this directive but, if so, *MUST* explicitly warn the user that an unvalidated response has been provided. The warning *MUST* be provided on each unvalidated access, and *SHOULD* require explicit user confirmation.

#### proxy-revalidate

The `proxy-revalidate` directive has the same meaning as the `must-revalidate` directive, except that it does not apply to non-shared user agent caches. It can be used on a response to an authenticated request to permit the user's cache to store and later return the response without needing to revalidate it (since it has already been authenticated once by that user), while still requiring proxies that service many users to revalidate each time (in order to make sure that each user has been authenticated). Note that such authenticated responses also need the `public` cache control directive in order to allow them to be cached at all.

### 14.9.5 No-Transform Directive

`no-transform`

Implementors of intermediate caches (proxies) have found it useful to convert the media type of certain entity bodies. A non-transparent proxy might, for example, convert between image formats in order to save cache space or to reduce the amount of traffic on a slow link.

Serious operational problems occur, however, when these transformations are applied to entity bodies intended for certain kinds of applications. For example, applications for medical imaging, scientific data analysis and those using end-to-end authentication, all depend on receiving an entity body that is bit for bit identical to the original entity-body.

Therefore, if a message includes the `no-transform` directive, an intermediate cache or proxy **MUST NOT** change those headers that are listed in section 13.5.2 as being subject to the `no-transform` directive. This implies that the cache or proxy **MUST NOT** change any aspect of the entity-body that is specified by these headers, including the value of the entity-body itself.

### 14.9.6 Cache Control Extensions

The `Cache-Control` header field can be extended through the use of one or more cache-extension tokens, each with an optional assigned value. Informational extensions (those which do not require a change in cache behavior) **MAY** be added without changing the semantics of other directives. Behavioral extensions are designed to work by acting as modifiers to the existing base of cache directives. Both the new directive and the standard directive are supplied, such that applications which do not understand the new directive will default to the behavior specified by the standard directive, and those that understand the new directive will recognize it as modifying the requirements associated with the standard directive. In this way, extensions to the cache-control directives can be made without requiring changes to the base protocol.

This extension mechanism depends on an HTTP cache obeying all of the cache-control directives defined for its native HTTP-version, obeying certain extensions, and ignoring all directives that it does not understand.

For example, consider a hypothetical new response directive called `community` which acts as a modifier to the `private` directive. We define this new directive to mean that, in addition to any non-shared cache, any cache which is shared only by members of the community named within its value may cache the response. An origin server wishing to allow the UCI community to use an otherwise private response in their shared cache(s) could do so by including

```
Cache-Control: private, community="UCI"
```

A cache seeing this header field will act correctly even if the cache does not understand the `community` cache-extension, since it will also see and understand the `private` directive and thus default to the safe behavior.

Unrecognized cache-directives **MUST** be ignored; it is assumed that any cache-directive likely to be unrecognized by an HTTP/1.1 cache will be combined with standard directives (or the response's default cacheability) such that the cache behavior will remain minimally correct even if the cache does not understand the extension(s).

## 14.10 Connection

The `Connection` general-header field allows the sender to specify options that are desired for that particular connection and **MUST NOT** be communicated by proxies over further connections.

The `Connection` header has the following grammar:

```
Connection = "Connection" ":" 1#(connection-token)
connection-token = token
```

HTTP/1.1 proxies **MUST** parse the `Connection` header field before a message is forwarded and, for each connection-token in this field, remove any header field(s) from the message with the same name as the connection-token. `Connection` options are signaled by the presence of a connection-token in the `Connection` header field,

not by any corresponding additional header field(s), since the additional header field may not be sent if there are no parameters associated with that connection option.

Message headers listed in the `Connection` header **MUST NOT** include end-to-end headers, such as `Cache-Control`.

HTTP/1.1 defines the “close” connection option for the sender to signal that the connection will be closed after completion of the response. For example,

```
Connection: close
```

in either the request or the response header fields indicates that the connection **SHOULD NOT** be considered ‘persistent’ (section 8.1) after the current request/response is complete.

HTTP/1.1 applications that do not support persistent connections **MUST** include the “close” connection option in every message.

A system receiving an HTTP/1.0 (or lower-version) message that includes a `Connection` header **MUST**, for each connection-token in this field, remove and ignore any header field(s) from the message with the same name as the connection-token. This protects against mistaken forwarding of such header fields by pre-HTTP/1.1 proxies. See section 19.6.2.

## 14.11 Content-Encoding

The `Content-Encoding` entity-header field is used as a modifier to the media-type. When present, its value indicates what additional content codings have been applied to the entity-body, and thus what decoding mechanisms must be applied in order to obtain the media-type referenced by the `Content-Type` header field. `Content-Encoding` is primarily used to allow a document to be compressed without losing the identity of its underlying media type.

```
Content-Encoding = "Content-Encoding" ":" 1#content-coding
```

Content codings are defined in section 3.5. An example of its use is

```
Content-Encoding: gzip
```

The content-coding is a characteristic of the entity identified by the `Request-URI`. Typically, the entity-body is stored with this encoding and is only decoded before rendering or analogous usage. However, a non-transparent proxy **MAY** modify the content-coding if the new coding is known to be acceptable to the recipient, unless the “no-transform” cache-control directive is present in the message.

If the content-coding of an entity is not “identity”, then the response **MUST** include a `Content-Encoding` entity-header (section 14.11) that lists the non-identity content-coding(s) used.

If the content-coding of an entity in a request message is not acceptable to the origin server, the server **SHOULD** respond with a status code of 415 (Unsupported Media Type).

If multiple encodings have been applied to an entity, the content codings **MUST** be listed in the order in which they were applied. Additional information about the encoding parameters **MAY** be provided by other entity-header fields not defined by this specification.

## 14.12 Content-Language

The `Content-Language` entity-header field describes the natural language(s) of the intended audience for the enclosed entity. Note that this might not be equivalent to all the languages used within the entity-body.

```
Content-Language = "Content-Language" ":" 1#language-tag
```

Language tags are defined in section 3.10. The primary purpose of `Content-Language` is to allow a user to identify and differentiate entities according to the user’s own preferred language. Thus, if the body content is intended only for a Danish-literate audience, the appropriate field is

```
Content-Language: da
```

If no `Content-Language` is specified, the default is that the content is intended for all language audiences. This might mean that the sender does not consider it to be specific to any natural language, or that the sender does not know for which language it is intended.

Multiple languages MAY be listed for content that is intended for multiple audiences. For example, a rendition of the "Treaty of Waitangi," presented simultaneously in the original Maori and English versions, would call for

```
Content-Language: mi, en
```

However, just because multiple languages are present within an entity does not mean that it is intended for multiple linguistic audiences. An example would be a beginner's language primer, such as "A First Lesson in Latin," which is clearly intended to be used by an English-literate audience. In this case, the `Content-Language` would properly only include "en".

`Content-Language` MAY be applied to any media type -- it is not limited to textual documents.

### 14.13 Content-Length

The `Content-Length` entity-header field indicates the size of the entity-body, in decimal number of OCTETs, sent to the recipient or, in the case of the HEAD method, the size of the entity-body that would have been sent had the request been a GET.

```
Content-Length = "Content-Length" ":" 1*DIGIT
```

An example is

```
Content-Length: 3495
```

Applications SHOULD use this field to indicate the transfer-length of the message-body, unless this is prohibited by the rules in section 4.4.

Any `Content-Length` greater than or equal to zero is a valid value. Section 4.4 describes how to determine the length of a message-body if a `Content-Length` is not given.

Note that the meaning of this field is significantly different from the corresponding definition in MIME, where it is an optional field used within the "message/external-body" content-type. In HTTP, it SHOULD be sent whenever the message's length can be determined prior to being transferred, unless this is prohibited by the rules in section 4.4.

### 14.14 Content-Location

The `Content-Location` entity-header field MAY be used to supply the resource location for the entity enclosed in the message when that entity is accessible from a location separate from the requested resource's URI. A server SHOULD provide a `Content-Location` for the variant corresponding to the response entity; especially in the case where a resource has multiple entities associated with it, and those entities actually have separate locations by which they might be individually accessed, the server SHOULD provide a `Content-Location` for the particular variant which is returned.

```
Content-Location = "Content-Location" ":"
                  ( absoluteURI | relativeURI )
```

The value of `Content-Location` also defines the base URI for the entity.

The `Content-Location` value is not a replacement for the original requested URI; it is only a statement of the location of the resource corresponding to this particular entity at the time of the request. Future requests MAY specify the `Content-Location` URI as the request-URI if the desire is to identify the source of that particular entity.

A cache cannot assume that an entity with a `Content-Location` different from the URI used to retrieve it can be used to respond to later requests on that `Content-Location` URI. However, the `Content-Location` can be used to differentiate between multiple entities retrieved from a single requested resource, as described in section 13.6.

If the `Content-Location` is a relative URI, the relative URI is interpreted relative to the `Request-URI`.

The meaning of the `Content-Location` header in PUT or POST requests is undefined; servers are free to ignore it in those cases.

## 14.15 Content-MD5

The `Content-MD5` entity-header field, as defined in RFC 1864 [23], is an MD5 digest of the entity-body for the purpose of providing an end-to-end message integrity check (MIC) of the entity-body. (Note: a MIC is good for detecting accidental modification of the entity-body in transit, but is not proof against malicious attacks.)

```
Content-MD5    = "Content-MD5" ":" md5-digest
md5-digest    = <base64 of 128 bit MD5 digest as per RFC 1864>
```

The `Content-MD5` header field MAY be generated by an origin server or client to function as an integrity check of the entity-body. Only origin servers or clients MAY generate the `Content-MD5` header field; proxies and gateways MUST NOT generate it, as this would defeat its value as an end-to-end integrity check. Any recipient of the entity-body, including gateways and proxies, MAY check that the digest value in this header field matches that of the entity-body as received.

The MD5 digest is computed based on the content of the entity-body, including any content-coding that has been applied, but not including any transfer-encoding applied to the message-body. If the message is received with a transfer-encoding, that encoding MUST be removed prior to checking the `Content-MD5` value against the received entity.

This has the result that the digest is computed on the octets of the entity-body exactly as, and in the order that, they would be sent if no transfer-encoding were being applied.

HTTP extends RFC 1864 to permit the digest to be computed for MIME composite media-types (e.g., `multipart/*` and `message/rfc822`), but this does not change how the digest is computed as defined in the preceding paragraph.

There are several consequences of this. The entity-body for composite types MAY contain many body-parts, each with its own MIME and HTTP headers (including `Content-MD5`, `Content-Transfer-Encoding`, and `Content-Encoding` headers). If a body-part has a `Content-Transfer-Encoding` or `Content-Encoding` header, it is assumed that the content of the body-part has had the encoding applied, and the body-part is included in the `Content-MD5` digest as is -- i.e., after the application. The `Transfer-Encoding` header field is not allowed within body-parts.

Conversion of all line breaks to CRLF MUST NOT be done before computing or checking the digest: the line break convention used in the text actually transmitted MUST be left unaltered when computing the digest.

Note: while the definition of `Content-MD5` is exactly the same for HTTP as in RFC 1864 for MIME entity-bodies, there are several ways in which the application of `Content-MD5` to HTTP entity-bodies differs from its application to MIME entity-bodies. One is that HTTP, unlike MIME, does not use `Content-Transfer-Encoding`, and does use `Transfer-Encoding` and `Content-Encoding`. Another is that HTTP more frequently uses binary content types than MIME, so it is worth noting that, in such cases, the byte order used to compute the digest is the transmission byte order defined for the type. Lastly, HTTP allows transmission of text types with any of several line break conventions and not just the canonical form using CRLF.

## 14.16 Content-Range

The `Content-Range` entity-header is sent with a partial entity-body to specify where in the full entity-body the partial body should be applied. Range units are defined in section 3.12.

```
Content-Range = "Content-Range" ":" content-range-spec
content-range-spec = byte-content-range-spec
byte-content-range-spec = bytes-unit SP
                        byte-range-resp-spec "/"
                        ( instance-length | "*" )

byte-range-resp-spec = (first-byte-pos "-" last-byte-pos)
```



```

instance-length = 1 *DIGIT | "*"

```

The header SHOULD indicate the total length of the full entity-body, unless this length is unknown or difficult to determine. The asterisk "\*" character means that the `instance-length` is unknown at the time when the response was generated.

Unlike `byte-ranges-specifier` values (see section 14.35.1), a `byte-range-resp-spec` MUST only specify one range, and MUST contain absolute byte positions for both the first and last byte of the range.

A `byte-content-range-spec` with a `byte-range-resp-spec` whose `last-byte-pos` value is less than its `first-byte-pos` value, or whose `instance-length` value is less than or equal to its `last-byte-pos` value, is invalid. The recipient of an invalid `byte-content-range-spec` MUST ignore it and any content transferred along with it.

A server sending a response with status code 416 (Requested range not satisfiable) SHOULD include a `Content-Range` field with a `byte-range-resp-spec` of "\*". The `instance-length` specifies the current length of the selected resource. A response with status code 206 (Partial Content) MUST NOT include a `Content-Range` field with a `byte-range-resp-spec` of "\*".

Examples of `byte-content-range-spec` values, assuming that the entity contains a total of 1234 bytes:

- The first 500 bytes:  
bytes 0-499/1234
- The second 500 bytes:  
bytes 500-999/1234
- All except for the first 500 bytes:  
bytes 500-1233/1234
- The last 500 bytes:  
bytes 734-1233/1234

When an HTTP message includes the content of a single range (for example, a response to a request for a single range, or to a request for a set of ranges that overlap without any holes), this content is transmitted with a `Content-Range` header, and a `Content-Length` header showing the number of bytes actually transferred. For example,

```

HTTP/1.1 206 Partial content
Date: Wed, 15 Nov 1995 06:25:24 GMT
Last-Modified: Wed, 15 Nov 1995 04:58:08 GMT
Content-Range: bytes 21010-47021/47022
Content-Length: 26012
Content-Type: image/gif

```

When an HTTP message includes the content of multiple ranges (for example, a response to a request for multiple non-overlapping ranges), these are transmitted as a multipart message. The multipart media type used for this purpose is "multipart/byteranges" as defined in appendix 19.2. See appendix 19.6.3 for a compatibility issue.

A response to a request for a single range MUST NOT be sent using the multipart/byteranges media type. A response to a request for multiple ranges, whose result is a single range, MAY be sent as a multipart/byteranges media type with one part. A client that cannot decode a multipart/byteranges message MUST NOT ask for multiple byte-ranges in a single request.

When a client requests multiple byte-ranges in one request, the server SHOULD return them in the order that they appeared in the request.

If the server ignores a `byte-range-spec` because it is syntactically invalid, the server SHOULD treat the request as if the invalid `Range` header field did not exist. (Normally, this means return a 200 response containing the full entity).

If the server receives a request (other than one including an `If-Range` request-header field) with an unsatisfiable `Range` request-header field (that is, all of whose `byte-range-spec` values have a `first-byte-pos` value

greater than the current length of the selected resource), it SHOULD return a response code of 416 (Requested range not satisfiable) (section 10.4.17).

Note: clients cannot depend on servers to send a 416 (Requested range not satisfiable) response instead of a 200 (OK) response for an unsatisfiable Range request-header, since not all servers implement this request-header.

## 14.17 Content-Type

The `Content-Type` entity-header field indicates the media type of the entity-body sent to the recipient or, in the case of the HEAD method, the media type that would have been sent had the request been a GET.

```
Content-Type = "Content-Type" ":" media-type
```

Media types are defined in section 3.7. An example of the field is

```
Content-Type: text/html; charset=ISO-8859-4
```

Further discussion of methods for identifying the media type of an entity is provided in section 7.2.1.

## 14.18 Date

The `Date` general-header field represents the date and time at which the message was originated, having the same semantics as `orig-date` in RFC 822. The field value is an `HTTP-date`, as described in section 3.3.1; it MUST be sent in RFC 1123 [8]-date format.

```
Date = "Date" ":" HTTP-date
```

An example is

```
Date: Tue, 15 Nov 1994 08:12:31 GMT
```

Origin servers MUST include a `Date` header field in all responses, except in these cases:

1. If the response status code is 100 (Continue) or 101 (Switching Protocols), the response MAY include a `Date` header field, at the server's option.
2. If the response status code conveys a server error, e.g. 500 (Internal Server Error) or 503 (Service Unavailable), and it is inconvenient or impossible to generate a valid `Date`.
3. If the server does not have a clock that can provide a reasonable approximation of the current time, its responses MUST NOT include a `Date` header field. In this case, the rules in section 14.18.1 MUST be followed.

A received message that does not have a `Date` header field MUST be assigned one by the recipient if the message will be cached by that recipient or gatewayed via a protocol which requires a `Date`. An HTTP implementation without a clock MUST NOT cache responses without revalidating them on every use. An HTTP cache, especially a shared cache, SHOULD use a mechanism, such as NTP [28], to synchronize its clock with a reliable external standard.

Clients SHOULD only send a `Date` header field in messages that include an entity-body, as in the case of the PUT and POST requests, and even then it is optional. A client without a clock MUST NOT send a `Date` header field in a request.

The `HTTP-date` sent in a `Date` header SHOULD NOT represent a date and time subsequent to the generation of the message. It SHOULD represent the best available approximation of the date and time of message generation, unless the implementation has no means of generating a reasonably accurate date and time. In theory, the date ought to represent the moment just before the entity is generated. In practice, the date can be generated at any time during the message origination without affecting its semantic value.

### 14.18.1 Clockless Origin Server Operation

Some origin server implementations might not have a clock available. An origin server without a clock **MUST NOT** assign `Expires` or `Last-Modified` values to a response, unless these values were associated with the resource by a system or user with a reliable clock. It **MAY** assign an `Expires` value that is known, at or before server configuration time, to be in the past (this allows "pre-expiration" of responses without storing separate `Expires` values for each resource).

## 14.19 ETag

The `ETag` response-header field provides the current value of the entity tag for the requested variant. The headers used with entity tags are described in sections 14.24, 14.26 and 14.44. The entity tag **MAY** be used for comparison with other entities from the same resource (see section 13.3.3).

```
ETag = "ETag" ":" entity-tag
```

Examples:

```
ETag: "xyzzy"
ETag: W/"xyzzy"
ETag: ""
```

## 14.20 Expect

The `Expect` request-header field is used to indicate that particular server behaviors are required by the client.

```
Expect          = "Expect" ":" 1#expectation
expectation    = "100-continue" | expectation-extension
expectation-extension = token [ "=" ( token | quoted-string )
                               *expect-params ]
expect-params  = ";" token [ "=" ( token | quoted-string ) ]
```

A server that does not understand or is unable to comply with any of the expectation values in the `Expect` field of a request **MUST** respond with appropriate error status. The server **MUST** respond with a 417 (Expectation Failed) status if any of the expectations cannot be met or, if there are other problems with the request, some other 4xx status.

This header field is defined with extensible syntax to allow for future extensions. If a server receives a request containing an `Expect` field that includes an expectation-extension that it does not support, it **MUST** respond with a 417 (Expectation Failed) status.

Comparison of expectation values is case-insensitive for unquoted tokens (including the 100-continue token), and is case-sensitive for quoted-string expectation-extensions.

The `Expect` mechanism is hop-by-hop: that is, an HTTP/1.1 proxy **MUST** return a 417 (Expectation Failed) status if it receives a request with an expectation that it cannot meet. However, the `Expect` request-header itself is end-to-end; it **MUST** be forwarded if the request is forwarded.

Many older HTTP/1.0 and HTTP/1.1 applications do not understand the `Expect` header.

See section 8.2.3 for the use of the 100 (continue) status.

## 14.21 Expires

The `Expires` entity-header field gives the date/time after which the response is considered stale. A stale cache entry may not normally be returned by a cache (either a proxy cache or a user agent cache) unless it is first validated with the origin server (or with an intermediate cache that has a fresh copy of the entity). See section 13.2 for further discussion of the expiration model.

The presence of an `Expires` field does not imply that the original resource will change or cease to exist at, before, or after that time.

The format is an absolute date and time as defined by `HTTP-date` in section 3.3.1; it **MUST** be in RFC 1123 date format:

```
Expires = "Expires" ":" HTTP-date
```

An example of its use is

```
Expires: Thu, 01 Dec 1994 16:00:00 GMT
```

Note: if a response includes a `Cache-Control` field with the `max-age` directive (see section 14.9.3), that directive overrides the `Expires` field.

HTTP/1.1 clients and caches **MUST** treat other invalid date formats, especially including the value "0", as in the past (i.e., "already expired").

To mark a response as "already expired," an origin server sends an `Expires` date that is equal to the `Date` header value. (See the rules for expiration calculations in section 13.2.4.)

To mark a response as "never expires," an origin server sends an `Expires` date approximately one year from the time the response is sent. HTTP/1.1 servers **SHOULD NOT** send `Expires` dates more than one year in the future.

The presence of an `Expires` header field with a date value of some time in the future on a response that otherwise would by default be non-cacheable indicates that the response is cacheable, unless indicated otherwise by a `Cache-Control` header field (section 14.9).

## 14.22 From

The `From` request-header field, if given, **SHOULD** contain an Internet e-mail address for the human user who controls the requesting user agent. The address **SHOULD** be machine-usable, as defined by "mailbox" in RFC 822 [9] as updated by RFC 1123 [8]:

```
From = "From" ":" mailbox
```

An example is:

```
From: webmaster@w3.org
```

This header field **MAY** be used for logging purposes and as a means for identifying the source of invalid or unwanted requests. It **SHOULD NOT** be used as an insecure form of access protection. The interpretation of this field is that the request is being performed on behalf of the person given, who accepts responsibility for the `method` performed. In particular, robot agents **SHOULD** include this header so that the person responsible for running the robot can be contacted if problems occur on the receiving end.

The Internet e-mail address in this field **MAY** be separate from the Internet host which issued the request. For example, when a request is passed through a proxy the original issuer's address **SHOULD** be used.

The client **SHOULD NOT** send the `From` header field without the user's approval, as it might conflict with the user's privacy interests or their site's security policy. It is strongly recommended that the user be able to disable, enable, and modify the value of this field at any time prior to a request.

## 14.23 Host

The `Host` request-header field specifies the Internet host and port number of the resource being requested, as obtained from the original URI given by the user or referring resource (generally an HTTP URL, as described in section 3.2.2). The `Host` field value **MUST** represent the naming authority of the origin server or gateway given by the original URL. This allows the origin server or gateway to differentiate between internally-ambiguous URLs, such as the root "/" URL of a server for multiple host names on a single IP address.

```
Host = "Host" ":" host [ ":" port ] ; Section 3.2.2
```

A "host" without any trailing port information implies the default port for the service requested (e.g., "80" for an HTTP URL). For example, a request on the origin server for `<http://www.w3.org/pub/WWW/>` would properly include:

```
GET /pub/WWW/ HTTP/1.1
Host: www.w3.org
```

A client **MUST** include a `Host` header field in all HTTP/1.1 request messages. If the requested URI does not include an Internet host name for the service being requested, then the `Host` header field **MUST** be given with an empty value. An HTTP/1.1 proxy **MUST** ensure that any request message it forwards does contain an appropriate `Host` header field that identifies the service being requested by the proxy. All Internet-based HTTP/1.1 servers **MUST** respond with a 400 (Bad Request) status code to any HTTP/1.1 request message which lacks a `Host` header field.

See sections 5.2 and 19.6.1.1 for other requirements relating to `Host`.

## 14.24 If-Match

The `If-Match` request-header field is used with a method to make it conditional. A client that has one or more entities previously obtained from the resource can verify that one of those entities is current by including a list of their associated entity tags in the `If-Match` header field. Entity tags are defined in section 3.11. The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead. It is also used, on updating requests, to prevent inadvertent modification of the wrong version of a resource. As a special case, the value "\*" matches any current entity of the resource.

```
If-Match = "If-Match" ":" ( "*" | 1#entity-tag )
```

If any of the entity tags match the entity tag of the entity that would have been returned in the response to a similar GET request (without the `If-Match` header) on that resource, or if "\*" is given and any current entity exists for that resource, then the server **MAY** perform the requested method as if the `If-Match` header field did not exist.

A server **MUST** use the strong comparison function (see section 13.3.3) to compare the entity tags in `If-Match`.

If none of the entity tags match, or if "\*" is given and no current entity exists, the server **MUST NOT** perform the requested method, and **MUST** return a 412 (Precondition Failed) response. This behavior is most useful when the client wants to prevent an updating method, such as PUT, from modifying a resource that has changed since the client last retrieved it.

If the request would, without the `If-Match` header field, result in anything other than a 2xx or 412 status, then the `If-Match` header **MUST** be ignored.

The meaning of "`If-Match: *`" is that the method **SHOULD** be performed if the representation selected by the origin server (or by a cache, possibly using the `Vary` mechanism, see section 14.44) exists, and **MUST NOT** be performed if the representation does not exist.

A request intended to update a resource (e.g., a PUT) **MAY** include an `If-Match` header field to signal that the request method **MUST NOT** be applied if the entity corresponding to the `If-Match` value (a single entity tag) is no longer a representation of that resource. This allows the user to indicate that they do not wish the request to be successful if the resource has been changed without their knowledge. Examples:

```
If-Match: "xyzzy"
If-Match: "xyzzy", "r2d2xxxx", "c3piozzzz"
If-Match: *
```

The result of a request having both an `If-Match` header field and either an `If-None-Match` or an `If-Modified-Since` header fields is undefined by this specification.

## 14.25 If-Modified-Since

The `If-Modified-Since` request-header field is used with a method to make it conditional: if the requested variant has not been modified since the time specified in this field, an entity will not be returned from the server; instead, a 304 (not modified) response will be returned without any message-body.

```
If-Modified-Since = "If-Modified-Since" ":" HTTP-date
```

An example of the field is:

```
If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

A GET method with an `If-Modified-Since` header and no `Range` header requests that the identified entity be transferred only if it has been modified since the date given by the `If-Modified-Since` header. The algorithm for determining this includes the following cases:

- a) If the request would normally result in anything other than a 200 (OK) status, or if the passed `If-Modified-Since` date is invalid, the response is exactly the same as for a normal GET. A date which is later than the server's current time is invalid.
- b) If the variant has been modified since the `If-Modified-Since` date, the response is exactly the same as for a normal GET.
- c) If the variant has not been modified since a valid `If-Modified-Since` date, the server SHOULD return a 304 (Not Modified) response.

The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead.

Note: The `Range` request-header field modifies the meaning of `If-Modified-Since`; see section 14.35 for full details.

Note: `If-Modified-Since` times are interpreted by the server, whose clock might not be synchronized with the client.

Note: When handling an `If-Modified-Since` header field, some servers will use an exact date comparison function, rather than a less-than function, for deciding whether to send a 304 (Not Modified) response. To get best results when sending an `If-Modified-Since` header field for cache validation, clients are advised to use the exact date string received in a previous `Last-Modified` header field whenever possible.

Note: If a client uses an arbitrary date in the `If-Modified-Since` header instead of a date taken from the `Last-Modified` header for the same request, the client should be aware of the fact that this date is interpreted in the server's understanding of time. The client should consider unsynchronized clocks and rounding problems due to the different encodings of time between the client and server. This includes the possibility of race conditions if the document has changed between the time it was first requested and the `If-Modified-Since` date of a subsequent request, and the possibility of clock-skew-related problems if the `If-Modified-Since` date is derived from the client's clock without correction to the server's clock. Corrections for different time bases between client and server are at best approximate due to network latency.

The result of a request having both an `If-Modified-Since` header field and either an `If-Match` or an `If-Unmodified-Since` header fields is undefined by this specification.

## 14.26 If-None-Match

The `If-None-Match` request-header field is used with a method to make it conditional. A client that has one or more entities previously obtained from the resource can verify that none of those entities is current by including a list of their associated entity tags in the `If-None-Match` header field. The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead. It is also used to prevent a method (e.g. PUT) from inadvertently modifying an existing resource when the client believes that the resource does not exist.

As a special case, the value "\*" matches any current entity of the resource.

```
If-None-Match = "If-None-Match" ":" ( "*" | 1#entity-tag )
```

If any of the entity tags match the entity tag of the entity that would have been returned in the response to a similar GET request (without the `If-None-Match` header) on that resource, or if "\*" is given and any current entity exists for that resource, then the server MUST NOT perform the requested method, unless required to do so because the

resource's modification date fails to match that supplied in an `If-Modified-Since` header field in the request. Instead, if the request method was `GET` or `HEAD`, the server **SHOULD** respond with a 304 (Not Modified) response, including the cache-related header fields (particularly `ETag`) of one of the entities that matched. For all other request methods, the server **MUST** respond with a status of 412 (Precondition Failed).

See section 13.3.3 for rules on how to determine if two entities tags match. The weak comparison function can only be used with `GET` or `HEAD` requests.

If none of the entity tags match, then the server **MAY** perform the requested method as if the `If-None-Match` header field did not exist, but **MUST** also ignore any `If-Modified-Since` header field(s) in the request. That is, if no entity tags match, then the server **MUST NOT** return a 304 (Not Modified) response.

If the request would, without the `If-None-Match` header field, result in anything other than a 2xx or 304 status, then the `If-None-Match` header **MUST** be ignored. (See section 13.3.4 for a discussion of server behavior when both `If-Modified-Since` and `If-None-Match` appear in the same request.)

The meaning of "`If-None-Match: *`" is that the method **MUST NOT** be performed if the representation selected by the origin server (or by a cache, possibly using the `Vary` mechanism, see section 14.44) exists, and **SHOULD** be performed if the representation does not exist. This feature is intended to be useful in preventing races between `PUT` operations.

Examples:

```
If-None-Match: "xyzzy"
If-None-Match: W/"xyzzy"
If-None-Match: "xyzzy", "r2d2xxxx", "c3piozzzz"
If-None-Match: W/"xyzzy", W/"r2d2xxxx", W/"c3piozzzz"
If-None-Match: *
```

The result of a request having both an `If-None-Match` header field and either an `If-Match` or an `If-Unmodified-Since` header fields is undefined by this specification.

## 14.27 If-Range

If a client has a partial copy of an entity in its cache, and wishes to have an up-to-date copy of the entire entity in its cache, it could use the `Range` request-header with a conditional `GET` (using either or both of `If-Unmodified-Since` and `If-Match`.) However, if the condition fails because the entity has been modified, the client would then have to make a second request to obtain the entire current entity-body.

The `If-Range` header allows a client to "short-circuit" the second request. Informally, its meaning is 'if the entity is unchanged, send me the part(s) that I am missing; otherwise, send me the entire new entity.'

```
If-Range = "If-Range" ":" ( entity-tag | HTTP-date )
```

If the client has no entity tag for an entity, but does have a `Last-Modified` date, it **MAY** use that date in an `If-Range` header. (The server can distinguish between a valid `HTTP-date` and any form of `entity-tag` by examining no more than two characters.) The `If-Range` header **SHOULD** only be used together with a `Range` header, and **MUST** be ignored if the request does not include a `Range` header, or if the server does not support the sub-range operation.

If the entity tag given in the `If-Range` header matches the current entity tag for the entity, then the server **SHOULD** provide the specified sub-range of the entity using a 206 (Partial content) response. If the entity tag does not match, then the server **SHOULD** return the entire entity using a 200 (OK) response.

## 14.28 If-Unmodified-Since

The `If-Unmodified-Since` request-header field is used with a method to make it conditional. If the requested resource has not been modified since the time specified in this field, the server **SHOULD** perform the requested operation as if the `If-Unmodified-Since` header were not present.

If the requested variant has been modified since the specified time, the server **MUST NOT** perform the requested operation, and **MUST** return a 412 (Precondition Failed).

```
If-Unmodified-Since = "If-Unmodified-Since" ":" HTTP-date
```

An example of the field is:

```
If-Unmodified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

If the request normally (i.e., without the `If-Unmodified-Since` header) would result in anything other than a 2xx or 412 status, the `If-Unmodified-Since` header **SHOULD** be ignored.

If the specified date is invalid, the header is ignored.

The result of a request having both an `If-Unmodified-Since` header field and either an `If-None-Match` or an `If-Modified-Since` header fields is undefined by this specification.

## 14.29 Last-Modified

The `Last-Modified` entity-header field indicates the date and time at which the origin server believes the variant was last modified.

```
Last-Modified = "Last-Modified" ":" HTTP-date
```

An example of its use is

```
Last-Modified: Tue, 15 Nov 1994 12:45:26 GMT
```

The exact meaning of this header field depends on the implementation of the origin server and the nature of the original resource. For files, it may be just the file system last-modified time. For entities with dynamically included parts, it may be the most recent of the set of last-modify times for its component parts. For database gateways, it may be the last-update time stamp of the record. For virtual objects, it may be the last time the internal state changed.

An origin server **MUST NOT** send a `Last-Modified` date which is later than the server's time of message origination. In such cases, where the resource's last modification would indicate some time in the future, the server **MUST** replace that date with the message origination date.

An origin server **SHOULD** obtain the `Last-Modified` value of the entity as close as possible to the time that it generates the `Date` value of its response. This allows a recipient to make an accurate assessment of the entity's modification time, especially if the entity changes near the time that the response is generated.

HTTP/1.1 servers **SHOULD** send `Last-Modified` whenever feasible.

## 14.30 Location

The `Location` response-header field is used to redirect the recipient to a location other than the `Request-URI` for completion of the request or identification of a new resource. For 201 (Created) responses, the `Location` is that of the new resource which was created by the request. For 3xx responses, the location **SHOULD** indicate the server's preferred URI for automatic redirection to the resource. The field value consists of a single absolute URI.

```
Location = "Location" ":" absoluteURI
```

An example is:

```
Location: http://www.w3.org/pub/WWW/People.html
```

Note: The `Content-Location` header field (section 14.14) differs from `Location` in that the `Content-Location` identifies the original location of the entity enclosed in the request. It is therefore possible for a response to contain header fields for both `Location` and `Content-Location`. Also see section 13.10 for cache requirements of some methods.

## 14.31 Max-Forwards

The `Max-Forwards` request-header field provides a mechanism with the `TRACE` (section 9.8) and `OPTIONS` (section 9.2) methods to limit the number of proxies or gateways that can forward the request to the next inbound



server. This can be useful when the client is attempting to trace a request chain which appears to be failing or looping in mid-chain.

```
Max-Forwards = "Max-Forwards" ":" 1*DIGIT
```

The `Max-Forwards` value is a decimal integer indicating the remaining number of times this request message may be forwarded.

Each proxy or gateway recipient of a `TRACE` or `OPTIONS` request containing a `Max-Forwards` header field **MUST** check and update its value prior to forwarding the request. If the received value is zero (0), the recipient **MUST NOT** forward the request; instead, it **MUST** respond as the final recipient. If the received `Max-Forwards` value is greater than zero, then the forwarded message **MUST** contain an updated `Max-Forwards` field with a value decremented by one (1).

The `Max-Forwards` header field **MAY** be ignored for all other methods defined by this specification and for any extension methods for which it is not explicitly referred to as part of that method definition.

### 14.32 Pragma

The `Pragma` general-header field is used to include implementation-specific directives that might apply to any recipient along the request/response chain. All pragma directives specify optional behavior from the viewpoint of the protocol; however, some systems **MAY** require that behavior be consistent with the directives.

```
Pragma = "Pragma" ":" 1#pragma-directive
pragma-directive = "no-cache" | extension-pragma
extension-pragma = token [ "=" ( token | quoted-string ) ]
```

When the `no-cache` directive is present in a request message, an application **SHOULD** forward the request toward the origin server even if it has a cached copy of what is being requested. This pragma directive has the same semantics as the `no-cache` cache-directive (see section 14.9) and is defined here for backward compatibility with HTTP/1.0. Clients **SHOULD** include both header fields when a `no-cache` request is sent to a server not known to be HTTP/1.1 compliant.

Pragma directives **MUST** be passed through by a proxy or gateway application, regardless of their significance to that application, since the directives might be applicable to all recipients along the request/response chain. It is not possible to specify a pragma for a specific recipient; however, any pragma directive not relevant to a recipient **SHOULD** be ignored by that recipient.

HTTP/1.1 caches **SHOULD** treat `Pragma: no-cache` as if the client had sent `Cache-Control: no-cache`. No new `Pragma` directives will be defined in HTTP.

Note: because the meaning of `Pragma: no-cache` as a response header field is not actually specified, it does not provide a reliable replacement for `Cache-Control: no-cache` in a response.

### 14.33 Proxy-Authenticate

The `Proxy-Authenticate` response-header field **MUST** be included as part of a 407 (Proxy Authentication Required) response. The field value consists of a challenge that indicates the authentication scheme and parameters applicable to the proxy for this `Request-URI`.

```
Proxy-Authenticate = "Proxy-Authenticate" ":" 1#challenge
```

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. Unlike `WWW-Authenticate`, the `Proxy-Authenticate` header field applies only to the current connection and **SHOULD NOT** be passed on to downstream clients. However, an intermediate proxy might need to obtain its own credentials by requesting them from the downstream client, which in some circumstances will appear as if the proxy is forwarding the `Proxy-Authenticate` header field.

## 14.34 Proxy-Authorization

The `Proxy-Authorization` request-header field allows the client to identify itself (or its user) to a proxy which requires authentication. The `Proxy-Authorization` field value consists of credentials containing the authentication information of the user agent for the proxy and/or realm of the resource being requested.

```
Proxy-Authorization = "Proxy-Authorization" ":" credentials
```

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. Unlike `Authorization`, the `Proxy-Authorization` header field applies only to the next outbound proxy that demanded authentication using the `Proxy-Authenticate` field. When multiple proxies are used in a chain, the `Proxy-Authorization` header field is consumed by the first outbound proxy that was expecting to receive credentials. A proxy MAY relay the credentials from the client request to the next proxy if that is the mechanism by which the proxies cooperatively authenticate a given request.

## 14.35 Range

### 14.35.1 Byte Ranges

Since all HTTP entities are represented in HTTP messages as sequences of bytes, the concept of a byte range is meaningful for any HTTP entity. (However, not all clients and servers need to support byte-range operations.)

Byte range specifications in HTTP apply to the sequence of bytes in the entity-body (not necessarily the same as the message-body).

A byte range operation MAY specify a single range of bytes, or a set of ranges within a single entity.

```
ranges-specifier = byte-ranges-specifier
byte-ranges-specifier = bytes-unit "=" byte-range-set
byte-range-set = 1#( byte-range-spec | suffix-byte-range-spec )
byte-range-spec = first-byte-pos "-" [last-byte-pos]
first-byte-pos = 1*DIGIT
last-byte-pos = 1*DIGIT
```

The `first-byte-pos` value in a `byte-range-spec` gives the byte-offset of the first byte in a range. The `last-byte-pos` value gives the byte-offset of the last byte in the range; that is, the byte positions specified are inclusive. Byte offsets start at zero.

If the `last-byte-pos` value is present, it MUST be greater than or equal to the `first-byte-pos` in that `byte-range-spec`, or the `byte-range-spec` is syntactically invalid. The recipient of a `byte-range-set` that includes one or more syntactically invalid `byte-range-spec` values MUST ignore the header field that includes that `byte-range-set`.

If the `last-byte-pos` value is absent, or if the value is greater than or equal to the current length of the entity-body, `last-byte-pos` is taken to be equal to one less than the current length of the entity-body in bytes.

By its choice of `last-byte-pos`, a client can limit the number of bytes retrieved without knowing the size of the entity.

```
suffix-byte-range-spec = "-" suffix-length
suffix-length = 1*DIGIT
```

A `suffix-byte-range-spec` is used to specify the suffix of the entity-body, of a length given by the `suffix-length` value. (That is, this form specifies the last N bytes of an entity-body.) If the entity is shorter than the specified `suffix-length`, the entire entity-body is used.

If a syntactically valid `byte-range-set` includes at least one `byte-range-spec` whose `first-byte-pos` is less than the current length of the entity-body, or at least one `suffix-byte-range-spec` with a non-zero `suffix-length`, then the `byte-range-set` is satisfiable. Otherwise, the `byte-range-set` is unsatisfiable. If the `byte-range-set` is unsatisfiable, the server SHOULD return a response with a status of 416 (Requested range not satisfiable). Otherwise, the server SHOULD return a response with a status of 206 (Partial Content) containing the satisfiable ranges of the entity-body.

Examples of `byte-ranges-specifier` values (assuming an entity-body of length 10000):

- The first 500 bytes (byte offsets 0-499, inclusive):

- bytes=0-499
- The second 500 bytes (byte offsets 500-999, inclusive):
  - bytes=500-999
- The final 500 bytes (byte offsets 9500-9999, inclusive):
  - bytes=-500
- Or
  - bytes=9500-
- The first and last bytes only (bytes 0 and 9999):
  - bytes=0-0,-1
- Several legal but not canonical specifications of the second 500 bytes (byte offsets 500-999, inclusive):
  - bytes=500-600,601-999
  - bytes=500-700,601-999

### 14.35.2 Range Retrieval Requests

HTTP retrieval requests using conditional or unconditional GET methods MAY request one or more sub-ranges of the entity, instead of the entire entity, using the Range request header, which applies to the entity returned as the result of the request:

```
Range = "Range" ":" ranges-specifier
```

A server MAY ignore the Range header. However, HTTP/1.1 origin servers and intermediate caches ought to support byte ranges when possible, since Range supports efficient recovery from partially failed transfers, and supports efficient partial retrieval of large entities.

If the server supports the Range header and the specified range or ranges are appropriate for the entity:

- The presence of a Range header in an unconditional GET modifies what is returned if the GET is otherwise successful. In other words, the response carries a status code of 206 (Partial Content) instead of 200 (OK).
- The presence of a Range header in a conditional GET (a request using one or both of If-Modified-Since and If-None-Match, or one or both of If-Unmodified-Since and If-Match) modifies what is returned if the GET is otherwise successful and the condition is true. It does not affect the 304 (Not Modified) response returned if the conditional is false.

In some cases, it might be more appropriate to use the If-Range header (see section 14.27) in addition to the Range header.

If a proxy that supports ranges receives a Range request, forwards the request to an inbound server, and receives an entire entity in reply, it SHOULD only return the requested range to its client. It SHOULD store the entire received response in its cache if that is consistent with its cache allocation policies.

## 14.36 Referer

The Referer[sic] request-header field allows the client to specify, for the server's benefit, the address (URI) of the resource from which the Request-URI was obtained (the "referrer", although the header field is misspelled.) The Referer request-header allows a server to generate lists of back-links to resources for interest, logging, optimized caching, etc. It also allows obsolete or mistyped links to be traced for maintenance. The Referer field MUST NOT be sent if the Request-URI was obtained from a source that does not have its own URI, such as input from the user keyboard.

```
Referer = "Referer" ":" ( absoluteURI | relativeURI )
```

Example:

```
Referer: http://www.w3.org/hypertext/DataSources/Overview.html
```

If the field value is a relative URI, it SHOULD be interpreted relative to the Request-URI. The URI MUST NOT include a fragment. See section 15.1.3 for security considerations.

### 14.37 Retry-After

The `Retry-After` response-header field can be used with a 503 (Service Unavailable) response to indicate how long the service is expected to be unavailable to the requesting client. This field MAY also be used with any 3xx (Redirection) response to indicate the minimum time the user-agent is asked wait before issuing the redirected request. The value of this field can be either an HTTP-date or an integer number of seconds (in decimal) after the time of the response.

```
Retry-After = "Retry-After" ":" ( HTTP-date | delta-seconds )
```

Two examples of its use are

```
Retry-After: Fri, 31 Dec 1999 23:59:59 GMT
Retry-After: 120
```

In the latter example, the delay is 2 minutes.

### 14.38 Server

The `Server` response-header field contains information about the software used by the origin server to handle the request. The field can contain multiple product tokens (section 3.8) and comments identifying the server and any significant subproducts. The product tokens are listed in order of their significance for identifying the application.

```
Server = "Server" ":" 1*( product | comment )
```

Example:

```
Server: CERN/3.0 libwww/2.17
```

If the response is being forwarded through a proxy, the proxy application **MUST NOT** modify the `Server` response-header. Instead, it **SHOULD** include a `Via` field (as described in section 14.45).

Note: Revealing the specific software version of the server might allow the server machine to become more vulnerable to attacks against software that is known to contain security holes. Server implementors are encouraged to make this field a configurable option.

### 14.39 TE

The `TE` request-header field indicates what extension transfer-codings it is willing to accept in the response and whether or not it is willing to accept trailer fields in a chunked transfer-coding. Its value may consist of the keyword "trailers" and/or a comma-separated list of extension transfer-coding names with optional accept parameters (as described in section 3.6).

```
TE = "TE" ":" #( t-codings )
t-codings = "trailers" | ( transfer-extension [ accept-params ] )
```

The presence of the keyword "trailers" indicates that the client is willing to accept trailer fields in a chunked transfer-coding, as defined in section 3.6.1. This keyword is reserved for use with transfer-coding values even though it does not itself represent a transfer-coding.

Examples of its use are:

```
TE: deflate
TE:
TE: trailers, deflate;q=0.5
```

The `TE` header field only applies to the immediate connection. Therefore, the keyword **MUST** be supplied within a `Connection` header field (section 14.10) whenever `TE` is present in an HTTP/1.1 message.

A server tests whether a transfer-coding is acceptable, according to a `TE` field, using these rules:

1. The "chunked" transfer-coding is always acceptable. If the keyword "trailers" is listed, the client indicates that it is willing to accept trailer fields in the chunked response on behalf of itself and any downstream clients. The implication is that, if given, the client is stating that either all downstream clients are willing to accept trailer fields in the forwarded response, or that it will attempt to buffer the response on behalf of downstream recipients.

Note: HTTP/1.1 does not define any means to limit the size of a chunked response such that a client can be assured of buffering the entire response.

2. If the transfer-coding being tested is one of the transfer-codings listed in the TE field, then it is acceptable unless it is accompanied by a qvalue of 0. (As defined in section 3.9, a qvalue of 0 means “not acceptable.”)
3. If multiple transfer-codings are acceptable, then the acceptable transfer-coding with the highest non-zero qvalue is preferred. The “chunked” transfer-coding always has a qvalue of 1.

If the TE field-value is empty or if no TE field is present, the only transfer-coding is “chunked”. A message with no transfer-coding is always acceptable.

## 14.40 Trailer

The Trailer general field value indicates that the given set of header fields is present in the trailer of a message encoded with chunked transfer-coding.

```
Trailer = "Trailer" ":" 1#field-name
```

An HTTP/1.1 message SHOULD include a Trailer header field in a message using chunked transfer-coding with a non-empty trailer. Doing so allows the recipient to know which header fields to expect in the trailer.

If no Trailer header field is present, the trailer SHOULD NOT include any header fields. See section 3.6.1 for restrictions on the use of trailer fields in a “chunked” transfer-coding.

Message header fields listed in the Trailer header field MUST NOT include the following header fields:

- Transfer-Encoding
- Content-Length
- Trailer

## 14.41 Transfer-Encoding

The Transfer-Encoding general-header field indicates what (if any) type of transformation has been applied to the message body in order to safely transfer it between the sender and the recipient. This differs from the content-coding in that the transfer-coding is a property of the message, not of the entity.

```
Transfer-Encoding = "Transfer-Encoding" ":" 1#transfer-coding
```

Transfer-codings are defined in section 3.6. An example is:

```
Transfer-Encoding: chunked
```

If multiple encodings have been applied to an entity, the transfer-codings MUST be listed in the order in which they were applied. Additional information about the encoding parameters MAY be provided by other entity-header fields not defined by this specification.

Many older HTTP/1.0 applications do not understand the Transfer-Encoding header.

## 14.42 Upgrade

The Upgrade general-header allows the client to specify what additional communication protocols it supports and would like to use if the server finds it appropriate to switch protocols. The server MUST use the Upgrade header field within a 101 (Switching Protocols) response to indicate which protocol(s) are being switched.

```
Upgrade = "Upgrade" ":" 1#product
```

For example,

```
Upgrade: HTTP/2.0, SHTTP/1.3, IRC/6.9, RTA/x11
```

The `Upgrade` header field is intended to provide a simple mechanism for transition from HTTP/1.1 to some other, incompatible protocol. It does so by allowing the client to advertise its desire to use another protocol, such as a later version of HTTP with a higher major version number, even though the current request has been made using HTTP/1.1. This eases the difficult transition between incompatible protocols by allowing the client to initiate a request in the more commonly supported protocol while indicating to the server that it would like to use a “better” protocol if available (where “better” is determined by the server, possibly according to the nature of the method and/or resource being requested).

The `Upgrade` header field only applies to switching application-layer protocols upon the existing transport-layer connection. `Upgrade` cannot be used to insist on a protocol change; its acceptance and use by the server is optional. The capabilities and nature of the application-layer communication after the protocol change is entirely dependent upon the new protocol chosen, although the first action after changing the protocol **MUST** be a response to the initial HTTP request containing the `Upgrade` header field.

The `Upgrade` header field only applies to the immediate connection. Therefore, the `upgrade` keyword **MUST** be supplied within a `Connection` header field (section 14.10) whenever `Upgrade` is present in an HTTP/1.1 message.

The `Upgrade` header field cannot be used to indicate a switch to a protocol on a different connection. For that purpose, it is more appropriate to use a 301, 302, 303, or 305 redirection response.

This specification only defines the protocol name “HTTP” for use by the family of Hypertext Transfer Protocols, as defined by the HTTP version rules of section 3.1 and future updates to this specification. Any token can be used as a protocol name; however, it will only be useful if both the client and server associate the name with the same protocol.

### 14.43 User-Agent

The `User-Agent` request-header field contains information about the user agent originating the request. This is for statistical purposes, the tracing of protocol violations, and automated recognition of user agents for the sake of tailoring responses to avoid particular user agent limitations. User agents **SHOULD** include this field with requests. The field can contain multiple product tokens (section 3.8) and comments identifying the agent and any subproducts which form a significant part of the user agent. By convention, the product tokens are listed in order of their significance for identifying the application.

```
User-Agent = "User-Agent" ":" 1*( product | comment )
```

Example:

```
User-Agent: CERN-LineMode/2.15 libwww/2.17b3
```

### 14.44 Vary

The `Vary` field value indicates the set of request-header fields that fully determines, while the response is fresh, whether a cache is permitted to use the response to reply to a subsequent request without revalidation. For uncacheable or stale responses, the `Vary` field value advises the user agent about the criteria that were used to select the representation. A `Vary` field value of “\*” implies that a cache cannot determine from the request headers of a subsequent request whether this response is the appropriate representation. See section 13.6 for use of the `Vary` header field by caches.

```
Vary = "Vary" ":" ( "*" | 1#field-name )
```

An HTTP/1.1 server **SHOULD** include a `Vary` header field with any cacheable response that is subject to server-driven negotiation. Doing so allows a cache to properly interpret future requests on that resource and informs the user agent about the presence of negotiation on that resource. A server **MAY** include a `Vary` header field with a non-cacheable response that is subject to server-driven negotiation, since this might provide the user agent with useful information about the dimensions over which the response varies at the time of the response.

A `Vary` field value consisting of a list of field-names signals that the representation selected for the response is based on a selection algorithm which considers **ONLY** the listed request-header field values in selecting the most

appropriate representation. A cache MAY assume that the same selection will be made for future requests with the same values for the listed field names, for the duration of time for which the response is fresh.

The field-names given are not limited to the set of standard request-header fields defined by this specification. Field names are case-insensitive.

A Vary field value of "\*" signals that unspecified parameters not limited to the request-headers (e.g., the network address of the client), play a role in the selection of the response representation. The "\*" value MUST NOT be generated by a proxy server; it may only be generated by an origin server.

## 14.45 Via

The Via general-header field MUST be used by gateways and proxies to indicate the intermediate protocols and recipients between the user agent and the server on requests, and between the origin server and the client on responses. It is analogous to the "Received" field of RFC 822 [9] and is intended to be used for tracking message forwards, avoiding request loops, and identifying the protocol capabilities of all senders along the request/response chain.

```
Via = "Via" ":" 1#( received-protocol received-by [ comment ] )
received-protocol = [ protocol-name "/" ] protocol-version
protocol-name     = token
protocol-version  = token
received-by       = ( host [ ":" port ] ) | pseudonym
pseudonym         = token
```

The received-protocol indicates the protocol version of the message received by the server or client along each segment of the request/response chain. The received-protocol version is appended to the Via field value when the message is forwarded so that information about the protocol capabilities of upstream applications remains visible to all recipients.

The protocol-name is optional if and only if it would be "HTTP". The received-by field is normally the host and optional port number of a recipient server or client that subsequently forwarded the message. However, if the real host is considered to be sensitive information, it MAY be replaced by a pseudonym. If the port is not given, it MAY be assumed to be the default port of the received-protocol.

Multiple Via field values represents each proxy or gateway that has forwarded the message. Each recipient MUST append its information such that the end result is ordered according to the sequence of forwarding applications.

Comments MAY be used in the Via header field to identify the software of the recipient proxy or gateway, analogous to the User-Agent and Server header fields. However, all comments in the Via field are optional and MAY be removed by any recipient prior to forwarding the message.

For example, a request message could be sent from an HTTP/1.0 user agent to an internal proxy code-named "fred", which uses HTTP/1.1 to forward the request to a public proxy at nowhere.com, which completes the request by forwarding it to the origin server at www.ics.uci.edu. The request received by www.ics.uci.edu would then have the following Via header field:

```
Via: 1.0 fred, 1.1 nowhere.com (Apache/1.1)
```

Proxies and gateways used as a portal through a network firewall SHOULD NOT, by default, forward the names and ports of hosts within the firewall region. This information SHOULD only be propagated if explicitly enabled. If not enabled, the received-by host of any host behind the firewall SHOULD be replaced by an appropriate pseudonym for that host.

For organizations that have strong privacy requirements for hiding internal structures, a proxy MAY combine an ordered subsequence of Via header field entries with identical received-protocol values into a single such entry. For example,

```
Via: 1.0 ricky, 1.1 ethel, 1.1 fred, 1.0 lucy
could be collapsed to
Via: 1.0 ricky, 1.1 mertz, 1.0 lucy
```

Applications **SHOULD NOT** combine multiple entries unless they are all under the same organizational control and the hosts have already been replaced by pseudonyms. Applications **MUST NOT** combine entries which have different received-protocol values.

## 14.46 Warning

The `Warning` general-header field is used to carry additional information about the status or transformation of a message which might not be reflected in the message. This information is typically used to warn about a possible lack of semantic transparency from caching operations or transformations applied to the entity body of the message.

Warning headers are sent with responses using:

```
Warning      = "Warning" ":" 1#warning-value
warning-value = warn-code SP warn-agent SP warn-text
               [SP warn-date]

warn-code    = 3DIGIT
warn-agent   = ( host [ ":" port ] ) | pseudonym
               ; the name or pseudonym of the server adding
               ; the Warning header, for use in debugging
warn-text    = quoted-string
warn-date    = <"> HTTP-date <">
```

A response **MAY** carry more than one `Warning` header.

The `warn-text` **SHOULD** be in a natural language and character set that is most likely to be intelligible to the human user receiving the response. This decision **MAY** be based on any available knowledge, such as the location of the cache or user, the `Accept-Language` field in a request, the `Content-Language` field in a response, etc. The default language is English and the default character set is ISO-8859-1.

If a character set other than ISO-8859-1 is used, it **MUST** be encoded in the `warn-text` using the method described in RFC 2047 [14].

Warning headers can in general be applied to any message, however some specific `warn-codes` are specific to caches and can only be applied to response messages. New `Warning` headers **SHOULD** be added after any existing `Warning` headers. A cache **MUST NOT** delete any `Warning` header that it received with a message. However, if a cache successfully validates a cache entry, it **SHOULD** remove any `Warning` headers previously attached to that entry except as specified for specific `Warning` codes. It **MUST** then add any `Warning` headers received in the validating response. In other words, `Warning` headers are those that would be attached to the most recent relevant response.

When multiple `Warning` headers are attached to a response, the user agent ought to inform the user of as many of them as possible, in the order that they appear in the response. If it is not possible to inform the user of all of the warnings, the user agent **SHOULD** follow these heuristics:

- Warnings that appear early in the response take priority over those appearing later in the response.
- Warnings in the user's preferred character set take priority over warnings in other character sets but with identical `warn-codes` and `warn-agents`.

Systems that generate multiple `Warning` headers **SHOULD** order them with this user agent behavior in mind.

Requirements for the behavior of caches with respect to Warnings are stated in section 13.1.2.

This is a list of the currently-defined `warn-codes`, each with a recommended `warn-text` in English, and a description of its meaning.

110 Response is stale

**MUST** be included whenever the returned response is stale.



**111 Revalidation failed**

MUST be included if a cache returns a stale response because an attempt to revalidate the response failed, due to an inability to reach the server.

**112 Disconnected operation**

SHOULD be included if the cache is intentionally disconnected from the rest of the network for a period of time.

**113 Heuristic expiration**

MUST be included if the cache heuristically chose a freshness lifetime greater than 24 hours and the response's age is greater than 24 hours.

**199 Miscellaneous warning**

The warning text MAY include arbitrary information to be presented to a human user, or logged. A system receiving this warning MUST NOT take any automated action, besides presenting the warning to the user.

**214 Transformation applied**

MUST be added by an intermediate cache or proxy if it applies any transformation changing the content-coding (as specified in the `Content-Encoding` header) or media-type (as specified in the `Content-Type` header) of the response, or the entity-body of the response, unless this `Warning` code already appears in the response.

**299 Miscellaneous persistent warning**

The warning text MAY include arbitrary information to be presented to a human user, or logged. A system receiving this warning MUST NOT take any automated action.

If an implementation sends a message with one or more `Warning` headers whose version is HTTP/1.0 or lower, then the sender MUST include in each `warning-value` a `warn-date` that matches the date in the response.

If an implementation receives a message with a `warning-value` that includes a `warn-date`, and that `warn-date` is different from the `Date` value in the response, then that `warning-value` MUST be deleted from the message before storing, forwarding, or using it. (This prevents bad consequences of naive caching of `Warning` header fields.) If all of the `warning-values` are deleted for this reason, the `Warning` header MUST be deleted as well.

## 14.47 WWW-Authenticate

The `WWW-Authenticate` response-header field MUST be included in 401 (Unauthorized) response messages. The field value consists of at least one `challenge` that indicates the authentication scheme(s) and parameters applicable to the `Request-URI`.

```
WWW-Authenticate = "WWW-Authenticate" ":" 1#challenge
```

The HTTP access authentication process is described in "HTTP Authentication: Basic and Digest Access Authentication" [43]. User agents are advised to take special care in parsing the `WWW-Authenticate` field value as it might contain more than one challenge, or if more than one `WWW-Authenticate` header field is provided, the contents of a challenge itself can contain a comma-separated list of authentication parameters.

# 15 Security Considerations

This section is meant to inform application developers, information providers, and users of the security limitations in HTTP/1.1 as described by this document. The discussion does not include definitive solutions to the problems revealed, though it does make some suggestions for reducing security risks.

## 15.1 Personal Information

HTTP clients are often privy to large amounts of personal information (e.g. the user's name, location, mail address, passwords, encryption keys, etc.), and SHOULD be very careful to prevent unintentional leakage of this information via the HTTP protocol to other sources. We very strongly recommend that a convenient interface be provided for the

user to control dissemination of such information, and that designers and implementors be particularly careful in this area. History shows that errors in this area often create serious security and/or privacy problems and generate highly adverse publicity for the implementor's company.

### 15.1.1 Abuse of Server Log Information

A server is in the position to save personal data about a user's requests which might identify their reading patterns or subjects of interest. This information is clearly confidential in nature and its handling can be constrained by law in certain countries. People using the HTTP protocol to provide data are responsible for ensuring that such material is not distributed without the permission of any individuals that are identifiable by the published results.

### 15.1.2 Transfer of Sensitive Information

Like any generic data transfer protocol, HTTP cannot regulate the content of the data that is transferred, nor is there any *a priori* method of determining the sensitivity of any particular piece of information within the context of any given request. Therefore, applications SHOULD supply as much control over this information as possible to the provider of that information. Four header fields are worth special mention in this context: `Server`, `Via`, `Referer` and `From`.

Revealing the specific software version of the server might allow the server machine to become more vulnerable to attacks against software that is known to contain security holes. Implementors SHOULD make the `Server` header field a configurable option.

Proxies which serve as a portal through a network firewall SHOULD take special precautions regarding the transfer of header information that identifies the hosts behind the firewall. In particular, they SHOULD remove, or replace with sanitized versions, any `Via` fields generated behind the firewall.

The `Referer` header allows reading patterns to be studied and reverse links drawn. Although it can be very useful, its power can be abused if user details are not separated from the information contained in the `Referer`. Even when the personal information has been removed, the `Referer` header might indicate a private document's URI whose publication would be inappropriate.

The information sent in the `From` field might conflict with the user's privacy interests or their site's security policy, and hence it SHOULD NOT be transmitted without the user being able to disable, enable, and modify the contents of the field. The user MUST be able to set the contents of this field within a user preference or application defaults configuration.

We suggest, though do not require, that a convenient toggle interface be provided for the user to enable or disable the sending of `From` and `Referer` information.

The `User-Agent` (section 14.43) or `Server` (section 14.38) header fields can sometimes be used to determine that a specific client or server have a particular security hole which might be exploited. Unfortunately, this same information is often used for other valuable purposes for which HTTP currently has no better mechanism.

### 15.1.3 Encoding Sensitive Information in URI's

Because the source of a link might be private information or might reveal an otherwise private information source, it is strongly recommended that the user be able to select whether or not the `Referer` field is sent. For example, a browser client could have a toggle switch for browsing openly/anonymously, which would respectively enable/disable the sending of `Referer` and `From` information.

Clients SHOULD NOT include a `Referer` header field in a (non-secure) HTTP request if the referring page was transferred with a secure protocol.

Authors of services which use the HTTP protocol SHOULD NOT use GET based forms for the submission of sensitive data, because this will cause this data to be encoded in the `Request-URI`. Many existing servers, proxies, and user agents will log the request URI in some place where it might be visible to third parties. Servers can use POST-based form submission instead

#### 15.1.4 Privacy Issues Connected to Accept Headers

Accept request-headers can reveal information about the user to all servers which are accessed. The Accept-Language header in particular can reveal information the user would consider to be of a private nature, because the understanding of particular languages is often strongly correlated to the membership of a particular ethnic group. User agents which offer the option to configure the contents of an Accept-Language header to be sent in every request are strongly encouraged to let the configuration process include a message which makes the user aware of the loss of privacy involved.

An approach that limits the loss of privacy would be for a user agent to omit the sending of Accept-Language headers by default, and to ask the user whether or not to start sending Accept-Language headers to a server if it detects, by looking for any Vary response-header fields generated by the server, that such sending could improve the quality of service.

Elaborate user-customized accept header fields sent in every request, in particular if these include quality values, can be used by servers as relatively reliable and long-lived user identifiers. Such user identifiers would allow content providers to do click-trail tracking, and would allow collaborating content providers to match cross-server click-trails or form submissions of individual users. Note that for many users not behind a proxy, the network address of the host running the user agent will also serve as a long-lived user identifier. In environments where proxies are used to enhance privacy, user agents ought to be conservative in offering accept header configuration options to end users. As an extreme privacy measure, proxies could filter the accept headers in relayed requests. General purpose user agents which provide a high degree of header configurability SHOULD warn users about the loss of privacy which can be involved.

### 15.2 Attacks Based On File and Path Names

Implementations of HTTP origin servers SHOULD be careful to restrict the documents returned by HTTP requests to be only those that were intended by the server administrators. If an HTTP server translates HTTP URIs directly into file system calls, the server MUST take special care not to serve files that were not intended to be delivered to HTTP clients. For example, UNIX, Microsoft Windows, and other operating systems use “.” as a path component to indicate a directory level above the current one. On such a system, an HTTP server MUST disallow any such construct in the Request-URI if it would otherwise allow access to a resource outside those intended to be accessible via the HTTP server. Similarly, files intended for reference only internally to the server (such as access control files, configuration files, and script code) MUST be protected from inappropriate retrieval, since they might contain sensitive information. Experience has shown that minor bugs in such HTTP server implementations have turned into security risks.

### 15.3 DNS Spoofing

Clients using HTTP rely heavily on the Domain Name Service, and are thus generally prone to security attacks based on the deliberate mis-association of IP addresses and DNS names. Clients need to be cautious in assuming the continuing validity of an IP number/DNS name association.

In particular, HTTP clients SHOULD rely on their name resolver for confirmation of an IP number/DNS name association, rather than caching the result of previous host name lookups. Many platforms already can cache host name lookups locally when appropriate, and they SHOULD be configured to do so. It is proper for these lookups to be cached, however, only when the TTL (Time To Live) information reported by the name server makes it likely that the cached information will remain useful.

If HTTP clients cache the results of host name lookups in order to achieve a performance improvement, they MUST observe the TTL information reported by DNS.

If HTTP clients do not observe this rule, they could be spoofed when a previously-accessed server's IP address changes. As network renumbering is expected to become increasingly common [24], the possibility of this form of attack will grow. Observing this requirement thus reduces this potential security vulnerability.

This requirement also improves the load-balancing behavior of clients for replicated servers using the same DNS name and reduces the likelihood of a user's experiencing failure in accessing sites which use that strategy.

## 15.4 Location Headers and Spoofing

If a single server supports multiple organizations that do not trust one another, then it **MUST** check the values of `Location` and `Content-Location` headers in responses that are generated under control of said organizations to make sure that they do not attempt to invalidate resources over which they have no authority.

## 15.5 Content-Disposition Issues

RFC 1806 [35], from which the often implemented `Content-Disposition` (see section 19.5.1) header in HTTP is derived, has a number of very serious security considerations. `Content-Disposition` is not part of the HTTP standard, but since it is widely implemented, we are documenting its use and risks for implementors. See RFC 2183 [49] (which updates RFC 1806) for details.

## 15.6 Authentication Credentials and Idle Clients

Existing HTTP clients and user agents typically retain authentication information indefinitely. HTTP/1.1 does not provide a method for a server to direct clients to discard these cached credentials. This is a significant defect that requires further extensions to HTTP. Circumstances under which credential caching can interfere with the application's security model include but are not limited to:

- Clients which have been idle for an extended period following which the server might wish to cause the client to reprompt the user for credentials.
- Applications which include a session termination indication (such as a 'logout' or 'commit' button on a page) after which the server side of the application 'knows' that there is no further reason for the client to retain the credentials.

This is currently under separate study. There are a number of work-arounds to parts of this problem, and we encourage the use of password protection in screen savers, idle time-outs, and other methods which mitigate the security problems inherent in this problem. In particular, user agents which cache credentials are encouraged to provide a readily accessible mechanism for discarding cached credentials under user control.

## 15.7 Proxies and Caching

By their very nature, HTTP proxies are men-in-the-middle, and represent an opportunity for man-in-the-middle attacks. Compromise of the systems on which the proxies run can result in serious security and privacy problems. Proxies have access to security-related information, personal information about individual users and organizations, and proprietary information belonging to users and content providers. A compromised proxy, or a proxy implemented or configured without regard to security and privacy considerations, might be used in the commission of a wide range of potential attacks.

Proxy operators should protect the systems on which proxies run as they would protect any system that contains or transports sensitive information. In particular, log information gathered at proxies often contains highly sensitive personal information, and/or information about organizations. Log information should be carefully guarded, and appropriate guidelines for use developed and followed. (Section 15.1.1).

Caching proxies provide additional potential vulnerabilities, since the contents of the cache represent an attractive target for malicious exploitation. Because cache contents persist after an HTTP request is complete, an attack on the cache can reveal information long after a user believes that the information has been removed from the network. Therefore, cache contents should be protected as sensitive information.

Proxy implementors should consider the privacy and security implications of their design and coding decisions, and of the configuration options they provide to proxy operators (especially the default configuration).

Users of a proxy need to be aware that they are no trustworthier than the people who run the proxy; HTTP itself cannot solve this problem.

The judicious use of cryptography, when appropriate, may suffice to protect against a broad range of security and privacy attacks. Such cryptography is beyond the scope of the HTTP/1.1 specification.

### 15.7.1 Denial of Service Attacks on Proxies

They exist. They are hard to defend against. Research continues. Beware.

## 16 Acknowledgments

This specification makes heavy use of the augmented BNF and generic constructs defined by David H. Crocker for RFC 822 [9]. Similarly, it reuses many of the definitions provided by Nathaniel Borenstein and Ned Freed for MIME [7]. We hope that their inclusion in this specification will help reduce past confusion over the relationship between HTTP and Internet mail message formats.

The HTTP protocol has evolved considerably over the years. It has benefited from a large and active developer community--the many people who have participated on the *www-talk* mailing list--and it is that community which has been most responsible for the success of HTTP and of the World-Wide Web in general. Marc Andreessen, Robert Cailliau, Daniel W. Connolly, Bob Denny, John Franks, Jean-Francois Groff, Phillip M. Hallam-Baker, Håkon W. Lie, Ari Luotonen, Rob McCool, Lou Montulli, Dave Raggett, Tony Sanders, and Marc VanHeyningen deserve special recognition for their efforts in defining early aspects of the protocol.

This document has benefited greatly from the comments of all those participating in the HTTP-WG. In addition to those already mentioned, the following individuals have contributed to this specification:

Gary Adams	Ross Patterson
Harald Tveit Alvestrand	Albert Lunde
Keith Ball	John C. Mallery
Brian Behlendorf	Jean-Philippe Martin-Flatin
Paul Burchard	Mitra
Maurizio Codogno	David Morris
Mike Cowlishaw	Gavin Nicol
Roman Czyborra	Bill Perry
Michael A. Dolan	Jeffrey Perry
David J. Fiander	Scott Powers
Alan Freier	Owen Rees
Marc Hedlund	Luigi Rizzo
Greg Herlihy	David Robinson
Koen Holtman	Marc Salomon
Alex Hopmann	Rich Salz
Bob Jernigan	Allan M. Schiffman
Shel Kaphan	Jim Seidman
Rohit Khare	Chuck Shotton
John Klensin	Eric W. Sink
Martijn Koster	Simon E. Spero
Alexei Kosut	Richard N. Taylor
David M. Kristol	Robert S. Thau
Daniel LaLiberte	Bill (BearHeart) Weinman
Ben Laurie	Francois Yergeau
Paul J. Leach	Mary Ellen Zurko
Daniel DuBois	Josh Cohen

Much of the content and presentation of the caching design is due to suggestions and comments from individuals including: Shel Kaphan, Paul Leach, Koen Holtman, David Morris, and Larry Masinter.

Most of the specification of ranges is based on work originally done by Ari Luotonen and John Franks, with additional input from Steve Zilles.

Thanks to the “cave men” of Palo Alto. You know who you are.

Jim Gettys (the current editor of this document) wishes particularly to thank Roy Fielding, the previous editor of this document, along with John Klensin, Jeff Mogul, Paul Leach, Dave Kristol, Koen Holtman, John Franks, Josh Cohen, Alex Hopmann, Scott Lawrence, and Larry Masinter for their help. And thanks go particularly to Jeff Mogul and Scott Lawrence for performing the “MUST/MAY/SHOULD” audit.

The Apache Group, Anselm Baird-Smith, author of Jigsaw, and Henrik Frystyk implemented RFC 2068 early, and we wish to thank them for the discovery of many of the problems that this document attempts to rectify.

## 17 References

- [1] Alvestrand, H., “Tags for the Identification of Languages” RFC 1766, March 1995.
- [2] Anklesaria, F., McCahill, M., Lindner, P., Johnson, D., Torrey, D., and B. Alberti. “The Internet Gopher Protocol (a distributed document search and retrieval protocol)”, RFC 1436, March 1993.
- [3] Berners-Lee, T., “Universal Resource Identifiers in WWW,” RFC 1630, June 1994.
- [4] Berners-Lee, T., Masinter, L., and M. McCahill. “Uniform Resource Locators (URL),” RFC 1738, December 1994.
- [5] Berners-Lee, T. and D. Connolly. “Hypertext Markup Language - 2.0,” RFC 1866, November 1995.
- [6] Berners-Lee, T., Fielding, R. and H. Frystyk. “Hypertext Transfer Protocol -- HTTP/1.0,” RFC 1945, May 1996.
- [7] Freed, N., and N. Borenstein. “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.” RFC 2045, November 1996.
- [8] Braden, R., “Requirements for Internet Hosts -- Communication Layers,” STD 3, RFC 1123, October 1989.
- [9] D. H. Crocker, “Standard for The Format of ARPA Internet Text Messages,” STD 11, RFC 822, August 1982.
- [10] Davis, F., Kahle, B., Morris, H., Salem, J., Shen, T., Wang, R., Sui, J., and M. Grinbaum, “WAIS Interface Protocol Prototype Functional Specification.” (v1.5), Thinking Machines Corporation, April 1990.
- [11] Fielding, R., “Relative Uniform Resource Locators,” RFC 1808, June 1995.
- [12] Horton, M., and R. Adams. “Standard for Interchange of USENET Messages,” RFC 1036, December 1987.
- [13] Kantor, B. and P. Lapsley. “Network News Transfer Protocol,” RFC 977, February 1986.
- [14] Moore, K., “MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text”, RFC 2047, November 1996.
- [15] Nebel, E., and L. Masinter. “Form-based File Upload in HTML,” RFC 1867, November 1995.
- [16] Postel, J., “Simple Mail Transfer Protocol,” STD 10, RFC 821, August 1982.
- [17] Postel, J., “Media Type Registration Procedure,” RFC 1590, November 1996.
- [18] Postel, J. and J. Reynolds. “File Transfer Protocol,” STD 9, RFC 959, October 1985.
- [19] Reynolds, J. and J. Postel. “Assigned Numbers,” STD 2, RFC 1700, October 1994.

- [20] Sollins, K. and L. Masinter. "Functional Requirements for Uniform Resource Names," RFC 1737, December 1994.
- [21] US-ASCII. Coded Character Set - 7-Bit American Standard Code for Information Interchange. Standard ANSI X3.4-1986, ANSI, 1986.
- [22] ISO-8859. International Standard -- Information Processing -- 8-bit Single-Byte Coded Graphic Character Sets --  
Part 1: Latin alphabet No. 1, ISO-8859-1:1987.  
Part 2: Latin alphabet No. 2, ISO-8859-2, 1987.  
Part 3: Latin alphabet No. 3, ISO-8859-3, 1988.  
Part 4: Latin alphabet No. 4, ISO-8859-4, 1988.  
Part 5: Latin/Cyrillic alphabet, ISO-8859-5, 1988.  
Part 6: Latin/Arabic alphabet, ISO-8859-6, 1987.  
Part 7: Latin/Greek alphabet, ISO-8859-7, 1987.  
Part 8: Latin/Hebrew alphabet, ISO-8859-8, 1988.  
Part 9: Latin alphabet No. 5, ISO-8859-9, 1990.
- [23] Meyers, J., and M. Rose. "The Content-MD5 Header Field," RFC 1864, October 1995.
- [24] Carpenter, B. and Y. Rekhter. "Renumbering Needs Work," RFC 1900, February 1996.
- [25] Deutsch, P., "GZIP file format specification version 4.3,." RFC 1952, May 1996.
- [26] Venkata N. Padmanabhan, and Jeffrey C. Mogul. "Improving HTTP Latency", Computer Networks and ISDN Systems, v. 28, pp. 25-35, Dec. 1995. Slightly revised version of paper in Proc. 2nd International WWW Conference '94: Mosaic and the Web, Oct. 1994, which is available at <http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/mogul/HTTPLatency.html>.
- [27] Joe Touch, John Heidemann, and Katia Obraczka. "Analysis of HTTP Performance", <URL: <http://www.isi.edu/touch/pubs/http-perf96/>>, ISI Research Report ISI/RR-98-463, (original report dated Aug. 1996), USC/Information Sciences Institute, August 1998.
- [28] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis." RFC 1305, March 1992.
- [29] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3." RFC 1951, May 1996.
- [30] S. Spero, "Analysis of HTTP Performance Problems," <http://sunsite.unc.edu/mdma-release/http-prob.html>.
- [31] Deutsch, P. and J. Gailly. "ZLIB Compressed Data Format Specification version 3.3," RFC 1950, May 1996.
- [32] Franks, J., Hallam-Baker, P., Hostetler, J., Leach, P., Luotonen, A., Sink, E., and L. Stewart. "An Extension to HTTP: Digest Access Authentication," RFC 2069, January 1997.
- [33] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, January 1997.
- [34] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, March 1997.
- [35] Troost, R., and Dorner, S., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header," RFC 1806, June 1995.
- [36] Mogul, J.C., Fielding, R., Gettys, J., and H. Frystyk,., "Use and Interpretation of HTTP Version Numbers", RFC 2145, May 1997.

- [37] Palme, J., "Common Internet Message Headers," RFC 2076, February 1997.
- [38] Yergeau, F., "UTF-8, a transformation format of Unicode and ISO-10646," RFC 2279, January 1998.
- [39] Nielsen, H.F., Gettys, J., Baird-Smith, A., Prud'hommeaux, E., Lie, H., and C. Lilley. "Network Performance Effects of HTTP/1.1, CSS1, and PNG," Proceedings of ACM SIGCOMM '97, Cannes France, September 1997.
- [40] Freed, N., and N. Borenstein. "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types." RFC 2046, November 1996.
- [41] Alvestrand, H. T., "IETF Policy on Character Sets and Languages," RFC 2277, BCP 18, January 1998.
- [42] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax and Semantics," RFC 2396, August 1998.
- [43] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., Sink, E., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617, June 1999.
- [44] Luotonen, A., "Tunneling TCP based protocols through Web proxy servers," Work in Progress.
- [45] Palme, J., and A. Hopmann, "MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML)," RFC 2110, March 1997
- [46] Bradner, S., "The Internet Standards Process -- Revision 3," BCP 9, RFC 2026, Harvard University, October 1996.
- [47] Masinter, L., "Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)," RFC 2324, April 1998.
- [48] Freed, N., and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples," RFC 2049, November 1996.
- [49] Troost, R., Dorner, S., and K. Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field," RFC 2183, August 1997.

## 18 Authors' Addresses

### **Roy T. Fielding**

Information and Computer Science  
University of California  
Irvine, CA 92697-3425, USA

Fax: +1 (949) 824-1715  
EMail: fielding@ics.uci.edu

### **James Gettys**

World Wide Web Consortium  
MIT Laboratory for Computer Science  
545 Technology Square  
Cambridge, MA 02139, USA

EMail: jg@w3.org

### **Jeffrey C. Mogul**

Western Research Laboratory  
Compaq Computer Corporation  
250 University Avenue  
Palo Alto, California, 94305, USA



E-Mail: mogul@wrl.dec.com

**Henrik Frystyk Nielsen**  
World Wide Web Consortium  
MIT Laboratory for Computer Science  
545 Technology Square  
Cambridge, MA 02139, USA

E-Mail: frystyk@w3.org

**Larry Masinter**  
Xerox Corporation  
3333 Coyote Hill Road  
Palo Alto, CA 94034, USA

E-Mail: masinter@parc.xerox.com

**Paul J. Leach**  
Microsoft Corporation  
1 Microsoft Way  
Redmond, WA 98052, USA

E-Mail: paulle@microsoft.com

**Tim Berners-Lee**  
Director, World Wide Web Consortium  
MIT Laboratory for Computer Science  
545 Technology Square  
Cambridge, MA 02139, USA

Fax: +1 (617) 258 8682  
E-Mail: timbl@w3.org

## 19 Appendices

### 19.1 Internet Media Type message/http and application/http

In addition to defining the HTTP/1.1 protocol, this document serves as the specification for the Internet media type "message/http" and "application/http". The message/http type can be used to enclose a single HTTP request or response message, provided that it obeys the MIME restrictions for all "message" types regarding line length and encodings. The application/http type can be used to enclose a pipeline of one or more HTTP request or response messages (not intermixed). The following is to be registered with IANA [17].

```
Media Type name:      message
Media subtype name:   http
Required parameters:  none
Optional parameters:  version, msgtype
  version: The HTTP-Version number of the enclosed message
            (e.g., "1.1"). If not present, the version can be
            determined from the first line of the body.
  msgtype: The message type -- "request" or "response". If not
            present, the type can be determined from the first
            line of the body.
Encoding considerations: only "7bit", "8bit", or "binary" are
                           permitted
Security considerations: none
```

```

Media Type name:      application
Media subtype name:   http
Required parameters:  none
Optional parameters:  version, msgtype
  version: The HTTP-Version number of the enclosed messages
           (e.g., "1.1"). If not present, the version can be
           determined from the first line of the body.
  msgtype: The message type -- "request" or "response". If not
           present, the type can be determined from the first
           line of the body.
Encoding considerations: HTTP messages enclosed by this type
                           are in "binary" format; use of an appropriate
                           Content-Transfer-Encoding is required when
                           transmitted via E-mail.
Security considerations: none

```

## 19.2 Internet Media Type multipart/byteranges

When an HTTP 206 (Partial Content) response message includes the content of multiple ranges (a response to a request for multiple non-overlapping ranges), these are transmitted as a multipart message-body. The media type for this purpose is called "multipart/byteranges".

The multipart/byteranges media type includes two or more parts, each with its own Content-Type and Content-Range fields. The required boundary parameter specifies the boundary string used to separate each body-part.

```

Media Type name:      multipart
Media subtype name:   byteranges
Required parameters:  boundary
Optional parameters:  none
Encoding considerations: only "7bit", "8bit", or "binary" are
                           permitted
Security considerations: none

```

For example:

```

HTTP/1.1 206 Partial Content
Date: Wed, 15 Nov 1995 06:25:24 GMT
Last-Modified: Wed, 15 Nov 1995 04:58:08 GMT
Content-type: multipart/byteranges; boundary=THIS_STRING_SEPARATES

--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 500-999/8000

...the first range...
--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 7000-7999/8000

...the second range
--THIS_STRING_SEPARATES--

```

Notes:

- 1) Additional CRLFs may precede the first boundary string in the entity.
- 2) Although RFC 2046 [40] permits the boundary string to be quoted, some existing implementations handle a quoted boundary string incorrectly.

3) A number of browsers and servers were coded to an early draft of the byteranges specification to use a media type of multipart/x-byteranges, which is almost, but not quite compatible with the version documented in HTTP/1.1.

### 19.3 Tolerant Applications

Although this document specifies the requirements for the generation of HTTP/1.1 messages, not all applications will be correct in their implementation. We therefore recommend that operational applications be tolerant of deviations whenever those deviations can be interpreted unambiguously.

Clients **SHOULD** be tolerant in parsing the `Status-Line` and servers tolerant when parsing the `Request-Line`. In particular, they **SHOULD** accept any amount of `SP` or `HT` characters between fields, even though only a single `SP` is required.

The line terminator for message-header fields is the sequence `CR LF`. However, we recommend that applications, when parsing such headers, recognize a single `LF` as a line terminator and ignore the leading `CR`.

The character set of an entity-body **SHOULD** be labeled as the lowest common denominator of the character codes used within that body, with the exception that not labeling the entity is preferred over labeling the entity with the labels `US-ASCII` or `ISO-8859-1`. See section 3.7.1 and 3.4.1.

Additional rules for requirements on parsing and encoding of dates and other potential problems with date encodings include:

- HTTP/1.1 clients and caches **SHOULD** assume that an RFC-850 date which appears to be more than 50 years in the future is in fact in the past (this helps solve the “year 2000” problem).
- An HTTP/1.1 implementation **MAY** internally represent a parsed `Expires` date as earlier than the proper value, but **MUST NOT** internally represent a parsed `Expires` date as later than the proper value.
- All expiration-related calculations **MUST** be done in GMT. The local time zone **MUST NOT** influence the calculation or comparison of an age or expiration time.
- If an HTTP header incorrectly carries a date value with a time zone other than GMT, it **MUST** be converted into GMT using the most conservative possible conversion.

### 19.4 Differences Between HTTP Entities and RFC 2045 Entities

HTTP/1.1 uses many of the constructs defined for Internet Mail (RFC 822 [9]) and the Multipurpose Internet Mail Extensions (MIME [7]) to allow entities to be transmitted in an open variety of representations and with extensible mechanisms. However, RFC 2045 discusses mail, and HTTP has a few features that are different from those described in RFC 2045. These differences were carefully chosen to optimize performance over binary connections, to allow greater freedom in the use of new media types, to make date comparisons easier, and to acknowledge the practice of some early HTTP servers and clients.

This appendix describes specific areas where HTTP differs from RFC 2045. Proxies and gateways to strict MIME environments **SHOULD** be aware of these differences and provide the appropriate conversions where necessary. Proxies and gateways from MIME environments to HTTP also need to be aware of the differences because some conversions might be required.

#### 19.4.1 MIME-Version

HTTP is not a MIME-compliant protocol. However, HTTP/1.1 messages **MAY** include a single `MIME-Version` general-header field to indicate what version of the MIME protocol was used to construct the message. Use of the `MIME-Version` header field indicates that the message is in full compliance with the MIME protocol (as defined in RFC 2045[7]). Proxies/gateways are responsible for ensuring full compliance (where possible) when exporting HTTP messages to strict MIME environments.

```
MIME-Version = "MIME-Version" ":" 1*DIGIT "." 1*DIGIT
```

MIME version "1.0" is the default for use in HTTP/1.1. However, HTTP/1.1 message parsing and semantics are defined by this document and not the MIME specification.

#### 19.4.2 Conversion to Canonical Form

RFC 2045 [7] requires that an Internet mail entity be converted to canonical form prior to being transferred, as described in section 4 of RFC 2049 [48]. Section 3.7.1 of this document describes the forms allowed for subtypes of the "text" media type when transmitted over HTTP. RFC 2046 requires that content with a type of "text" represent line breaks as CRLF and forbids the use of CR or LF outside of line break sequences. HTTP allows CRLF, bare CR, and bare LF to indicate a line break within text content when a message is transmitted over HTTP.

Where it is possible, a proxy or gateway from HTTP to a strict MIME environment SHOULD translate all line breaks within the text media types described in section 3.7.1 of this document to the RFC 2049 canonical form of CRLF. Note, however, that this might be complicated by the presence of a `Content-Encoding` and by the fact that HTTP allows the use of some character sets which do not use octets 13 and 10 to represent CR and LF, as is the case for some multi-byte character sets.

Implementors should note that conversion will break any cryptographic checksums applied to the original content unless the original content is already in canonical form. Therefore, the canonical form is recommended for any content that uses such checksums in HTTP.

#### 19.4.3 Conversion of Date Formats

HTTP/1.1 uses a restricted set of date formats (section 3.3.1) to simplify the process of date comparison. Proxies and gateways from other protocols SHOULD ensure that any `Date` header field present in a message conforms to one of the HTTP/1.1 formats and rewrite the date if necessary.

#### 19.4.4 Introduction of Content-Encoding

RFC 2045 does not include any concept equivalent to HTTP/1.1's `Content-Encoding` header field. Since this acts as a modifier on the media type, proxies and gateways from HTTP to MIME-compliant protocols MUST either change the value of the `Content-Type` header field or decode the entity-body before forwarding the message. (Some experimental applications of `Content-Type` for Internet mail have used a media-type parameter of `" ; conversions=<content-coding> "` to perform a function equivalent to `Content-Encoding`. However, this parameter is not part of RFC 2045.)

#### 19.4.5 No Content-Transfer-Encoding

HTTP does not use the `Content-Transfer-Encoding` (CTE) field of RFC 2045. Proxies and gateways from MIME-compliant protocols to HTTP MUST remove any non-identity CTE ("quoted-printable" or "base64") encoding prior to delivering the response message to an HTTP client.

Proxies and gateways from HTTP to MIME-compliant protocols are responsible for ensuring that the message is in the correct format and encoding for safe transport on that protocol, where "safe transport" is defined by the limitations of the protocol being used. Such a proxy or gateway SHOULD label the data with an appropriate `Content-Transfer-Encoding` if doing so will improve the likelihood of safe transport over the destination protocol.

#### 19.4.6 Introduction of Transfer-Encoding

HTTP/1.1 introduces the `Transfer-Encoding` header field (section 14.41). Proxies/gateways MUST remove any transfer-coding prior to forwarding a message via a MIME-compliant protocol.

A process for decoding the "chunked" transfer-coding (section 3.6) can be represented in pseudo-code as:

```
length := 0
read chunk-size, chunk-extension (if any) and CRLF
while (chunk-size > 0) {
```