

1 SPENCER HOSIE (CA Bar No. 101777)
shosie@hosielaw.com
2 BRUCE WECKER (CA Bar No. 078530)
bwecker@hosielaw.com
3 GEORGE F. BISHOP (CA Bar No. 89205)
gbishop@hosielaw.com
4 HOSIE RICE LLP
188 The Embarcadero, Suite 750
5 San Francisco, CA 94105
6 (415) 247-6000 Tel.
(415) 247-6001 Fax

7 *Attorneys for Plaintiff*
8 **BACKWEB TECHNOLOGIES, LTD.**

10 UNITED STATES DISTRICT COURT
11 FOR THE NORTHERN DISTRICT OF CALIFORNIA
12 SAN FRANCISCO DIVISION

13 BACKWEB TECHNOLOGIES, LTD.

14 Plaintiff,

15 v.

16 MICROSOFT CORPORATION,

17 Defendant.
18
19
20
21
22
23
24
25
26
27
28

ORIGINAL
FILED
MAR 20 2009
RICHARD W. WIEKING
CLERK, U.S. DISTRICT COURT,
NORTHERN DISTRICT OF CALIFORNIA

HRL

CV 09 Case No. 1224

ORIGINAL COMPLAINT AND
DEMAND FOR JURY TRIAL

1 Plaintiff BackWeb Technologies, LTD. ("BackWeb" or "Plaintiff") hereby files its
2 complaint against Defendant Microsoft Corporation ("Microsoft or "Defendant"), for patent
3 infringement. For its complaint, Plaintiff alleges, on personal knowledge as to its own acts
4 and on information and belief as to all other matters, as follows:

5 **PARTIES**

6 1. BackWeb is a corporation organized under the laws of the State of Israel, and
7 has its principal place of business in Rosh Ha'ayin, Israel. BackWeb's wholly owned
8 subsidiary, BackWeb Technologies, Inc. maintains its principal office in San Jose,
9 California. BackWeb is and at all pertinent times was the assignee and owner of the patents
10 at issue in this case.

12 2. Defendant Microsoft, on information and belief, is a corporation organized
13 under the laws of the State of Washington. Microsoft is doing business in Washington, and
14 has its principal place of business in Redmond, Washington.

15 **JURISDICTION AND VENUE**

16 3. This complaint asserts a cause of action for patent infringement under the
17 Patent Act, 35 U.S.C. § 271. This Court has subject matter jurisdiction over this matter by
18 virtue of 28 U.S.C. § 1338(a). Venue is proper in this Court by virtue of 28 U.S.C. § 1391(b)
19 and (c) and 28 U.S.C. § 1400(b), in that Defendant Microsoft may be found in this district,
20 has committed acts of infringement in this district, and a substantial part of the events or
21 omissions giving rise to the claim occurred and a substantial part of property that is the
22 subject of the action is situated in this district.

23 4. This Court has personal jurisdiction over Defendant Microsoft because it has a
24 place of business in, and provides infringing products and services in, the Northern District
25 of California.
26
27
28

1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8

5. Pursuant to Civil LR 3-2(c), this case should be subject to district-wide assignment because it is an Intellectual Property Action.

BACKGROUND

The BackWeb Patents

6. Plaintiff owns a patent, U.S. Patent No. 5,913,040 (“’040 Patent”), issued on June 15, 1999, to inventors Yuval Rakavy and Eli Barkat. A true and correct copy of the ’040 Patent is attached as Exhibit “A” and is incorporated herein by reference. Plaintiff is the legal and rightful owner of the ’040 Patent.

7. The '040 Patent contains fourteen (14) patent claims covering unique and novel methods and processes for transmitting digital information in background mode over a communications link between a computer network and a local computer and throttling the transfer speed to create minimal interference with other processes communicating over the communications link. The digital information described in the patent could be in a variety of forms, including, but not limited to, news, weather, stock quotes, sports scores, software updates or trip reservation information.

8. Plaintiff also owns two continuation patents, U.S. Patent No. 6,317,789 (“’789 Patent”) and U.S. Patent No. 6,539,429 (“’429 Patent”), issued on November 13, 2001 and March 25, 2003, respectively. A true and correct copy of the ’789 Patent is attached as Exhibit “B” and is incorporated herein by reference. Plaintiff is the legal and rightful owner of the ’789 Patent. A true and correct copy of the ’429 Patent is attached as Exhibit “C” and is incorporated herein by reference. Plaintiff is the legal and rightful owner of the ’429 Patent. The two continuation patents contain twenty-eight (28) patent claims covering unique and novel methods, processes and systems for transmitting digital information in

1 background mode over a communications link between a computer network and a local
2 computer with minimal interference with other processes communicating over the
3 communications link. Plaintiff's three patents in this patent family will be referred to herein
4 as its Transparent Update Patents.

5 9. Plaintiff owns a patent, U.S. Patent No. 6,374,289 ("289 Patent"), issued on
6 April 16, 2002, to inventors Hubert Delaney, Adi Ruppin, Lior Hass, and Ofer Faigon. The
7 '289 Patent contains twenty-three (23) patent claims covering a unique and novel method for
8 distributing data packages across a hybrid peer-to-peer network, the network featuring a
9 server, a plurality of peer clients attached to the network, and lists of data packages
10 identifying the location of the data package in at least one of the plurality of peer clients, for
11 transmission. A true and correct copy of the '289 Patent is attached as Exhibit "D" and is
12 incorporated herein by reference. Plaintiff is the legal and rightful owner of the '289 Patent.
13

14 **Microsoft's Infringing Goods and Services**

15 10. In 2001, Microsoft introduced a technology that it calls Background
16 Intelligent Transfer Service (BITS). BITS transfers files in the foreground or background,
17 throttles the transfers to preserve the responsiveness of other network applications, and
18 automatically resumes file transfers after network disconnects and machine restarts. In 2007,
19 Microsoft began the commercial distribution of version 3.0 of BITS, that adds the capability
20 of transferring files in a peer to peer networking fashion. Microsoft manufactures, uses and
21 sells products that infringe the three Transparent Update Patents. With the introduction of
22 BITS Ver. 3.0, Microsoft has also infringed BackWeb's '289 Patent.
23
24
25
26
27
28

COUNT I
(Patent Infringement)

11. Plaintiff incorporates by reference the allegations of paragraphs 1 through 10 above.

12. BackWeb is the owner of the '040, '789, and '429 patents.

13. Microsoft has infringed and is still infringing the Transparent Update Patents, by, without authority, consent, right or license, and in direct infringement of the patents, making, using, offering for sale and/or selling digital information transfer products using the methods, processes and apparatuses claimed in the patents in this country. This conduct constitutes infringement under 35 U.S.C. § 271(a).

14. In addition, Microsoft has infringed and is still infringing the Transparent Update Patents in this country, through, *inter alia*, its active inducement of others to make, use, and/or sell the systems, products and methods claimed in one or more claims of the patents. This conduct constitutes infringement under 35 U.S.C. § 271(b).

15. In addition, Microsoft has infringed and is still infringing the Transparent Update Patents in this country through, *inter alia*, providing and selling goods and services including products designed for use in practicing one or more claims of the Transparent Update Patents, where the goods and services constitute a material part of the invention and are not staple articles of commerce, and which have no use other than infringing one or more claims of the Transparent Update Patents. Microsoft has committed these acts with knowledge that the goods and services it provides are specially made for use in a manner that directly infringes the Transparent Update Patents. This conduct constitutes infringement under 35 U.S.C. § 271(c).

1 16. Microsoft's infringing conduct is unlawful and willful. Microsoft's willful
2 conduct makes this an exceptional case as provided in 35 U.S.C. § 285.

3 17. As a result of Microsoft's infringement, Plaintiff has been damaged, and will
4 continue to be damaged, until they are enjoined from further acts of infringement.

5 18. Microsoft will continue to infringe the Transparent Update Patents unless
6 enjoined by this Court. Plaintiff faces real, substantial and irreparable damage and injury of
7 a continuing nature from Microsoft's infringement for which Plaintiff has no adequate
8 remedy at law.
9

10 **COUNT II**
11 **(Patent Infringement)**

12 19. Plaintiff incorporates by reference the allegations of paragraphs 1 through 10
13 above.

14 20. BackWeb is the owner of the '289 Patent.

15 21. Microsoft has infringed and is still infringing the '289 Patent, by, without
16 authority, consent, right or license, and in direct infringement of the patents, making, using,
17 offering for sale and/or selling digital information transfer products using the methods,
18 processes and apparatuses claimed in the patent in this country. This conduct constitutes
19 infringement under 35 U.S.C. § 271(a).
20

21 22. In addition, Microsoft has infringed and is still infringing the '289 Patent in
22 this country, through, *inter alia*, its active inducement of others to make, use, and/or sell the
23 systems, products and methods claimed in one or more claims of the patent. This conduct
24 constitutes infringement under 35 U.S.C. § 271(b).

25 23. In addition, Microsoft has infringed and is still infringing the '289 Patent in
26 this country through, *inter alia*, providing and selling goods and services including products
27
28

1 designed for use in practicing one or more claims of the '289 Patent, where the goods and
2 services constitute a material part of the invention and are not staple articles of commerce,
3 and which have no use other than infringing one or more claims of the '289 Patent.

4 Microsoft has committed these acts with knowledge that the goods and services it provides
5 are specially made for use in a manner that directly infringes the '289 Patent. This conduct
6 constitutes infringement under 35 U.S.C. § 271(c).

7
8 24. Microsoft's infringing conduct is unlawful and willful. Defendant Microsoft's
9 willful conduct makes this an exceptional case as provided in 35 U.S.C. § 285.

10 25. As a result of Microsoft's infringement, Plaintiff has been damaged, and will
11 continue to be damaged, until they are enjoined from further acts of infringement.

12 26. Microsoft will continue to infringe the '289 Patent unless enjoined by this
13 Court. Plaintiff faces real, substantial and irreparable damage and injury of a continuing
14 nature from Defendant Microsoft's infringement for which Plaintiff has no adequate remedy
15 at law.
16

17 WHEREFORE, Plaintiff prays:

18 (a) That this Court find Defendant has committed acts of patent infringement
19 under the Patent Act, 35 U.S.C. § 271;

20 (b) That this Court enter judgment that:

21 (i) The Transparent Update Patents are valid and enforceable;

22 (ii) Defendant Microsoft has willfully infringed the Transparent
23 Update Patents;

24 (iii) The '289 Patent is valid and enforceable; and

25 (iv) Defendant Microsoft has willfully infringed the '289 Patent;
26
27
28

1 (c) That this Court issue a preliminary and final injunction enjoining
2 Microsoft, its officers, agents, servants, employees and attorneys, and any other person in
3 active concert or participation with them, from continuing the acts herein complained of,
4 and more particularly, that Microsoft and such other persons be permanently enjoined
5 and restrained from further infringing the Transparent Update Patents;

6 (d) That this Court issue a preliminary and final injunction enjoining
7 Microsoft, its officers, agents, servants, employees and attorneys, and any other person in
8 active concert or participation with them, from continuing the acts herein complained of,
9 and more particularly, that Microsoft and such other persons be permanently enjoined
10 and restrained from further infringing the '289 Patent;

11 (e) That this Court award Plaintiff the damages to which it is entitled due to
12 Defendant Microsoft's patent infringement, with both pre-judgment and post-judgment
13 interest;

14 (f) That Defendant Microsoft's infringement of the BackWeb Patents be
15 adjudged willful and that the damages to Plaintiff be increased by three times the amount
16 found or assessed pursuant to 35 U.S.C. § 284;

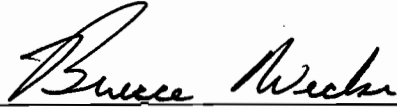
17 (g) That this be adjudged an exceptional case and that Plaintiff be awarded its
18 attorney's fees in this action pursuant to 35 U.S.C. § 285;

19 (h) That this Court award Plaintiff its costs and disbursements in this civil
20 action, including reasonable attorney's fees; and

21 (i) That this Court grant Plaintiff such other and further relief, in law or in
22 equity, both general and special, to which it may be entitled.
23
24
25
26
27
28

1 Dated: March 20, 2009

Respectfully submitted,

2
3 

4 SPENCER HOSIE (CA Bar No. 101777)

5 shosie@hosielaw.com

6 BRUCE WECKER (CA Bar No. 078530)

bwecker@hosielaw.com

7 GEORGE F. BISHOP (CA Bar No. 89205)

gbishop@hosielaw.com

8 HOSIE RICE LLP

188 The Embarcadero, Suite 750

9 San Francisco, CA 94105

(415) 247-6000 Tel.

10 (415) 247-6001 Fax

11 *Attorneys for Plaintiff*

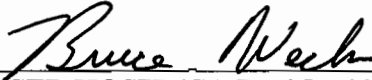
12 *BACKWEB TECHNOLOGIES, LTD.*

DEMAND FOR JURY TRIAL

Plaintiff, by its undersigned attorneys, demands a trial by jury on all issues so triable.

Dated: March 20, 2009

Respectfully submitted,



SPENCER HOSIE (CA Bar No. 101777)

shosie@hosielaw.com

BRUCE WECKER (CA Bar No. 078530)

bwecker@hosielaw.com

GEORGE F. BISHOP (CA Bar No. 89205)

gbishop@hosielaw.com

HOSIE RICE LLP

188 The Embarcadero, Suite 750

San Francisco, CA 94105

(415) 247-6000 Tel.

(415) 247-6001 Fax

Attorneys for Plaintiff

BACKWEB TECHNOLOGIES, LTD.

EXHIBIT A



US005913040A

United States Patent [19]
Rakavy et al.

[11] **Patent Number:** **5,913,040**
[45] **Date of Patent:** **Jun. 15, 1999**

[54] **METHOD AND APPARATUS FOR TRANSMITTING AND DISPLAYING INFORMATION BETWEEN A REMOTE NETWORK AND A LOCAL COMPUTER**

[75] Inventors: **Yuval Rakavy; Eli Barkat**, both of Jerusalem, Israel

[73] Assignee: **Backweb Ltd.**, Jerusalem, Israel

[21] Appl. No.: **08/517,666**
[22] Filed: **Aug. 22, 1995**

[51] **Int. Cl.⁶** **G06F 13/00**
[52] **U.S. Cl.** **395/200.62; 395/200.47; 395/200.54; 370/229**
[58] **Field of Search** **395/200.01, 200.07, 395/200.11, 200.13, 200.3, 200.47, 200.54, 200.62, 200.65; 370/229, 230, 232, 235, 234; 348/10, 522**

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,719,567	1/1988	Whittington et al.	364/DIG. 1
4,799,146	1/1989	Chauvel .	
5,099,420	3/1992	Baxlow et al.	395/299
5,105,184	4/1992	Pirani et al. .	
5,165,012	11/1992	Crandall et al. .	
5,220,564	6/1993	Tuch et al.	370/94.1
5,283,639	2/1994	Each et al. .	
5,285,442	2/1994	Iwamura et al.	370/234
5,305,195	4/1994	Murphy .	
5,313,455	5/1994	van der Wal et al.	370/232
5,319,455	6/1994	Hoarty et al. .	
5,321,740	6/1994	Gregorek et al. .	
5,347,632	9/1994	Filepp et al. .	
5,355,501	10/1994	Gross et al. .	
5,361,091	11/1994	Hoarty et al. .	

5,390,172	2/1995	Kuang .	
5,412,416	5/1995	Nemirofsky .	
5,455,826	10/1995	Ozveren et al.	370/232
5,488,609	1/1996	Hluchyj et al.	370/84
5,504,744	4/1996	Adams et al.	370/232
5,555,377	9/1996	Christensen et al.	395/200.11
5,600,364	2/1997	Hendricks et al.	348/1
5,604,542	2/1997	Dedrick	348/552
5,675,742	10/1997	Jain et al.	395/200.56

OTHER PUBLICATIONS

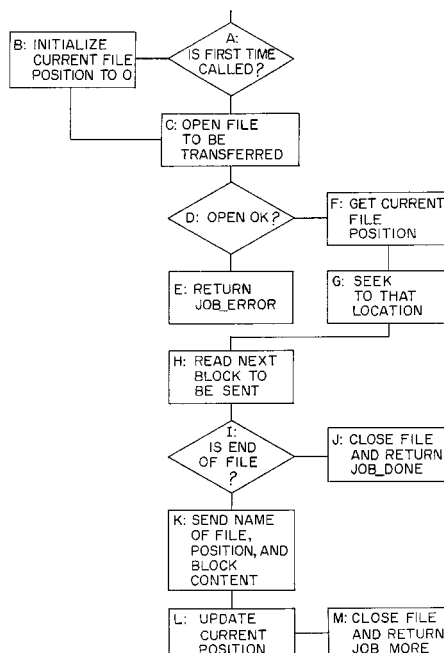
J. Rigdon, Comming Soon to the Internet: Tools to Add Glitz to the Web's Offerings, Wall Street Journal, Aug. 15, 1995.
J. Martin, TCP/IP Networking, PTR Prentice Hall, 1994 (pp. 147-148).

Primary Examiner—Meng-Ai T. An
Assistant Examiner—Walter D. Davis, Jr.
Attorney, Agent, or Firm—Skadden, Arps et al.

[57] **ABSTRACT**

Methods and apparatus are provided for selecting advertisements and other information from a computer network database based on user defined preferences and transmitting the selected advertisement in background mode over a communications link between the computer network and a local computer with minimal interference with other processes communicating over the communications link. This method includes monitoring the communications link and transmitting portions of the advertisement when the communications link line utilization is below a preestablished threshold. Methods and apparatus are also provided for displaying or otherwise presenting the selected advertisements on the user's computer. Additional methods and apparatus are provided for selecting and presenting information stored on a local storage media based on user defined preferences.

14 Claims, 8 Drawing Sheets



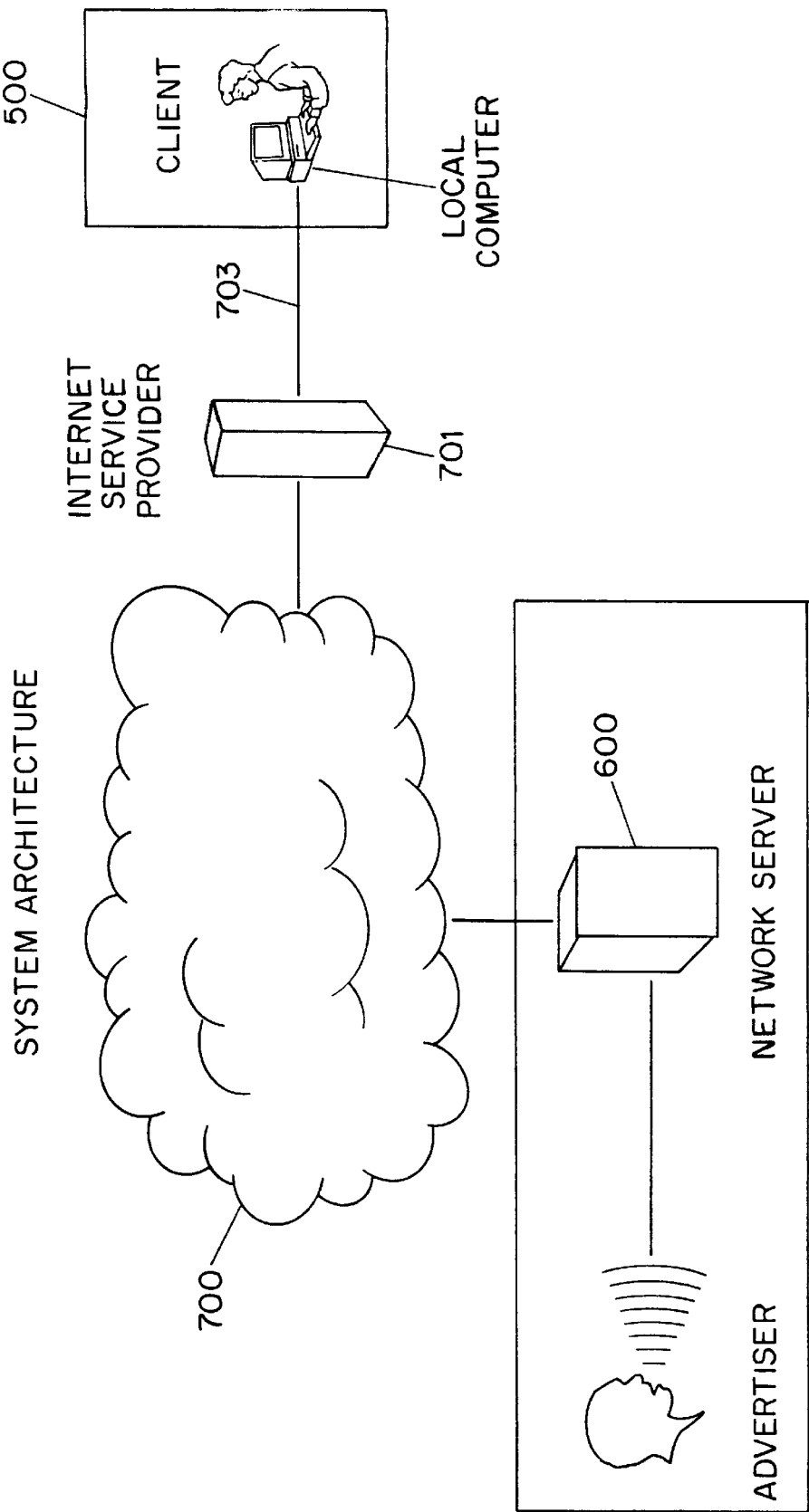


FIG. 1

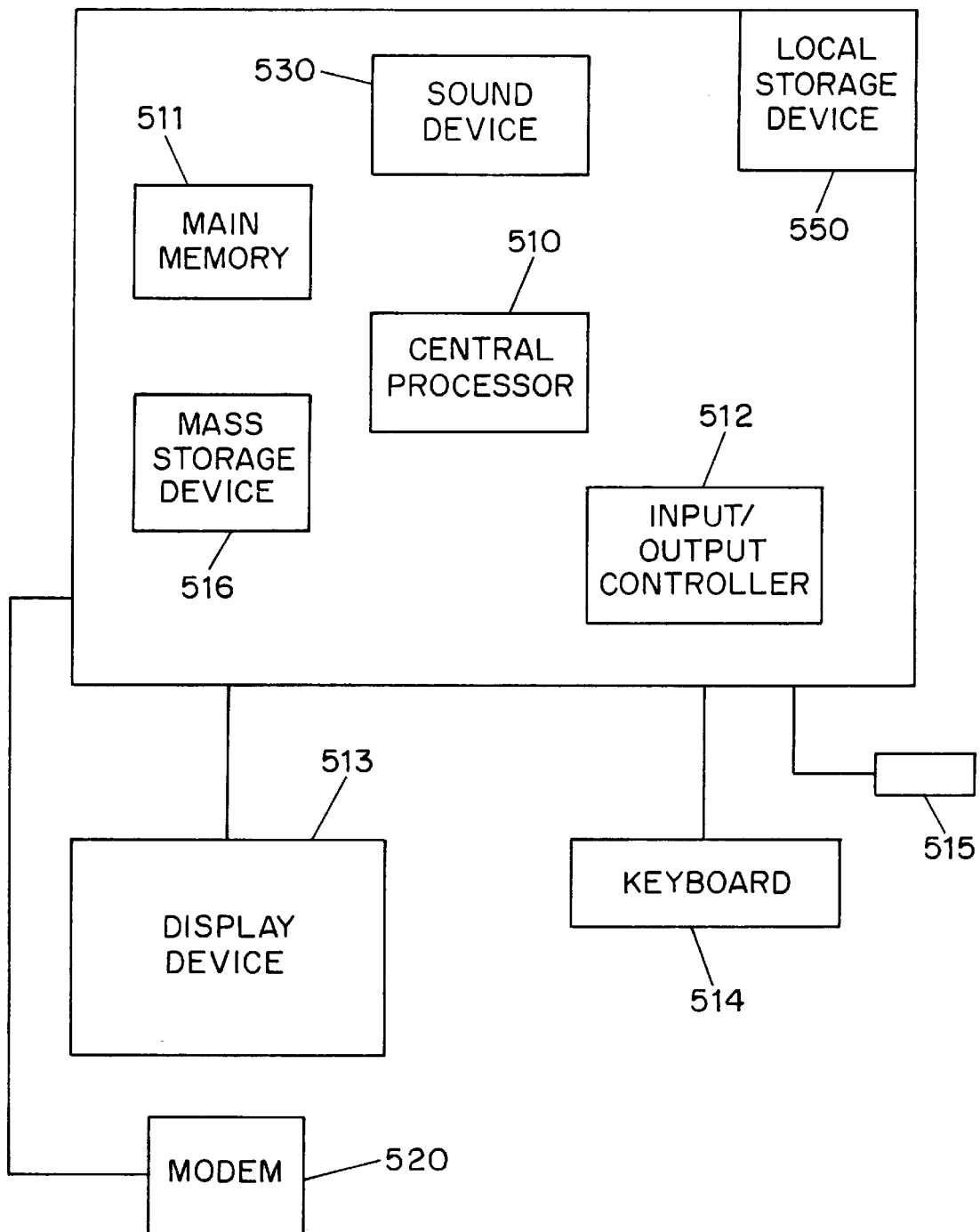


FIG. 2

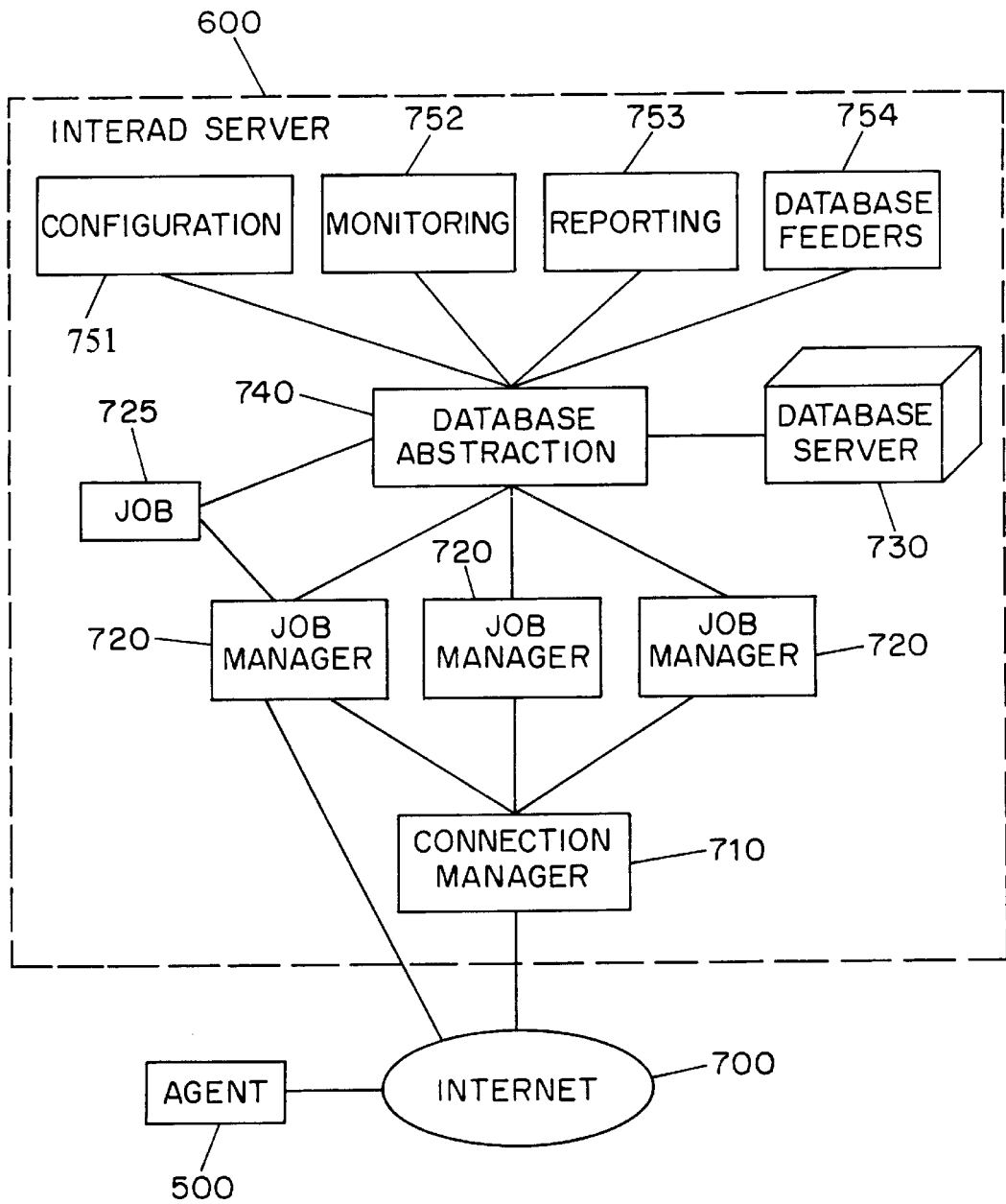


FIG. 3

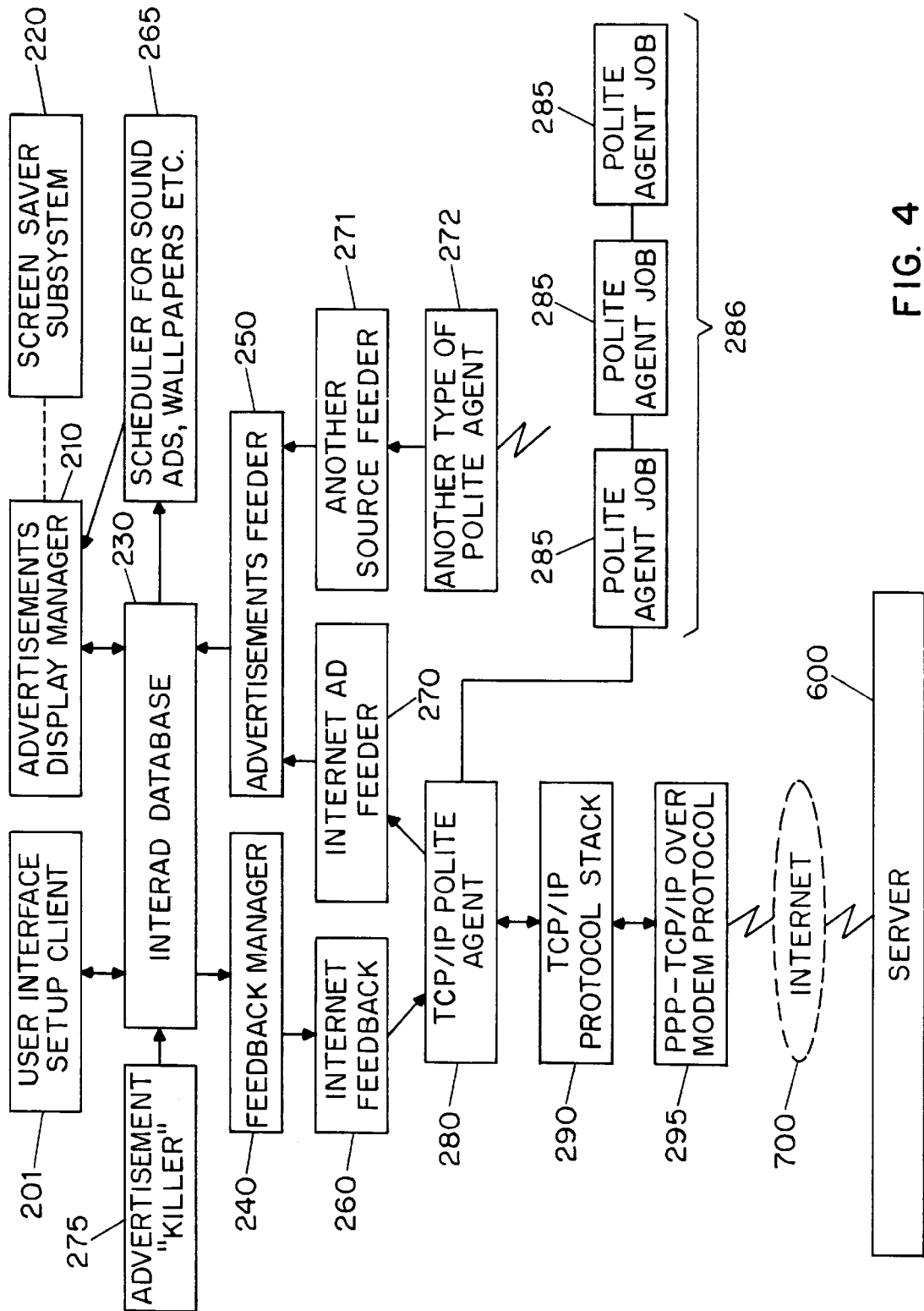


FIG. 4

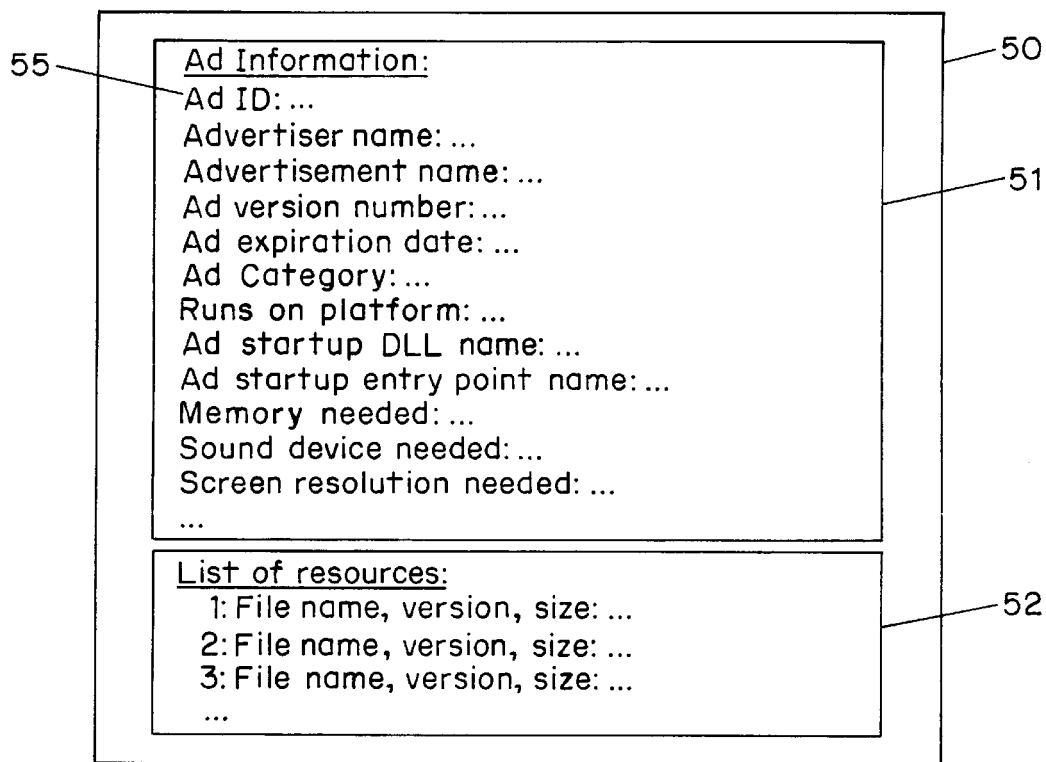


FIG. 5

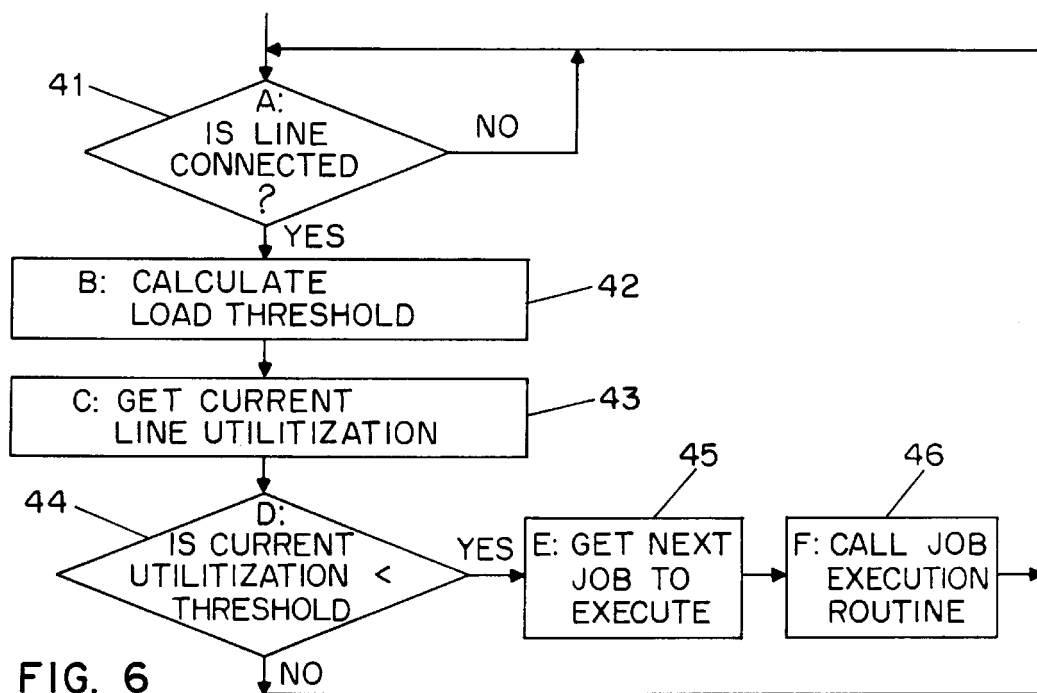


FIG. 6

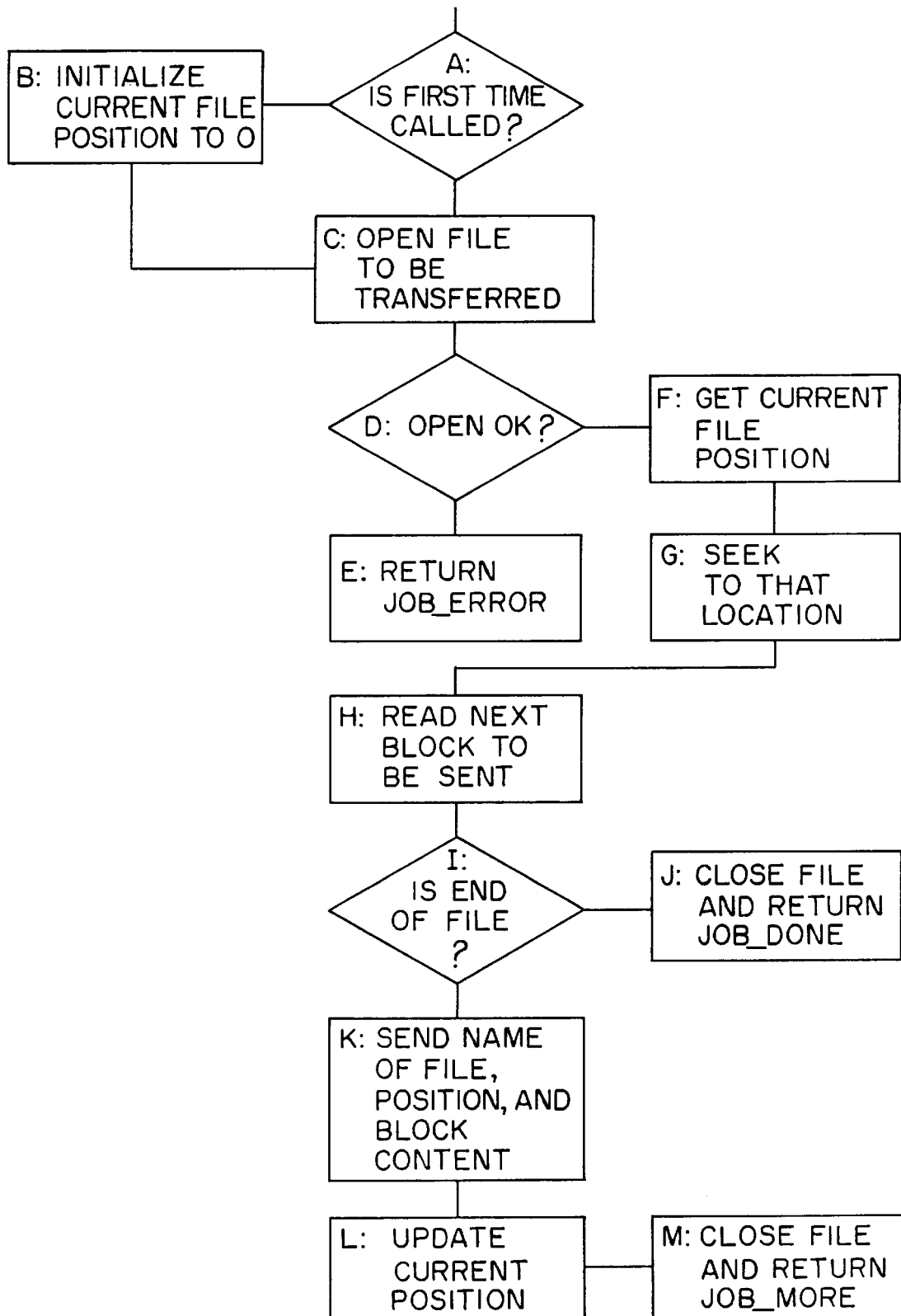


FIG. 7

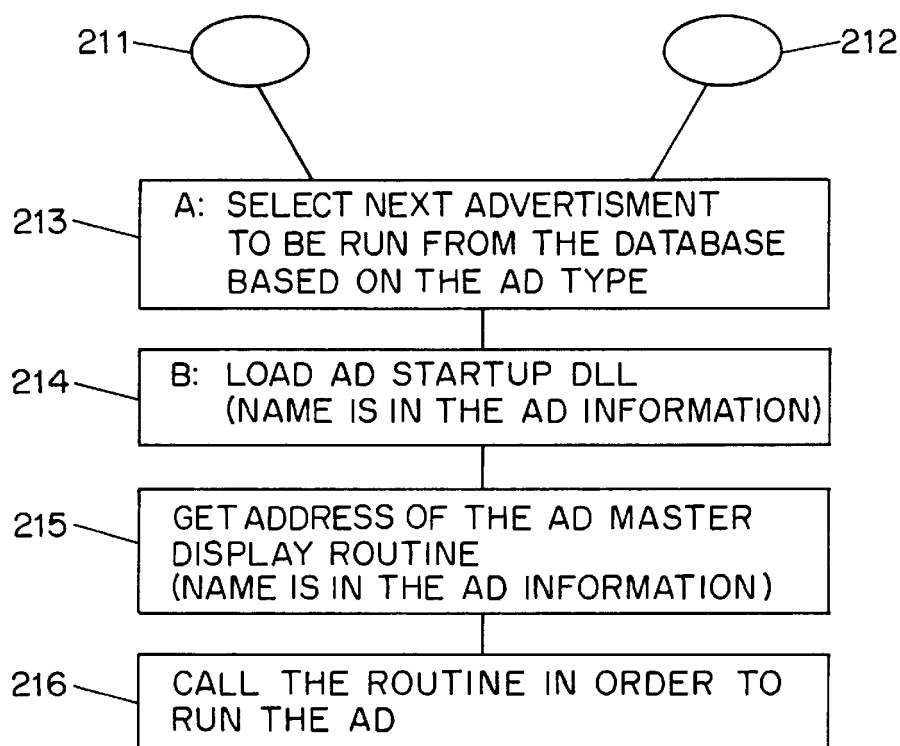


FIG. 8

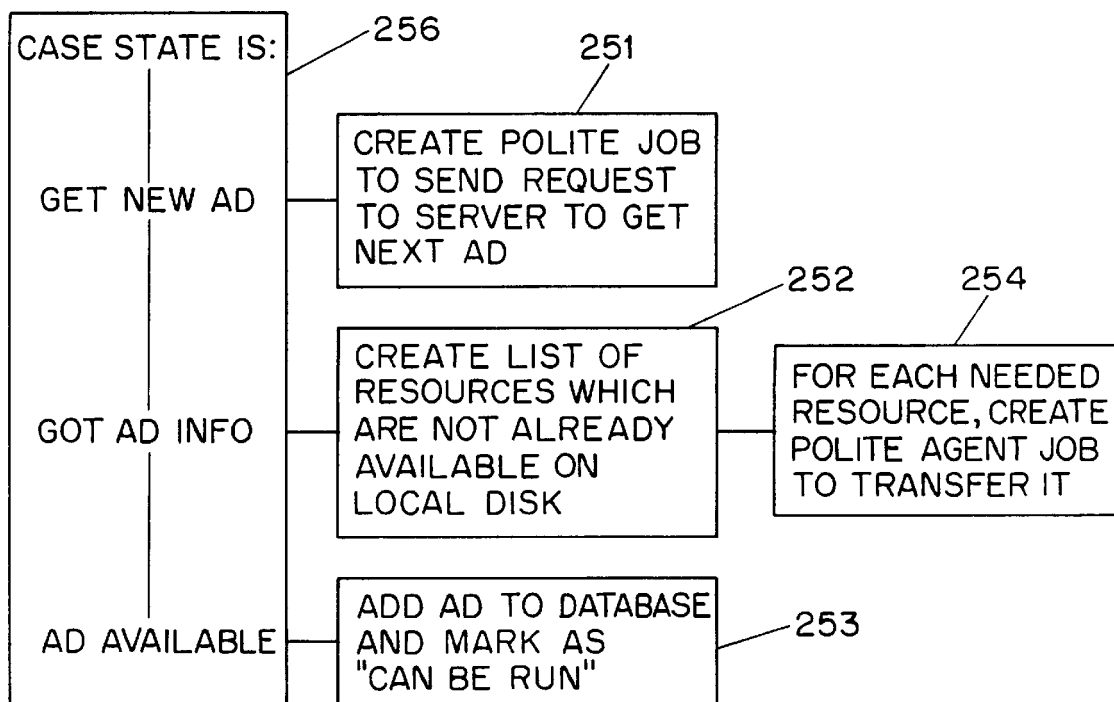


FIG. 9

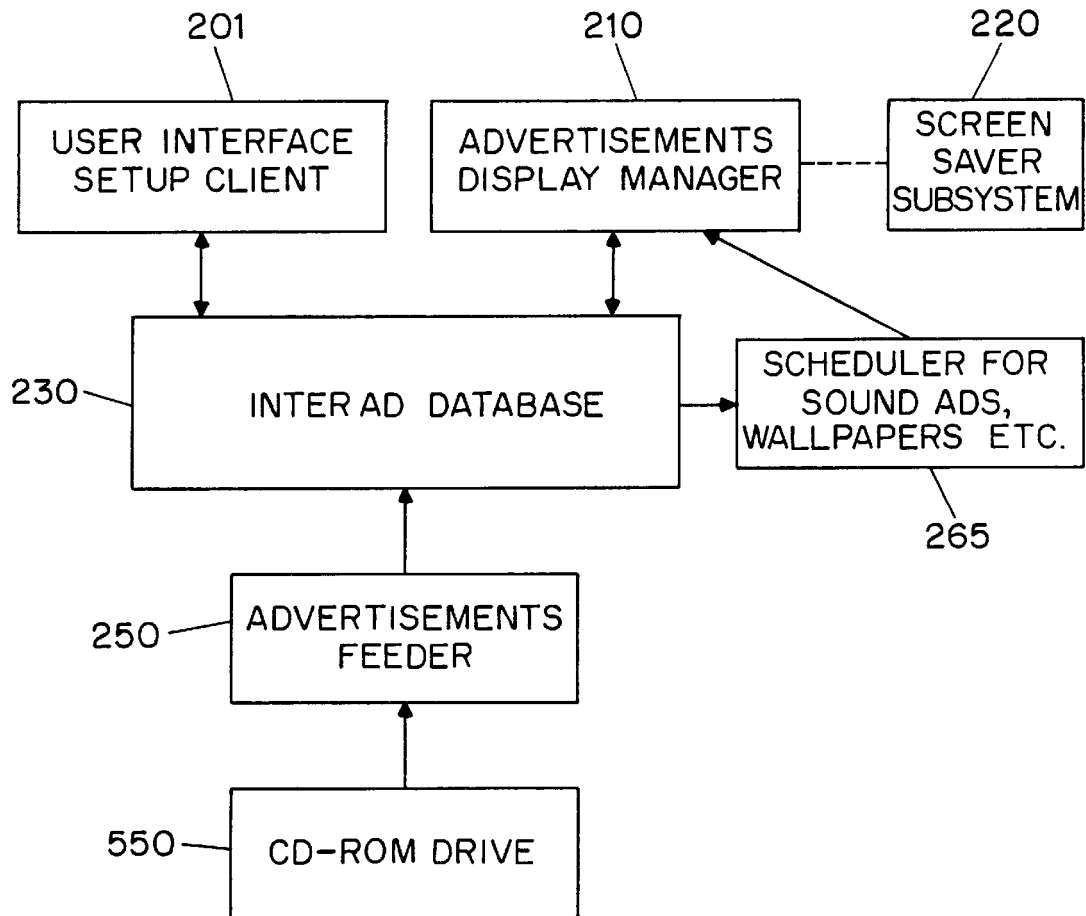


FIG. 10

METHOD AND APPARATUS FOR TRANSMITTING AND DISPLAYING INFORMATION BETWEEN A REMOTE NETWORK AND A LOCAL COMPUTER

FIELD OF THE INVENTION

This invention relates generally to advertisement computer display systems and more particularly to a method and system for displaying advertisements and other information on a computer based on general user selected criteria and transmitting such information from a remote network to the local computer.

BACKGROUND OF THE PRESENT INVENTION

There are two major forms of advertising which are currently being employed on the Internet and commercial on-line services. One form is the use of a small advertisement on WEB pages which are commonly accessed. For example, a portion of the screen display for WEB pages used to access Internet searches may include a corporate logo or other advertisement material. Typical of this style of advertising is the Netscape™ Internet Browser software available from Netscape Communications Corporation of Mountain View California, which presents a box containing logos for various corporations on the computer display when the user performs a search. This form of advertising, however, is not very sophisticated and does not encourage user interaction.

Another form of advertising on the Internet is the creation of WEB "pages" or sites by advertisers. One variant of the use of a WEB page displays advertisements in a portion of the viewing area. A second variant, often used by corporate or other advertisers, is the use of WEB sites which employ attractive graphics in the hope of having the user interact with various advertising schemes. In addition, product ordering is usually available from these WEB sites. In most cases, users access these WEB sites by one of the following methods: knowing the Internet address; keyword searching; linking from a different WEB site; through an electronic shopping mall type site; through other advertisements on the Internet; or through the use of programs known as search browsers.

Current advertisers have attempted to improve the attractiveness of these WEB pages by including the use of sound, animated or rotating logos or pictures, and scrolling information. One system, Hot Java, available from Sun Microsystems, supports the execution of small applications programs written in a specific programming language executing within the browser on the local computer. This allows the WEB pages to provide richer content, such as animation or scrolling sports scores across a user's computer display, and better interaction with users. These effects, however, are only available while the user is viewing the specific WEB page incorporating the Hot Java technology.

Despite the fast and furious growth in this advertising sector, WEB sites are still regarded as "passive" advertising used predominantly for a corporate image rather than for selling products. Specifically the following drawbacks describe the current state of advertising on the Internet: transmitting the advertising information consumes a large amount of the bandwidth of the communications link between the user's computer and the network; access is initiated by the user rather than the advertiser; the user rather than the advertiser pays for access; accessing a site is a time consuming "hit or miss" process; and the process may improve the corporate image but creates little product demand.

U.S. Pat. No. 5,105,184 to Pirani et al. ("Pirani") discloses a system integrating commercial advertisements with computer software. The system discloses integrating commercial advertisements with different types of screens. Pirani, however, does not provide for any user input at the local computer as to the types of advertisements which are to be displayed. Thus, users would be forced to view numerous advertisements of which they are likely to have no interest. This will attenuate the users attention to the advertisements and decrease their effectiveness.

As noted above, a significant problem with current methods for advertising on computer networks is the consumption of significant portions of the bandwidth of the communications link between the user's computer and the computer network. Prior systems have attempted to utilize essentially unused time in telephone networks to deliver advertising or other information. U.S. Pat. No. 5,321,740 to Gregorek, et al. ("Gregorek") discloses a marketing system over an existing telephone network which modifies a portion of the call processing system to play an informational announcement in place of the usual ringback or busy signals. Gregorek differs from the present invention in a number of ways, including the fact that it does not provide any means for interacting with computers over a computer network. Also, Gregorek delivers the informational announcement only during a short splice of time when the user is waiting for callback information.

Current file transfer protocols, such as the File Transfer Protocol ("FTP") and the Trivial File Transfer Protocol ("TFTP"), for transferring files from a remote network, such as the Internet, via a communications link to a local computer are designed to transfer files as quickly as possible. Each computer process executing such a protocol attempts to make maximum use of the available communication resources. This leads to interference and an inevitable slowing down of other computer processes attempting to communicate over the communications link. There exists a need, therefore, for a file transfer process which is designed to behave as a background task and have a minimal impact on foreground communications.

There also exists a need to utilize the computer to display locally stored advertisements. Several software products provide "yellow pages" on CD-ROMs or other media such as floppy disks. The user may use these yellow pages to search for products or advertisers by name or description. This system of advertising is limited, however, in that it requires the user to actively search for advertisers or products and therefore does not spontaneously display products to the user.

Microsoft Windows interface provides a rudimentary form of spontaneous advertising by incorporating a Microsoft Windows logo as an option in its screen saver utility. This system, however, offers only a single advertisement in response to a user's response and therefore does not offer a variety of periodically changed advertisement content based on a user's interests.

SUMMARY OF THE INVENTION

The object of the present invention is to provide a process for transmitting an information file between a local computer and a remote computer network over a communications link with minimal interference to other processes executing on the computer which are also transmitting over the communications link.

It is a further object of the present invention to provide a method and system of presenting individualized advertise-

ments and other informational messages on a computer by allowing a user to select from a variety of advertisement or informational categories.

It is a further object of the present invention to provide a method and system of downloading and presenting individualized advertisements and other informational messages from a remote network to a local computer based on a user's selection of advertisement or informational categories.

It is a further object of the present invention to provide such a method and system of downloading and presenting individualized advertisements and other informational messages from a network to a local computer with minimal interference with other data being transmitted between the network and the local computer.

In one variant of the present invention, all advertisements or other informational messages originate on a network server which is accessed via the Internet or alternate on-line method. Select advertisements are transparently downloaded from the network server and stored locally on the user's local computer using a novel type of software referred to herein as a "Polite Agent." In a second variation, the entire advertisement database is locally stored on the local computer or a removable media such as CD-ROM. Manipulation and display of the advertising message is performed by software residing on the user's PC in accordance with preconfigured user preference information.

The advertisement is preferably displayed during idle time as a screen saver utility when the computer is not receiving keyboard input or updating the user's display. Other techniques for displaying the advertisement, such as periodic audio-only messages, screen background wallpaper, cursor modifications, and display in a window on the user's computer display are also available.

Users may enter their preferences by directly choosing categories of advertising or other informational content which most interest them or through interactive games and quizzes. Users may directly respond to advertising messages by participating in contests, requesting further product information, or ordering the advertised product. The advertisements are made attractive to the user by employing a variety of video, animation, sound or any other multimedia effects. Content may be based on an interactive theme such as a contest or special discount offers for on-line customers.

The system monitors the user's interaction with the advertisements and produces raw data on how many times a particular advertisement was accessed as well as the user's response to advertisements. All pertinent information is stored and sent back to a network server where it is made available to the advertisers. User requests for additional information may be directed to the advertiser itself or to the advertiser's WEB site on the network.

The system further comprises the use of a background software process, the Polite Agent, for transferring information between the network and the local computer. The Polite Agent monitors the communications link between the network and the local computer and transfers small portions of the information when the communications link utilization rate is low. In this manner the Polite Agent avoids significant interference with other communications applications transmitting over the communications link. The Polite Agent may also be utilized to transmit other types of information content, such as news, weather, stock quotes, sports scores, software updates or trip reservation information.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, reference is made to the following Detailed

Description taken in conjunction with the accompanying drawings in which:

FIG. 1 is a functional block diagram of a system architecture in accordance with the present invention;

FIG. 2 is a functional block diagram of a local computer and its related components in accordance with the present invention;

FIG. 3 is a functional block diagram of the software architecture components of the advertising system network server in accordance with the present invention;

FIG. 4 is a functional block diagram of the software architecture components of the local computer in accordance with the present invention;

FIG. 5 is a schematic representation of an advertisement file in accordance with the present invention;

FIG. 6 is a flowchart illustrating a method for a polite agent for communicating information with a remote network in accordance with the present invention;

FIG. 7 is a schematic representation of a job for transmitting data in accordance with the present invention;

FIG. 8 is a flowchart illustrating a method for an advertisement display manager constructed in accordance with the present invention;

FIG. 9 is a flowchart illustrating a method for an advertisement feeder for downloading advertisements from a network constructed in accordance with the present invention; and

FIG. 10 is a functional block diagram of an advertising system for displaying a local database of advertisements constructed in accordance with an alternative embodiment of the present invention.

DETAILED DESCRIPTION

Preferred embodiments of the present invention will now be described with continued reference to the drawings.

System Architecture

1. Network Architecture

FIG. 1 shows an overall view of a preferred embodiment of the system architecture. The Local Computer **500** is physically connected to the Network Service Provider **701** via a Communications Link **703**. The Network Service Provider **701** provides access to the Network **700**. Advertising System Server **600** is one of the nodes on the Network **700**.

a. Local Computer

As shown in FIG. 2, the Local Computer **500** preferably includes a Central Processor **510**, a Main Memory **511**, an Input/Output Controller **512**, a Display Device **513**, input devices such as a Keyboard **514** and a Pointing Device **515** (e.g., mouse, track ball, pen, slide pointer or similar device), and a Mass Storage Device **516**. These components communicate through a system bus or similar architecture.

Additionally, the Local Computer **500** is preferably connected to an internal or external Modem **520** or like device for communication with the Network **700**. Alternatively, the Local Computer **500** may be connected via an ISDN adapter and an ISDN line for communications with the Network **700**. The Modem **520** optionally allows for the establishment of voice calls through software control.

The Local Computer **500** preferably also includes a Sound Device **530**. The Local Computer **500** may also include a Local Storage Device **540** such as a floppy disk, CD-ROM or like device for local storage of the Local Advertisement Database **550**.

The Local Computer **500** is preferably under the control of a multi-process operating system including a TCP/IP

interface, and most preferably operated under the Microsoft® Windows-95 platform available from Microsoft Corporation of Redmond, Wash. However, the present invention may be embodied on a variety of different platforms, including Macintosh, UNIX, NextStep, MS-DOS, and the like.

b. Network

The Network **700** is preferably the World-Wide Internet. The World-Wide Internet ("Internet") is a world-wide network connecting thousands of computer networks. The dominant protocol used for transmitting information between computers on the Internet is the TCP/IP Network Protocol. Computers connect to the Internet either a fixed connection, in which case they become a "permanent" node on the Internet, or a dial-up connection, in which case they act as a node on the network as long as the connection is active. Internet addresses are the numbering system used in TCP/IP communications to specify a particular network or computer on the network with which to communicate.

The invention may also be practiced with commercial on-line services such as America Online, available from America Online Inc., CompuServe, available from H&R Block Inc., Prodigy, available from Prodigy Services, Microsoft Network, available from Microsoft Corp., as well as other like services from a variety of companies such as AT&T Corporation and MCI Communications Corp.

c. Network Service Provider

The Network Service Provider **701** provides access to the Network **700**. Commercial providers include: BBN, Netcom, and Prarienet.

d. Advertising System Server

The system preferably includes at least one Advertising System Server **600**. The main roles of the Advertising System Server **600** are to store Advertisements **50**, transfer the Advertisements **50** to the Local Computer **500**, and collect user feedback. The Local Computer **500** will initiate communication with the Advertising System Server **600**. Each user is assigned a unique user-ID which can not be changed by the user. This user-ID is used by the Advertising System Server **600** to track each user's activity, including which Advertisements **50** have been downloaded to the user.

When the Local Computer **500** connects to the Advertising System Server **600**, the Local Computer **500** will upload the user's user-ID and the configuration and user preference information to the Advertising System Server **600**. The Advertising System Server **600** uses this information to select the next Advertisement **50** to be downloaded. The Local Computer **500** also may directly request a specific advertisement through the use of a unique Advertisement-ID assigned to each Advertisement **50**. If feedback information has been collected it also will be uploaded when the Local Computer **500** connects to the Advertising System Service.

In an alternate embodiment of the present invention, the selected advertisement may be stored on any one of the plurality of advertising system servers connected to the Network **700**. In this embodiment, the Local Computer **500** initiates communication with a predetermined advertising system server. The predetermined advertising system server will select the next Advertisement **50** to be downloaded and transmit the network address of the advertising system server storing the Advertisement **50**. The Local Computer **500** uses the transmitted network address to request the selected Advertisement **50** from the appropriate advertising system server.

The Advertisements **50** stored on the Advertising System Server **600** may be translatable to one or more natural

languages. The Advertising System Server **600** will use each user's native language-ID to transmit the appropriate natural language version of the Advertisement **50**.

2. Advertising System Server Software Architecture

a. Software Modules

FIG. **3** is a functional block diagram of a software architecture of the Advertising System Server **600** embodying the invention.

The Local Computer **500** initially connects to the Connection Manager **710** which is responsible for allocating an available Job Manager **720** and returning its address to the agent. The Job Manager **720** then handles all further conversation with the Local Computer **500**. As noted above, the architecture optionally allows for a plurality of advertising system servers each running a plurality of Job Managers **720**. Thus, the Job Manager address returned to the Local Computer **500** may contain both the Internet address of the server and the process identification. The Job Manager **720** identifies and authenticates the user-ID against the Server Database **730**.

The Job Manager **720** creates a Network Job **725** for each user it communicates with. Each Network Job **725** communicates with the Local Computer **500** to select and download Advertisements **50**; collect feedback from the Local Computer **500**; check the user's participation and any awards from contests, etc.; and upgrades and installs the Local Computer **500** software versions. The Network Job **725** is responsible for selecting the next downloaded Advertisement **50** based on user configuration and preference data as described herein.

Under most current network models, including the current implementation of the Internet, users are typically charged based on the amount of time they are connected to specific resources on the network. Thus, the current system of downloading advertisements and other information in the background does not increase the cost to the user, as transmission occurs in background mode while the user is already connected to the network Service Provider **701**. Future implementations of these networks, however, may charge users based on the amount of information, or number of network "packets" or other units of data, the user has received. On such networks, the system should be able to track the amount of information transmitted, such as by counting the number advertisements, advertisement resources, or network packets (also known as "datagrams"), frames, segments or other units of network data containing advertisements. The Network Service Provider **701** may use this information to charge the system generated transmissions to the advertisers rather than the users. The Advertising System Server **600**, and most preferably the Network Job **725**, will be responsible for tracking the amount of information transmitted by the system. Alternatively, the TCP/IP Polite Agent **280** or other software process on the Local Computer **500** will track this information.

The Server Database **730** contains the system information, including: the Advertisements **50** or other informational content; listings of users; listings of advertisers; listings of network service providers; billing information; audit logs and statistics. The Server Database **730** also maintains information on active connections and their activity. The Server Database **730** is accessed through the Database Abstraction **740** interface which provides a layer of interface for all modules on the Advertising System Server **700**.

In addition to providing Advertisements **50** and other informational content to local computers, the Advertising System Server **600** also provides various management

services, such as billing information, viewing and gathering statistics on feedback information, and advertisement display audit-logs which may be sorted according to various filters such as advertiser or advertising category. The Advertising System Server **700** includes various other software administration tools for maintaining the system, including: a Database Feeder **754** for modifying the Server Database **730**; Monitoring Tools **752** for viewing the activity of the system; Configuration Tools **751** for modifying the behavior of the system; and Reporting Tools **753** for creating reports concerning the system.

b. Advertisement Records

FIG. **5** shows a schematic representation of an Advertisement **50**. Each Advertisement **50** in the Server Database **730** comprises an Advertisement Information Record **51** and a Resource List **52**. The Advertisement Information Record **51** contains information identifying the advertisement (including the Advertisement-ID **55**), its category, its size, and the hardware required to display the advertisement, such as sound boards, screen resolution and multimedia requirements. The Resource List **52** contains a list of resources (bitmaps, animations, digitized audio segments, executable code, etc.) that must exist on the Local Computer **500** or associated local LAN in order to present the advertisement. The Resource List **52** includes a unique resource-ID, a resource type, and a resource pointer. The resource pointer identifies a file, a database record, a block of data, or other means of identifying the resource. In this manner, resources can be shared by various Advertisements **50**.

3. Local Computer Software Modules

a. Software Architecture

FIG. **4** is a functional block diagram of the software modules and processes of the software architecture for a preferred embodiment of the invention on the Local Computer **500**, including a User Interface Setup Process **201**, Advertisements Display Manager **210**, Screen Saver Subsystem **220**, User Preference and Advertisements Database **230**, Feedback Manager **240**, Advertisements Feeder **250**, Internet Feedback **260**, Advertisement Killer **275**, Scheduler **265**, Internet Feeder **270**, TCP/IP Polite Agent **280**, TCP/IP Protocol Stack **290**, and PPP-TCP/IP Over Modem Protocol **295**.

The User Interface Setup Process **201** allows the user to configure the behavior of the system on their desktops. The Advertising Display Manager **210** is responsible for selecting and displaying Advertisements **50** from the User Preference and Advertisements Database **230**. The Advertisements Feeder **250** adds new Advertisements **50** to the User Preference and Advertisements Database **230**, while the Advertisement Killer **275** purges old Advertisements **50**. The Scheduler **265** schedules the display of time dependent Advertisements **50**, such as background wallpaper, sound only advertisements, and cursor advertisements. The Feedback Manager **240** sends user preferences, user statistics and feedback information to the Advertising System Server **600**. The TCP/IP Protocol Stack **290** and PPP-TCP/IP Over Modem Protocol Subsystem **295** handle the lower level details of transferring information to the Network **700**. The TCP/IP Polite Agent **280** is responsible for monitoring the communications line utilization rate and transmitting data during times of low communications line utilization.

b. Platform Implemented Functions

The Screen Saver Subsystem **220** tracks user interaction with the system. When the Screen Saver Subsystem **220** detects that the system has been idle, for example, when there has been no user interaction with the computer (through the use of the keyboard, mouse, pointing device or

other user input device), for a preconfigured time, it activates the Advertisements Display Manager **210** which will select an advertisement and display it.

In prior systems, screen savers are graphically oriented displays which appear after the computer has been idle for a predetermined period of time. This change of display is primarily designed to prevent screen 'burn in' on the monitor. Screen savers of the prior art are limited to one or more predefined content themes. By utilizing on-line communications, the screen saver of the present invention provides a wide variety of potential content themes which may be personalized and modified on a timely basis in accordance with user preferences.

The general mechanisms for implementing the Screen Saver Subsystem **220** are known in the art. In the preferred embodiment, the Windows-95 operating system monitors user input and calls a preconfigured module when the user has not entered input for a predetermined period. In the preferred embodiment, this module is the Advertisements Display Manager **210**. The Screen Saver Subsystem **220** is further described in Microsoft's *Windows-95 Software Development Kit*, including: WIN 32 Overview and reference manuals (chapter 79—Screen Saver Library), available from Microsoft Corp., which is hereby incorporated by reference. On other platforms which do not provide similar functionality, the Advertisements Display Manager **210** itself must monitor for idle time.

The general mechanisms and protocols for communicating with a network, such as the Internet, or on-line service, are known in the art. See, e.g., Stallings, W., *Data and Computer Communications*, Second Edition, Macmillan Publishing Co., (1988). The preferred embodiment utilizes the TCP/IP protocol (Transport Control Protocol/Internet Protocol) which is also well known in the art. See, e.g., Martin J., *TCP/IP Networking*, PTR Prentice Hall (1994). The disclosure of each of the foregoing is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

The TCP/IP Protocol Stack **290** is a set of programs that mediate between application programs and the physical communication line. The TCP/IP Protocol Stack **290** provides application programs with a standard set of function calls for communicating with other application programs connected to the Network **700**. Thus the application programs, here the TCP/IP Polite Agent **280**, do not have to account for the nature of the physical communication line or error correction.

The PPP-TCP/IP Over Modem Protocol **295** module provides the ability to use the TCP/IP protocol over a specific type of physical communication line, i.e. a pair of modems connected over a telephone line. In the preferred embodiment, the functions of the TCP/IP Protocol Stack **290** and PPP-TCP/IP Over Modem Protocol **295** are implemented within the Windows-95 platform and are accessed from the TCP/IP Polite Agent **280** via system calls. See Microsoft Windows Socket Specifications (rev. 1.1), the disclosure of which is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

c. User Preference and Advertisement Database

The User Preference and Advertisement Database **230** contains various information needed by the system. The primary data stored is the advertisement information (including executable code modules, bitmaps, video clips and sound clips). The database also stores display statistics, configuration information and user preference data.

Typically, the User Preference and Advertisement Database **230** is located on the Mass Storage Device **516**, however, in LAN installations, the User Preference and Advertisement Database **230** may be stored on a LAN server. This optimizes storage for the system, since an Advertisement **50** needs to be loaded on the LAN only once and is available for display by each workstation on the LAN.

The User Preference and Advertisement Database **230** is preferably accessed through a well-defined Application Programmer's Interface (API), as is known in the art. In the preferred embodiment, this may be an OLE2 compound file or other database means supplied by third-party software vendors. Thus, the location of the User Preference and Advertisement Database **230** will be transparent to the other processes.

d. User Interface Setup

The User Interface Setup Process **201** allows the user to configure the behavior of the system. This process allows the user to input and view preferences as to advertising categories, as well as local computer configuration data.

Local configuration data typically includes:

- a) The Advertising System Server **600** Internet name or Internet address.
- b) The amount of disk space on the Local Computer **200** or LAN which may be allocated to the advertising system.
- c) The length of time an advertisement should be stored before it is deleted from the Local Computer **200**.
- d) The overhead which may be caused by the advertisement transfer process, including the communications line threshold.
- e) Screen saver delay time.
- f) Whether feedback information may be sent to the Network **700**.

User preference information typically includes:

- a) Listings of advertisement categories which are to be given high priority, and those categories which are to be banned from being downloaded or displayed. Typical advertisement categories are "SCUBA diving equipment," "Fast food vendors," "Toys for ages 8-14," or "Cigarettes" and the like. The actual list of categories will be provided by the Advertising System Server **600**.
- b) Time periods during which sound-only advertisement are to be played.
- c) Whether wallpaper or cursor advertisements are allowed.
- d) Whether animation is allowed.
- e) Time periods and types of foreground activities during which advertisements and feedback information may be transmitted.
- f) Identification of the user's natural language.

Additionally the User Preference and Advertisement Database **230** stores information on the Local Computer's **200** platform capabilities, such as sound boards, screen resolution and multi-media capabilities.

The Advertisement Feeder **250** will incorporate the advertising preference information, the Local Computer's platform capabilities, disk space limitations, and other configuration data into its request for new advertisements. The Advertising System Server **600** uses this information in selecting the next advertisement to be transmitted. The user preference and configuration data may alternatively be stored on the Advertising System Server **600**.

The Job Manager **720** on the Network Server **600** matches the user preferences and configuration data against the category information for the available Advertisements **50**. Advertisements **50** matching the user's high priority categories and platform capabilities are selected for downloading.

Lower priority matches are also selected occasionally on a pseudo-random basis, such that Advertisements **50** within the user's high priority categories will have higher probabilities of being downloaded. The system also allows for increasing the probability of downloading particular advertisements **50**. This allows advertisers to pay increased rates to distribute advertisements faster and to a wider range of users.

User's preferably enter their preferences using whatever interfaces are most natural for the underlying platform. In the preferred embodiment, the user enters information through standard Windows-95 dialog boxes.

In one embodiment of the invention, the Advertising Display Manager **210** allows users to respond to Advertisements **50** being presented by selecting a user grading box which allows users to judge the Advertisements **50** on a scale from "do not show me this advertisement again" to "excellent." This information may be used to modify the user preferences, and may also be incorporated into user feedback information and sent to the Advertising System Server **600** for use by the advertisers.

The User Interface Setup Process **201** also allows users to browse through Advertisements **50** stored on their local system (hard disk or local LAN network), as well as those Advertisements **50** available from the Network **700**. For Advertisements **50** stored locally, whether downloaded from the Network **700** to the User Preference and Advertisement Database **230** or available on a Local Advertisement Database **550** stored on floppy disk, CD-ROM or like device, the User Interface Setup Process **201** displays a menu with the available Advertisements **50**. Each Advertisement **50** stored on the Network **700** or Local Advertisement Database **550** may optionally include a preview segment. The user may have the system present the preview of the Advertisement **50** such as a still image, a short animation sequence, or a sound clip. The user may then select the particular advertisement to be displayed. For Advertisements stored on the Network **700**, the Advertising System Server **600** will transmit a list of available Advertisements **50**. Users may also disable specific advertisements, or all advertisements of a specific advertiser from being transmitted to the Local Computer **500** or displayed.

e. Advertisement Display Manager

The Advertisement Display Manager **210** selects and displays Advertisements **50** from the User Preference and Advertisements Database **230**. The Advertisement Display Manager **210** is typically activated by the Screen Saver Subsystem **220** when the user has not entered input for a predetermined time, or from the Scheduler **265**. Alternatively, the user may directly access the Advertisement Display Manager **210** from the platform, such as through selecting an icon or other common method.

The Advertisement Display Manager **210** will display the collection of bitmaps, animation, and sound clips associated with the Advertisement **50**. FIG. 8 shows a flowchart of a preferred method of an Advertisement Display Manager **210** in accordance with the principles of the present invention. The Advertisement Display Manager **210** is typically called by either the Screen Saver Subsystem **220** to display a screen saver type advertisement after the system has been idle for a predefined period or by the Scheduler **265** to modify the background wallpaper or present a sound-only type advertisement on a periodic basis. The Advertisement Display Manager **210** selects and presents the next Advertisement **50** of the specified type to be presented from the User Preference and Advertisements Database **230**.

In the preferred embodiment, the display and other presentation capabilities for each advertisement are self-

contained within the Advertisement **50** itself. In this manner the Advertisement Display Manager **210** can support a virtually unlimited number of presentation techniques. The code needed for presenting the advertisement such as a digital sound or video decoder or animation file player, is a resource available from the Resource List **52** within the Advertisement structure **50**. The resource may exist in a number of forms such as executable or interpreted code or scripting code such as that used in Hot Java. When the resource consists of interpreted or scripted code, the interpreter itself becomes an additional resource which must be made available to the Local Computer. If necessary, the Advertisement Feeder **250** will download this resource to the Local Computer **500**, using the same techniques as used to download other advertisement data.

Many platforms, including the preferred Windows-95 platform, include a multi-media subsystem that provides APIs for playing animation, sound clips, video clips, etc. See Win32 Programmer's Reference Manual, hereby incorporated by reference. Alternatively, there are a wide variety of stand-alone tools suitable for providing such functions on Windows, Macintosh and other platforms.

In the preferred embodiment, each Advertisement **50** will include a small .DLL with an entry point with a pre-defined name. This entry point will be called by the Advertisement Display Manager **210** in order to display the Advertisement **50**. The advertisement entry point is specific for each Advertisement **50**. When the advertisement entry point is called, the particular code needed to present the given Advertisement **50** will be executed.

User interaction with the Advertisement Display Manager **210** is preferably initiated by pressing a predesignated key, for example **F10**. When the Advertisement Display Manager **210** is active, all user input is routed directly to the Advertisement Display Manager **210**, thus allowing for user interaction with Advertisements **50**. The Advertisement Display Manager **210** selectively forwards certain keys to the default operating system routine, which will typically terminate the Advertisement Display Manager **210**. The user may interact with the Advertisement Display Manager **210** through a number of ways, including answering questioners, initiating a WEB browser to connect directly to an advertiser WEB page on the Network **700**, or automatically initiating a voice connection through the Modem **520** to the advertiser.

Additional aspects of the present invention utilize a variety of techniques for presenting the Advertisements **50**. These techniques include displaying advertising as the background "wallpaper" of the display or modifying the cursor to include an advertiser's logo or other symbol. Additionally, small advertising logos or other advertising content may be placed on the Display Device **513** either at a fixed location on the Display Device **513** or fixed relative to user display windows such that when the user display window is moved on the display the advertisement will move with the window. In the preferred Windows-95 environment, these functions are performed through system calls as described in the Win32 Programmer's Reference Manual, available from Microsoft.

An additional presentation technique is the use of sound-only advertising. The Advertisement Display Manager **210** will make use of a Sound Device **530** on the Local Computer **200**. Any sound devices supported by the platform are suitable. In the preferred embodiment, this includes the Sound Blaster card, available from Creative Labs.

f. Scheduler

The Scheduler **265** keeps track of the list of timing-dependent operations. When the time comes to execute a

timing-dependent Advertisement **50**, as for example changing the wallpaper or playing a sound-only Advertisement **50**, the Scheduler **265** notifies the Advertising Display Manager **210**, which performs the required action.

g. Advertisements Feeder

The Advertisement Feeder **250**, is responsible for adding new Advertisements **50** to the User Preference and Advertisement Database **230**. Advertisements **50** preferably are provided from the Internet through the Internet Feeder **270**, however, the Advertisements Feeder **250** is not dependent on the type of advertisement source and may receive Advertisements **50** from other sources, such as commercial on-line services, via other feeder mechanisms and other types of polite agents, as shown by references **271** and **272**, respectively, in FIG. 4.

FIG. 9 shows a flow chart of a preferred embodiment of an Advertisement Feeder **250** constructed in accordance with the invention.

To download a new Advertisement **50**, the Advertisement Feeder **250** first creates a Polite Agent Job **285** to request the Advertising System Server **600** to select the next advertisement for downloading (step **251**). The Advertising System Server **600** selects the next Advertisement **50** to be transferred based on the individual user's preferences and configuration and pricing parameters attached to each Advertisement **50**. The Advertising System Server **600** sends the Resource List **52**, such as executable code modules, bitmaps, animation, sound clips, scripting systems, etc., that the Advertisement **50** needs in order to be presented. The Advertisement Feeder **250** queries the User Preference and Advertisement Database **230** and determines which resources are already available locally, i.e. on the user's PC or LAN. The Advertisement Feeder **250** creates a Polite Agent Job **285** for each resource not in the User Preference and Advertisement Database **230**, requesting the Advertising System Server **600** to download only the necessary resources (steps **252**, **254**). Once the resources have been downloaded, the Advertisement Feeder **250** adds the Advertisement **50** to the User Preference and Advertisement Database **230** (step **253**).

An important part of the functionality of the client system is the ability to resume the transfer of an Advertisement **50** which had been only partially transferred during the previous connection., i.e. the client system is preferably able to re-establish transmission of a file after a break in the Communications Link **703**. Preferably, the client system will resume transmission from the point in the file at which communications was broken off. In the preferred embodiment, this functionality is implemented within the TCP/IP Polite Agent **280** and each Polite Agent Job **285**.

h. Advertisement Killer

The Advertisement Killer **275** periodically scans the User Preference and Advertisements Database **230**, and purges Advertisements **50** that satisfy its purge criteria. Typical criterion include the total time the advertisement has been stored and the number of times displayed. Additionally, Advertisements **50** are purged on user demand through user interaction with the Advertisements **50** or the User Interface Setup Process **201**.

i. Feedback Manager

The Feedback Manager **220** is responsible for sending feedback information to the Advertising System Server **600**. This information includes statistics on displayed Advertisements **50**, including user ratings of specific advertisements and the time and length an advertisement was displayed. The Feedback Manager **220** also transmits information which was gathered from the user during interaction with the

Advertisements **50**, such as through games and questionnaires. This feedback information may be used as a basis for calculating the advertiser's charge.

j. Polite Agent Technology

The system incorporates a type of intelligent software agent technology referred to herein as a "Polite Agent." The role of the Polite Agent is to perform communication tasks in the background without imposing a noticeable overhead on the user. FIG. 6 illustrates the preferred embodiment of the TCP/IP Polite Agent **280** utilizing the TCP/IP protocol. The TCP/IP Polite Agent **280** transmits information during periods of low line utilization without causing a noticeable slowdown in the data transfer rate of other processes communicating over the Communications Link **703**. The TCP/IP Polite Agent **280** constantly monitors communications status and determines periods of low communication line utilization. It then uses the TCP/IP communications resources, available on the platform, to transfer a portion of the data. Preferably, the agent does not initiate the communication itself, but rather takes advantage of communications resources once the initial Communications Link **703** with the Network Service Provider **701** has been established, thus avoiding additional user charges.

If the communications resource utilization remains low and ample resources are available the software agent performs its designated data transfer task. Alternatively, if communications resource utilization becomes high due to other applications executing on the Local Computer **500** or the Communications Link **703** is disconnected (e.g., the line goes down), the TCP/IP Polite Agent **280** temporarily suspends its data transfer operation until ample resources are available once again. At that point, the TCP/IP Polite Agent **280** recovers the data transfer process from the point where the transfer was suspended, thereby avoiding the need to retransmit data.

Low line utilization occurs when the communications line is busy no more than a predetermined percentage of time. This threshold may be fixed (typically at 30%) user-configurable, or dynamic. When dynamically determined, the threshold may vary with a number of parameters such as the length of time the TCP/IP Polite Agent **280** has been waiting to transmit, the number or type of Polite Agent Jobs **285** on the Polite Agent Queue **286**, the amount of data which the TCP/IP Polite Agent wishes to transfer, and the type of data being transferred.

Neither the Advertisement Feeder **250** nor the Feedback Manager **240**, directly perform data transfer. Instead, they place Polite Agent Jobs **285** in the Polite Agent Queue **286** which will be called by the TCP/IP Polite Agent **280** when appropriate. The Polite Agent Jobs **285** perform the actual data transfer.

In the preferred embodiment of the invention, the target operating system will be Microsoft Windows-95 utilizing a TCP/IP protocol. Extension of these operations for different protocols or operating systems will be apparent to those of ordinary skill in the art.

In step A (**41**) of the TCP/IP Polite Agent process **280**, a check is made to see if the Communication Link **703** has been established. This can be done in various ways known to those skilled in the art. A preferred method is to "ping" (send a packet to and receive a response from) the Advertising System Server **600**. See, e.g., J. Martin, *TCP/IP Networking*, PTR Prentice Hall Inc. (1994) (pages 147-48), the disclosure of which is hereby incorporated by reference. An alternative method is to "ping" the Network Service Provider **701**.

In step B (**42**), the line utilization threshold is calculated. As noted above, this calculation may vary in different

embodiments of the present invention. Thus, the line utilization may be fixed, user-configurable or dynamic. The threshold calculation also preferably takes into account the load caused by communication generated by the Polite Agent Jobs **285** themselves. This prevents the TCP/IP Polite Agent from not transmitting when the Communications Link **703** is busy primarily due to its own communications.

In step C (**43**), the current communication line utilization is obtained. For TCP/IP under Windows-95, statistical information regarding the communication line utilization is available from the operating system, including such information as bytes/second. In the preferred embodiment, this sampling does not impose a significant overhead on the system and therefore does not cause any noticeable degradation of foreground processes.

In step D (**44**), the current communications line utilization is compared to the calculated threshold. If the current utilization is higher than the calculated threshold, the TCP/IP Polite Agent **280** will not perform communication and will return to step A. At this point the TCP/IP Polite Agent **280** may be temporarily suspended by the operating system.

In step E (**45**), the next Polite Agent Job **285** to be executed is selected. Several Polite Agent Jobs **285** can be pending on the Polite Agent Queue **286**. The TCP/IP Polite Agent **280** will alternate between the Polite Agent Jobs **285** preferably on a round-robin schedule allowing all of the Polite Agent Jobs **285** to execute in turn.

In step F (**46**), the Polite Agent Job **285** is executed. The Polite Agent Jobs **285** are designed in such way that they generate a small amount of communication, for example sending 1K of information, each time they are executed.

FIG. 7 shows a flow chart for a Polite Agent Job **285** embodying the present invention for transmitting data to the Network **700**. The Polite Agent Job **285** first checks (step A) to see whether this is the first time the job has been called by the TCP/IP Polite Agent **280**. If it is the first time the job has been called, the current file position is initialized to the beginning of the file (step B). The file to be transferred is then opened (step C). If the file open is unsuccessful, the Polite Agent Job **285** returns an ERROR flag to the calling TCP/IP Polite Agent **280** which then terminates the Polite Agent Job **285**, removes it from the Polite Agent Queue **286** and marks the job as terminated with error (steps D, E). If the file opens without error, the Polite Agent Job **285** seeks to the current file position in the transferring file and reads the next block of data (steps F, G, H). If no data is left to be read (end of file condition), the Polite Agent Job **285** closes the file and returns a DONE flag to the TCP/IP Polite Agent **280** which terminates the job and marks the Polite Agent Job **285** as completed successfully (steps I, J). If data was successfully read from the transferring file, the Polite Agent Job **285** transmits the name of the file, file position and file block contents to the Network **700** via the TCP/IP Protocol Stack **290** (steps I, K). The Polite Agent Job **285** then updates the current file position and stores it on persistent storage, such as the Local Computer's Mass Storage Device **516** (step L). Finally, the Polite Agent Job **285** closes the transfer file and returns a flag asking the TCP/IP Polite Agent **280** to again schedule and execute this job again.

It will be obvious to one of ordinary skill in the art how to modify the above type of Polite Agent Job **285** for other types of tasks, such as receiving files. The TCP/IP Polite Agent **280** is capable of transferring any type of information file including, executable code, digitized audio or video, and text, for a variety of types of content such as advertisements, news, or weather, etc. The information file transmitted by the Polite Agent Job **285** may consist of a true file type sup-

ported by the platform, or, alternatively, any block of data such as a database record.

In one variation of the present invention, the Polite Agent Job **285** receives as an input the current communication line utilization and a line utilization threshold value. The Polite Agent Job **285** uses this threshold to calibrate its operation by calculating how many network packets, bytes, or other units of data may be transferred without increasing the load beyond the line utilization threshold value.

One of the most common types of Polite Agent Jobs **285**, is a file-transfer task. Each time the file-transfer task is run it will send (or receive) a small portion of the file, typically between 0.5–1K.

Part of the information collected by the present invention may be sensitive in nature, for example, a user's responses to advertising contests may need to be authenticated to prevent fraudulent responses. The system will use known methods such as public key encryption and digital signatures in order to authenticate the information sent by the Local Computer **500**. These methods are well known to those in the art.

The technology of the Polite Agent is general and may be applied in other systems and to the transmission of other types of information content such as news and weather, sports scores and stock quotes, software support information and executable updates, and airlines reservations information.

4. Locally Stored Advertisement Database

FIG. **10** shows a variation of the present invention in which a plurality of Advertisements **50** are stored locally to the Local Computer **500** in a Local Advertisement Database **550** on the Local Storage Device **540** or LAN. In this embodiment, the Advertisements **50** are not down-loaded from the Advertising System Server **600**, but rather are selected from the Local Advertisement Database **550**. The Advertisements Feeder **250** selects Advertisements **50** for presentation on the Local Computer **500** from the Local Advertisement Database **550**. In this embodiment, the Advertisements Feeder **250** includes the selection functionality described for the Network Job **725** above. Selected Advertisements **50** are loaded into the User Preference and Advertisement Database **230** for display as described above.

It is understood that various other modifications will be apparent to and can be readily made by those skilled in the art without departing from the scope and spirit of the present invention. Accordingly, it is not intended that the scope of the claims be limited to the description or illustrations set forth herein, but rather that the claims be construed as encompassing all features of patentable novelty that reside in the present invention, including all features that would be treated as equivalents by those skilled in the art.

What is claimed is:

1. A method of presenting individualized advertisement items on a computer, said individualized advertisement items selected from a database of advertisement items stored on a network, said method comprising the steps of:

- (a) inputting user priorities on the computer from a predefined set of general categories of advertising information;
- (b) selecting a plurality of advertisement items for presentation from the database of advertisement items, said database of advertisement items containing at least one category of advertising information associated with each advertisement item, said selection based on said user priorities and said associated advertising category;
- (c) downloading said plurality of selected advertisement items from said database of advertisement items stored

on said network, said downloading of said set of selected advertisement items performed using the process comprising the steps of:

- (i) monitoring the communication line utilization rate for a communications link coupling the computer and network;
 - (ii) determining whether to transmit data in the current iteration based on said monitored line utilization rates;
 - (iii) if said determination of step (ii) indicates data may be transmitted, transmitting a portion of the remaining advertisement item between the network and the computer;
 - (iv) tracking the remaining untransmitted portion of the advertisement item, said tracking providing tracking information for any remaining untransmitted portion of the advertisement item;
 - (v) storing said tracking information indicating the last transmitted portion of said advertisement item, said tracking information being stored in persistent memory;
 - (vi) repeating steps (i)–(v) until the advertisement item has been transferred, whereby said downloading of said advertisement item continues from the last transmitted portion of said advertisement item after any intervening breaks in said communications link or breaks in the availability of said computer, and
 - (d) presenting at least one of said plurality of selected advertisement items on said computer, said at least one presented advertisement item being periodically varied from said plurality of selected advertisement items.
- 2.** The method of presenting individualized advertisement items of claim **1** wherein said presentation technique of step (d) is selected from the group consisting of screen-saver display, background wallpaper display, cursor display, fixed screen location display, relative screen location displayed and audio messages played at various times.
- 3.** The method of presenting individualized advertisement items of claim **1** further comprising the step of monitoring the communications link until the communications link coupling the computer and network has been established.
- 4.** A method of presenting information items on a computer, said information items selected from a remote database of information items on a remote network, said method comprising the steps of:
- (a) inputting user priorities on the computer from a predefined set of general information categories;
 - (b) selecting a plurality of information items for presentation from the remote database of information items, said database containing at least one information category associated with each information item, said selection based on said input user priorities and said associated information category;
 - (c) downloading the plurality of selected information items from the remote network to the computer, said downloading of the plurality of selected information items using the process comprising:
 - (i) monitoring a current communication line utilization rate for a communications link coupling the computer and the remote network;
 - (ii) determining whether to transmit data in the current iteration based on said monitored communications line utilization rate;
 - (iii) if said determination of step (ii) indicates data may be transmitted, transmitting a portion of the remaining information item between the remote network and the computer;

- (iv) tracking the remaining untransmitted portion of the information item, said tracking providing tracking information for any remaining untransmitted portion of the information item;
- (v) storing said tracking information indicating the last transmitted portion of said information item, said tracking information being stored in persistent memory;
- (vi) repeating steps (i)–(v) until the information item has been transferred, whereby said downloading of said information item continues from the last transmitted portion of said information item after any intervening breaks in said communications link or breaks in the availability of said computer; and

- (d) presenting at least one of said plurality of selected information items on said computer, said at least one presented information item being periodically varied from said plurality of selected information items.

5. The method of presenting individualized information items of claim 4 wherein said presentation technique of step (d) is selected from the group consisting of screen-saver display, background wallpaper display, cursor display, fixed screen location display, relative screen location display, and audio messages played at various times.

6. The method of presenting individualized information items of claim 4 further comprising the steps of:

- (e) collecting feedback information regarding the presented information items; and
- (f) uploading said feedback information to said remote network.

7. A process for transmitting a file of data between a client computer and a server computer coupled by a communications link on a computer network, said process comprising the steps of:

- (a) monitoring the communication line utilization rate for said communications link;
- (b) comparing said communication line utilization to preestablished values;
- (c) calibrating the amount of data to be transmitted based on said comparison of said communication line utilization to said preestablished values;
- (d) transmitting said calibrated amount of data;
- (e) tracking the remaining untransmitted portion of said file, said tracking providing tracking information for any remaining untransmitted portion of the file;
- (f) storing said tracking information indicating the last transmitted portion of said file, said tracking information being stored in persistent memory; and
- (g) repeating steps (a)–(f) until the file has been transferred,

whereby said process for transferring a file continues transmitting from the last transmitted portion of said file after any intervening breaks in said communications link or breaks in the availability of said client computer.

8. The process of claim 7 further comprising the step of: monitoring the communications link until the communications link coupling the first computer and second computer has been established.

9. The process of claim 7 wherein said file of data comprises executable code.

10. The process of claim 7 wherein said calibration of step (c) comprises the step of calculating the amount of data to be transferred without increasing the communications line utilization rate above a preestablished threshold value.

11. The process of claim 7 wherein said step (a) of monitoring the communications line utilization comprises the step of sampling the line utilization.

12. The process of transmitting a file of data of claim 7 wherein a plurality of categories of programs are transmitting or receiving data on said communications link including at least one category of program implementing said process of transmitting a file of data, said monitoring of step (a) providing line utilization rate information for said plurality of categories of programs, said calibration of step (c) taking into account said monitored line utilization for said at least one category of program implementing said process of transmitting a file of data.

13. A process for transmitting a file of data between a client computer and a server computer coupled by a communications link on a computer network, said process comprising the steps of:

- (a) monitoring the communication line utilization rate for said communications link;
- (b) utilizing said monitored line utilization rate to determine whether to transmit data in the current iteration, and to calculate the amount of data to be transmitted in the current iteration;
- (c) if said determination of step (b) indicates data should be transmitted in the current iteration, transmitting the amount of data calculated in step (b);
- (d) tracking the remaining untransmitted portion of the file, said tracking providing tracking information for any remaining untransmitted portion of the file;
- (e) storing said tracking information indicating the last transmitted portion of said file, said tracking information being stored in persistent memory; and
- (f) repeating steps (a)–(e) for a new iteration until the file has been transferred,

whereby said process for transferring said file utilizes said line utilization rate as a feedback mechanism for controlling the transfer of data via said communications link, said process further continuing transmission from the last transmitted portion of said file after any intervening breaks in said communications link or breaks in the availability of said client computer.

14. A process for transmitting a file between a server computer and a local computer, said local computer coupled to server computer by a communications link on a computer network, said process comprising the steps of:

- (a) monitoring the communications link to determine if the communications link coupling the local computer and server computer has been established;
- (b) determining a communication line utilization rate for the communications link;
- (c) if said communications link has been established, transmitting a portion of data from the remaining file between the network and the local computer, the amount of data in said portion being a function of said communication line utilization rate and one or more preestablished values;
- (d) tracking the remaining untransmitted portion of the file, said tracking providing tracking information for any remaining untransmitted portion of the file;
- (e) storing said tracking information indicating the last transmitted portion of said file, said tracking information being stored in persistent memory; and
- (f) repeating steps (a)–(e) until the file has been transferred,

whereby said process for transferring said file continues transmitting from the last transmitted portion of said file after any intervening breaks in said communications link or breaks in the availability of said local computer.

EXHIBIT B



US006317789B1

(12) **United States Patent**
Rakavy et al.

(10) **Patent No.:** **US 6,317,789 B1**
(45) **Date of Patent:** ***Nov. 13, 2001**

(54) **METHOD AND APPARATUS FOR TRANSMITTING AND DISPLAYING INFORMATION BETWEEN A REMOTE NETWORK AND A LOCAL COMPUTER**

(75) Inventors: **Yuval Rakavy; Eli Barkat**, both of Jerusalem (IL)

(73) Assignee: **Backweb, Ltd.**, Jerusalem (IL)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/274,612**

(22) Filed: **Mar. 23, 1999**

Related U.S. Application Data

(63) Continuation of application No. 08/517,666, filed on Aug. 22, 1995, now Pat. No. 5,913,040.

(51) **Int. Cl.⁷** **G06F 15/173**
(52) **U.S. Cl.** **709/224; 709/203**
(58) **Field of Search** **709/224, 225, 709/226, 229, 231, 233, 100, 103, 104, 105, 107, 223, 203; 345/736, 748**

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,719,567	1/1988	Whittington et al.	710/117
4,799,146	1/1989	Chauvel	710/260
5,099,420	3/1992	Barlow et al.	710/119
5,105,184	4/1992	Pirani et al.	345/115
5,165,012	11/1992	Crandall et al.	.
5,220,564	6/1993	Tuch et al.	379/257
5,226,041 *	7/1993	Waclawsky et al.	.
5,283,639	2/1994	Esch et al.	725/32
5,285,442	2/1994	Iwamura et al.	370/234
5,305,195	4/1994	Murphy	705/1
5,313,455	5/1994	Van der Wal et al.	370/232

5,319,455	6/1994	Hoarty et al.	725/34
5,321,740	6/1994	Gregorek et al.	345/347
5,347,632	9/1994	Filepp et al.	709/202
5,355,501	10/1994	Gross et al.	713/323
5,361,091	11/1994	Hoarty et al.	725/119
5,390,172	2/1995	Kuang	370/432
5,404,505 *	4/1995	Levinson	.
5,412,416	5/1995	Nemirofsky	725/36
5,428,789 *	6/1995	Waldron, III	.
5,455,826	10/1995	Ozveren et al.	370/232
5,488,609	1/1996	Hluchyj et al.	370/232
5,504,744	4/1996	Adams et al.	370/232

(List continued on next page.)

OTHER PUBLICATIONS

J. Ridgon, Coming Soon to the Internet: Tools to Add Glitz to the Web's Offerings, Wall Street Journal, Aug. 15, 1995.

J. Martin, TCP/IP Networking, PTR Prentic Hall, 1994 (pp. 147-148).

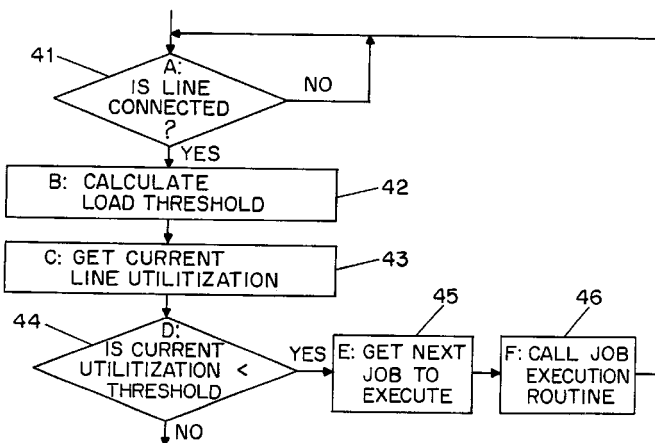
Primary Examiner—Mehmet B. Geckil

(74) *Attorney, Agent, or Firm*—Skadden, Arps, Slate, Meagher & Flom LLP

(57) **ABSTRACT**

Methods and apparatus are provided for selecting advertisements and other information from a computer network database based on user defined preferences and transmitting the selected advertisement in background mode over a communications link between the computer network and a local computer with minimal interference with other processes communicating over the communications link. This method includes monitoring the communications link and transmitting portions of the advertisement when the communications link line utilization is below a preestablished threshold. Methods and apparatus are also provided for displaying or otherwise presenting the selected advertisements on the user's computer. Additional methods and apparatus are provided for selecting and presenting information stored on a local storage media based on user defined preferences.

22 Claims, 8 Drawing Sheets



Page 2

U.S. PATENT DOCUMENTS			5,675,742	10/1997	Jain et al.	709/226
5,555,377	9/1996	Christensen et al.	5,684,960 *	11/1997	Geyer et al. .	
5,572,643 *	11/1996	Judson .	5,913,040 *	6/1999	Rakavy et al.	370/229
5,588,003 *	12/1996	Ohba et al. .	6,049,798 *	4/2000	Bishop et al.	707/10
5,600,364	2/1997	Hendricks et al.				725/9
5,604,542	2/1997	Dedrick 348/552				
			* cited by examiner			

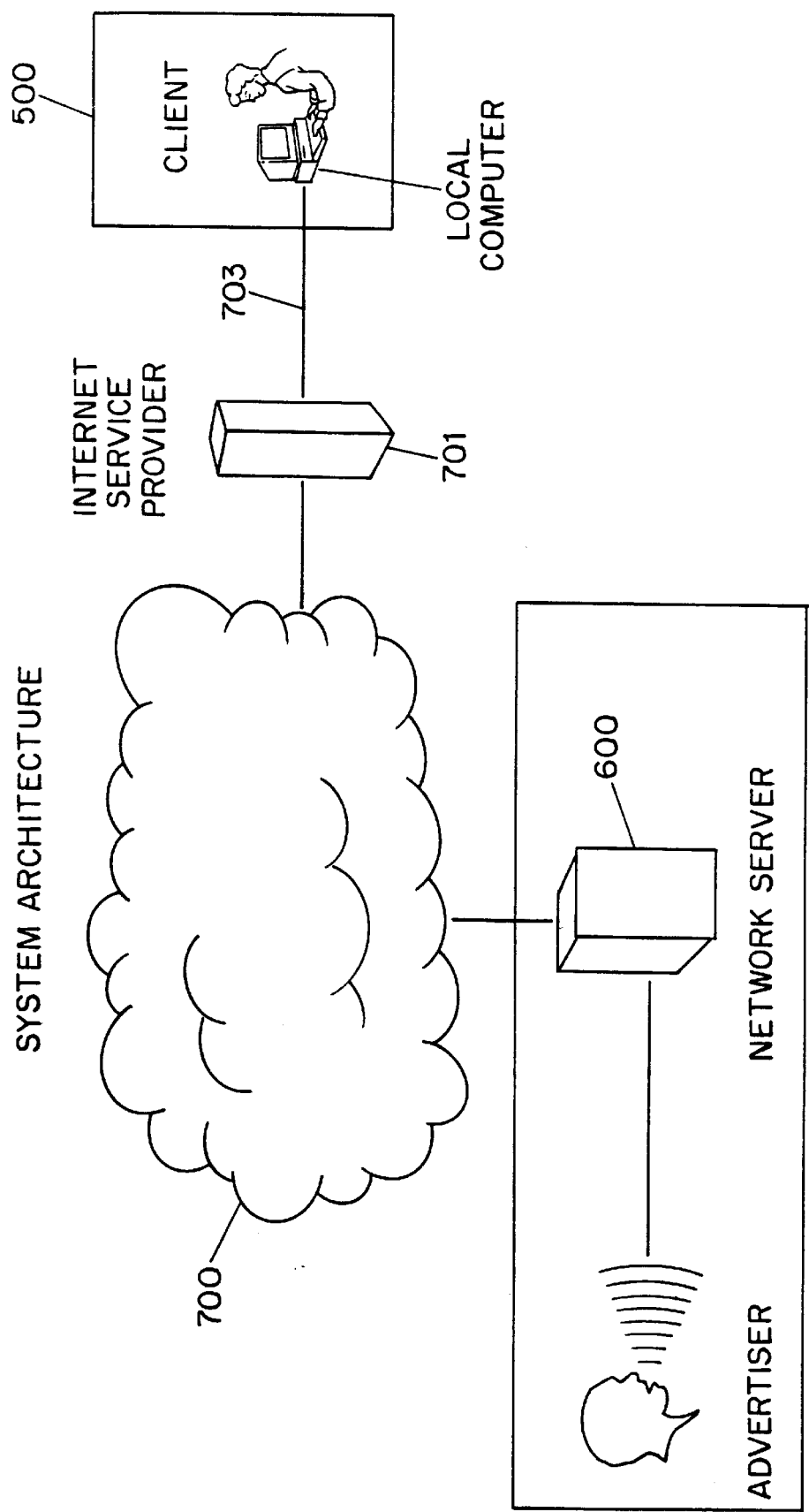


FIG. 1

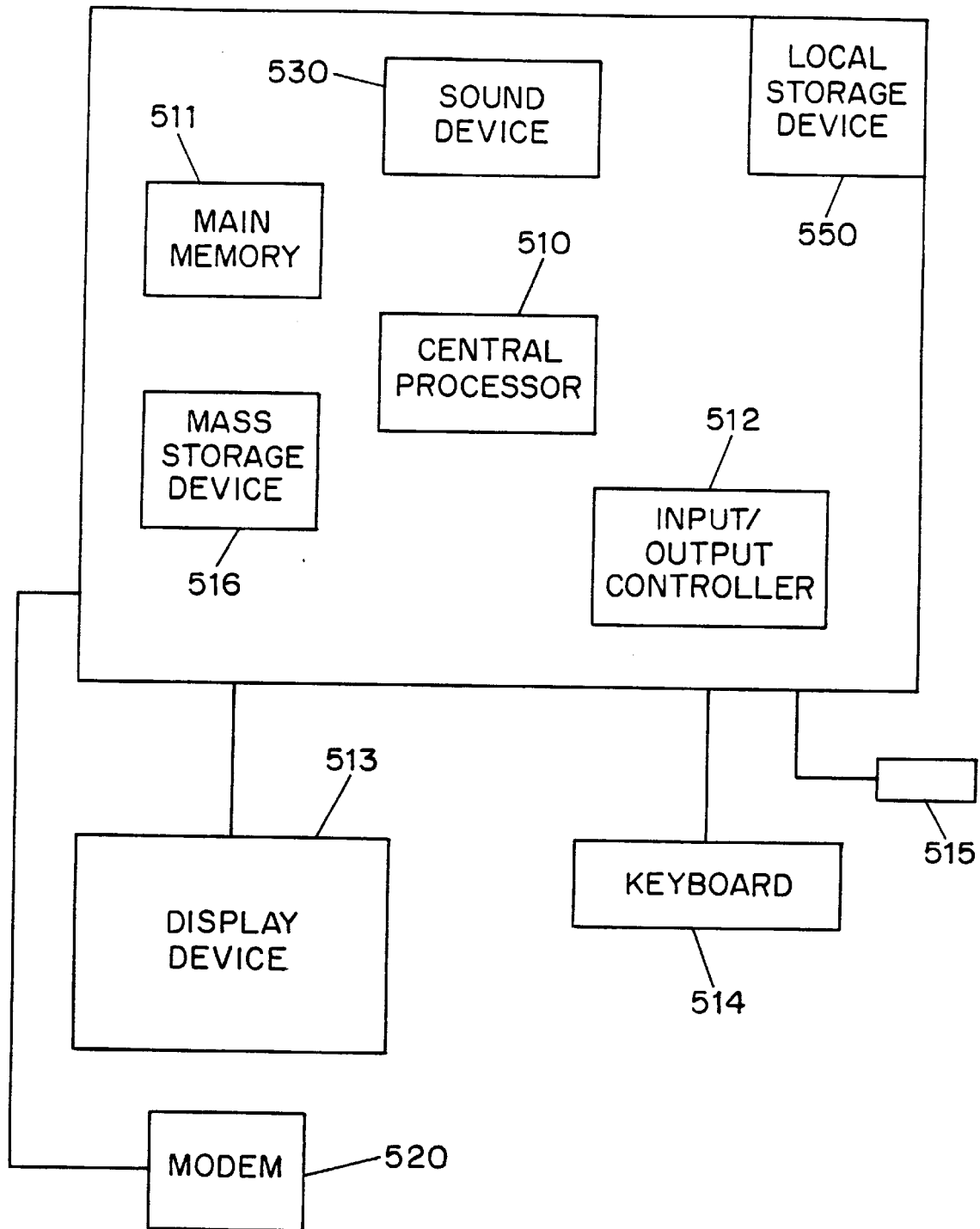


FIG. 2

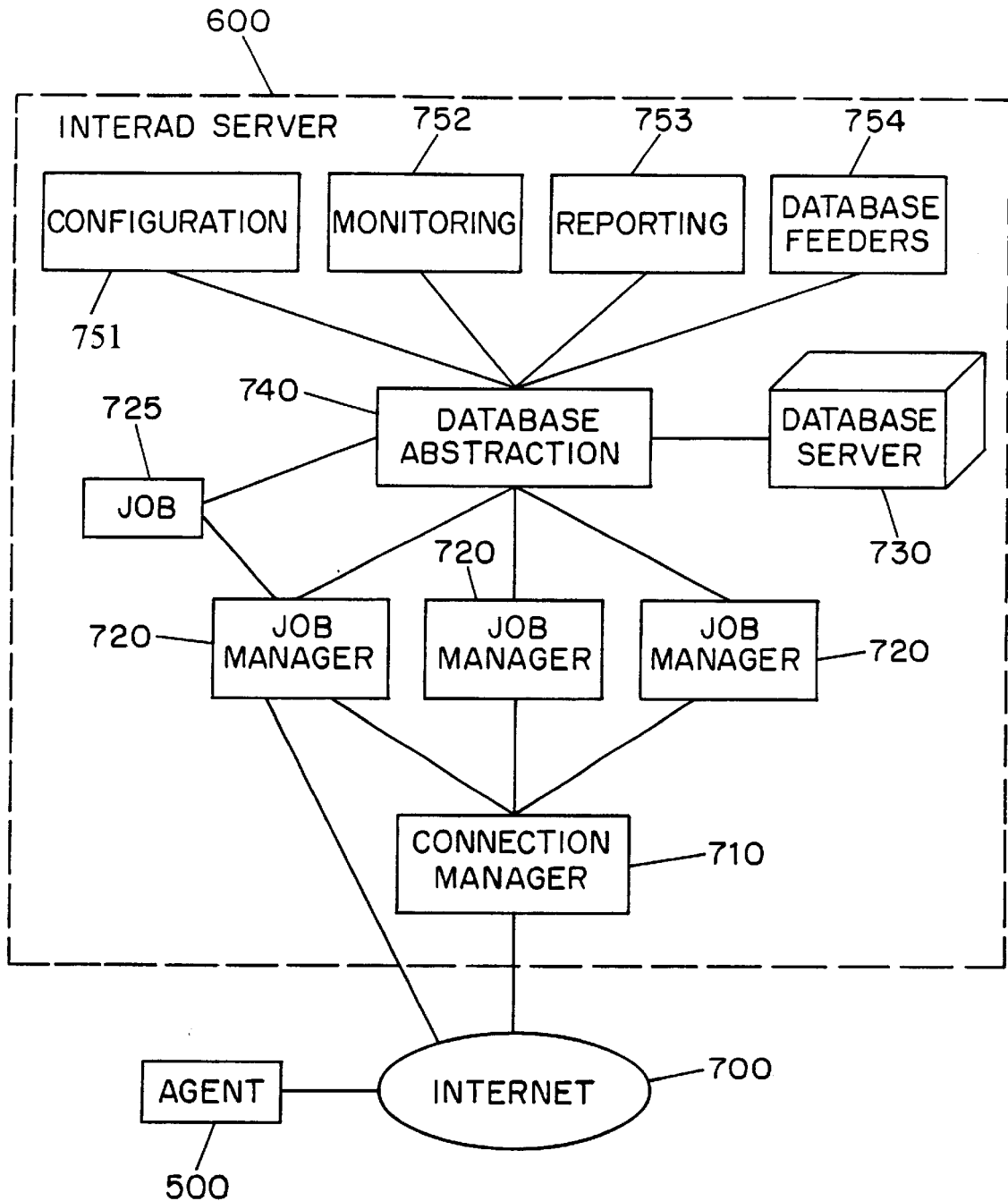
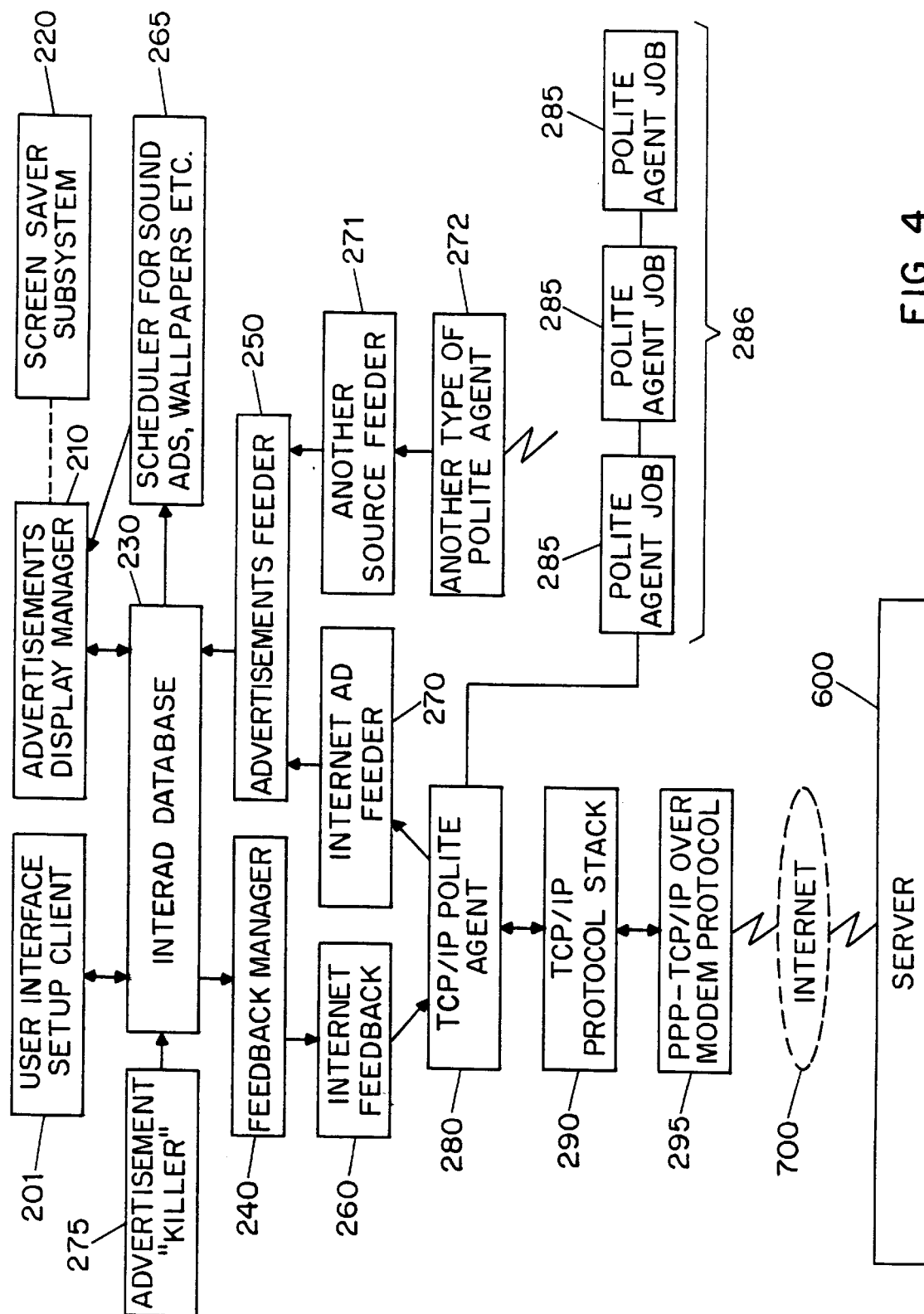


FIG. 3



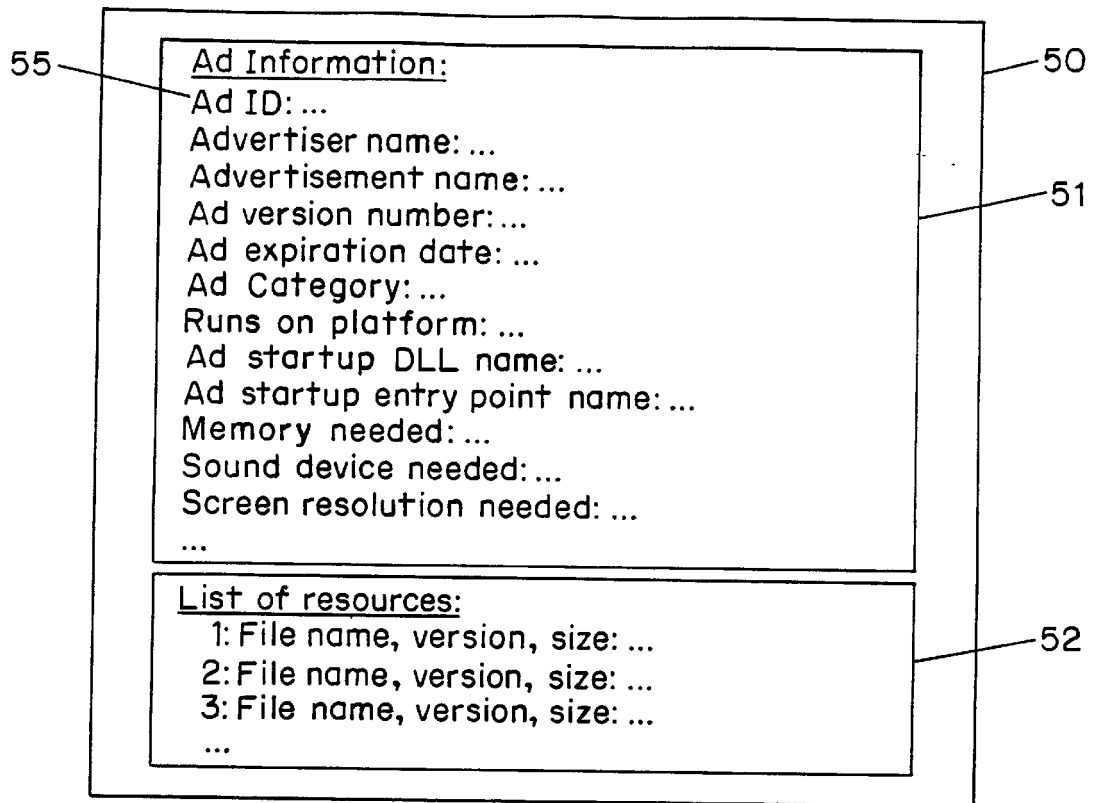


FIG. 5

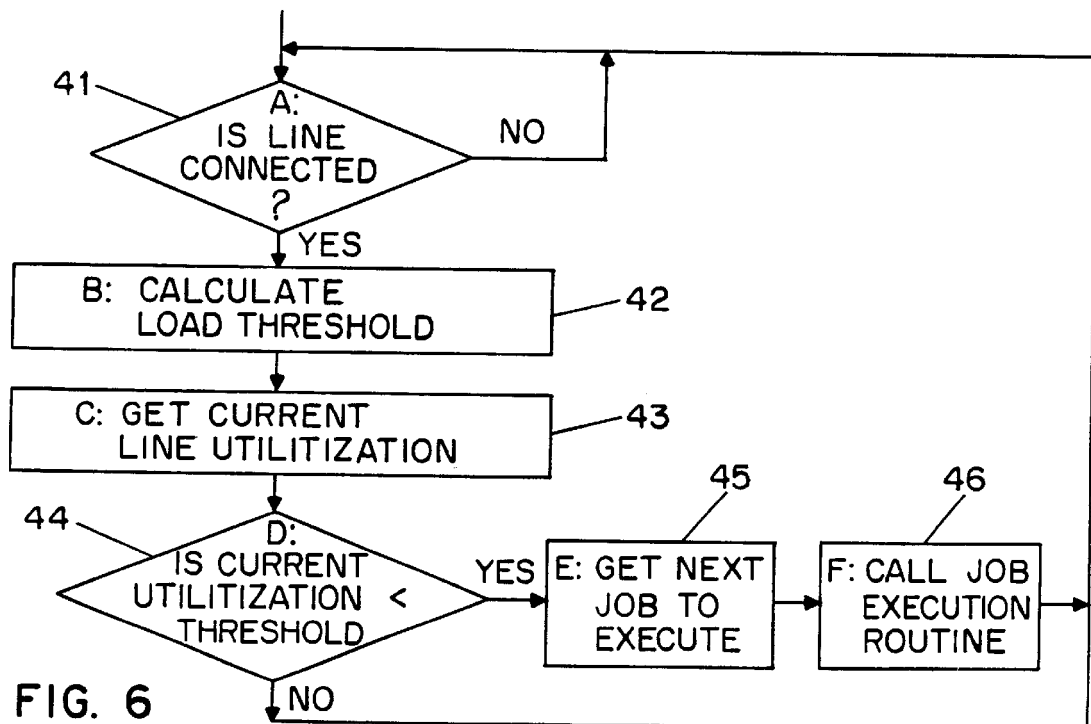


FIG. 6

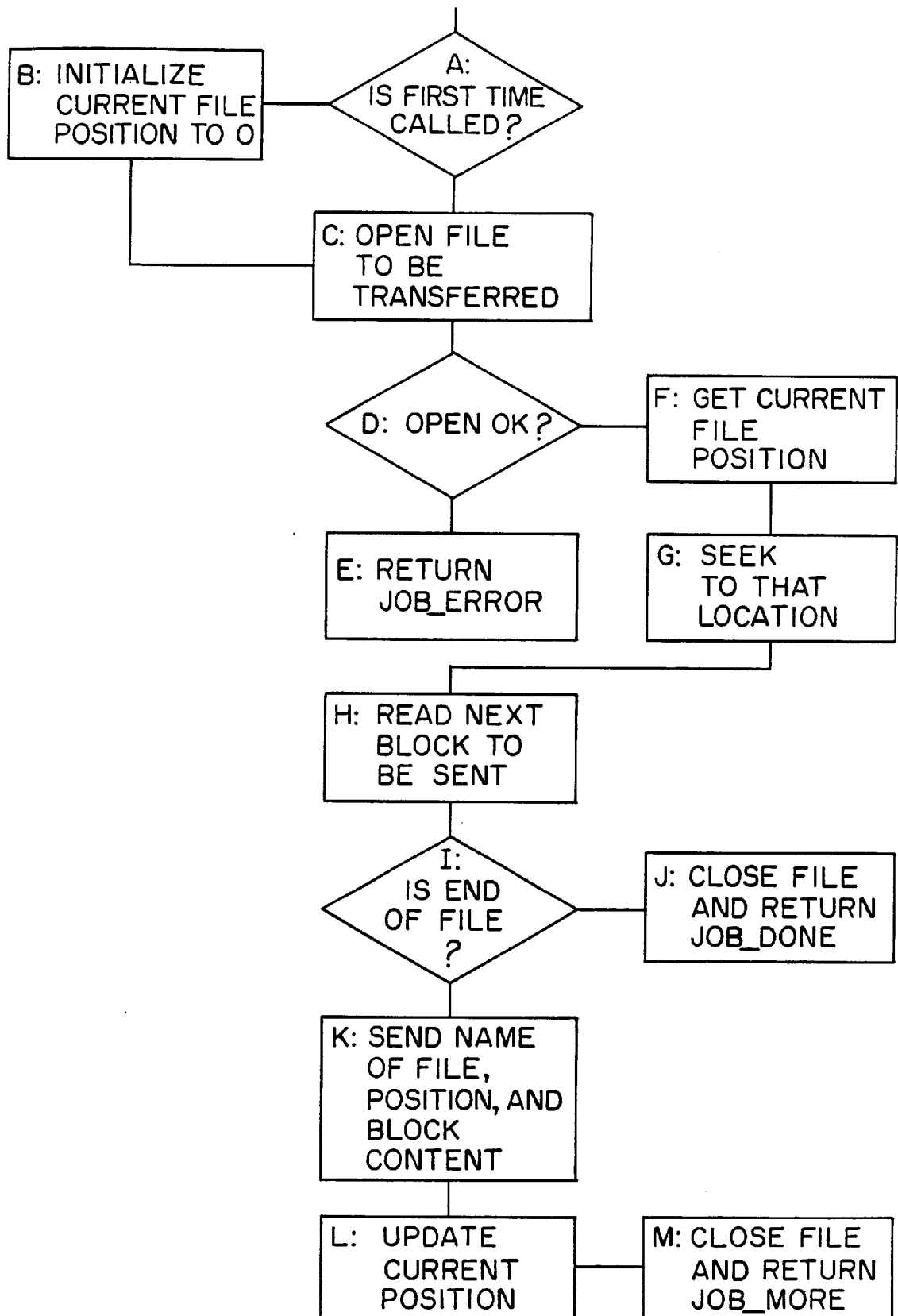


FIG. 7

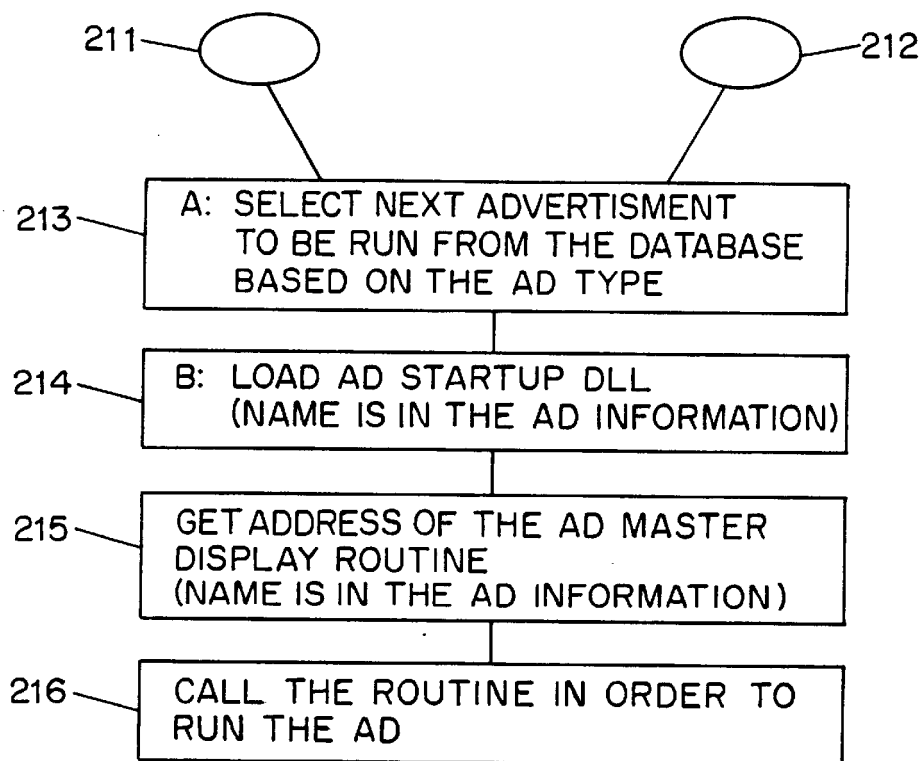


FIG. 8

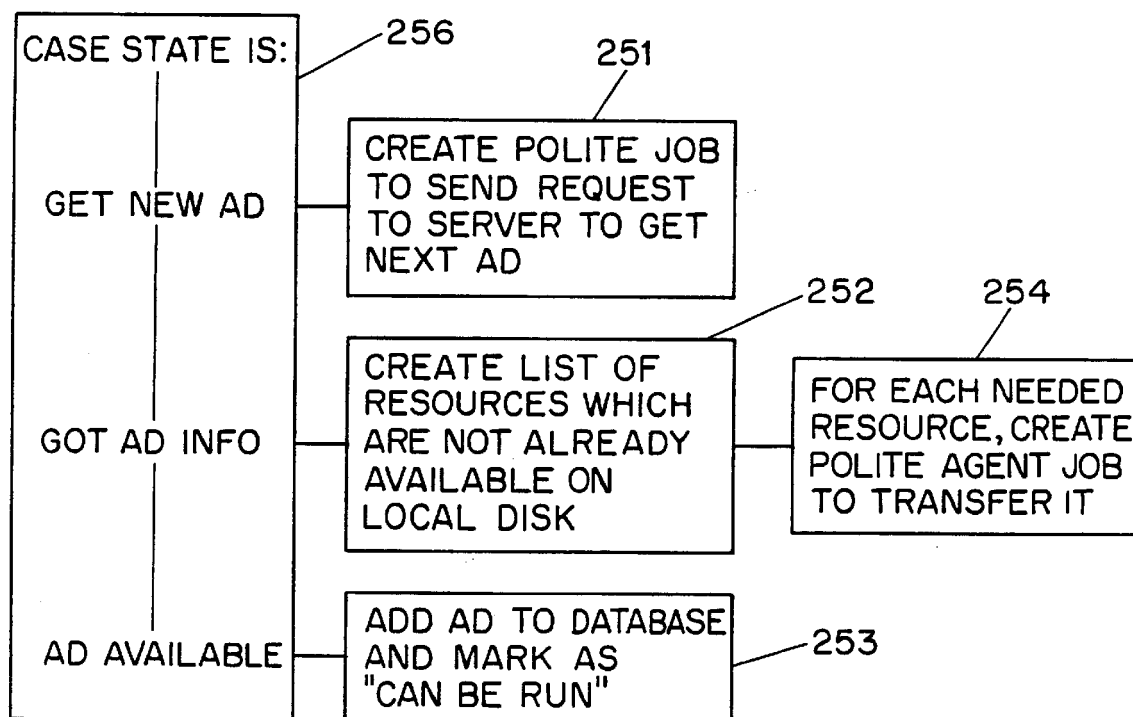


FIG. 9

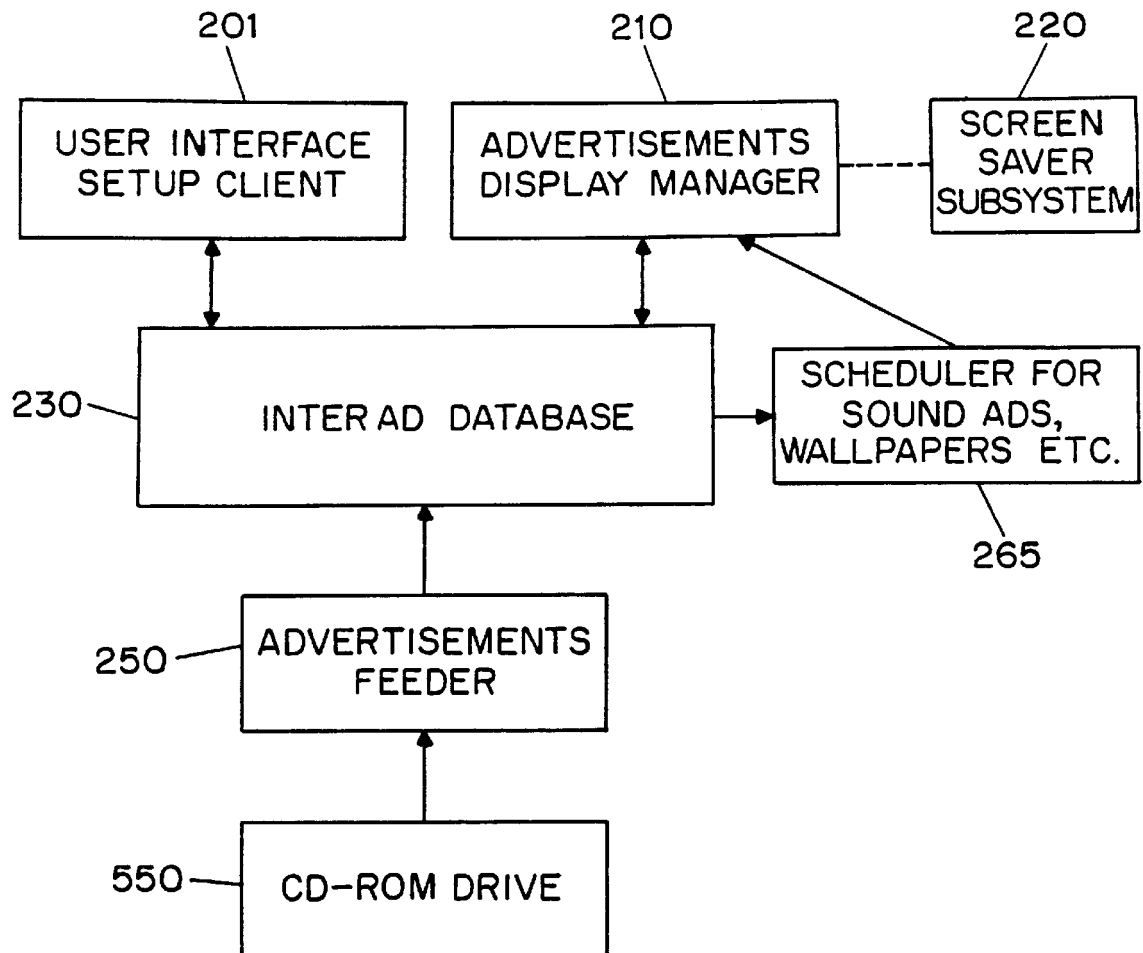


FIG. 10

METHOD AND APPARATUS FOR TRANSMITTING AND DISPLAYING INFORMATION BETWEEN A REMOTE NETWORK AND A LOCAL COMPUTER

CROSS-REFERENCE TO RELATED APPLICATIONS

This is a continuation of U.S. application Ser. No. 08/517,666, filed Aug. 22, 1995, which is now issued U.S. Pat. No. 5,913,040.

FIELD OF THE INVENTION

This invention relates generally to advertisement computer display systems and more particularly to a method and system for displaying advertisements and other information on a computer based on general user selected criteria and transmitting such information from a remote network to the local computer.

BACKGROUND OF THE PRESENT INVENTION

There are two major forms of advertising which are currently being employed on the Internet and commercial on-line services. One form is the use of a small advertisement on WEB pages which are commonly accessed. For example, a portion of the screen display for WEB pages used to access Internet searches may include a corporate logo or other advertisement material. Typical of this style of advertising is the Netscape™ Internet Browser software available from Netscape Communications Corporation of Mountain View Calif., which presents a box containing logos for various corporations on the computer display when the user performs a search. This form of advertising, however, is not very sophisticated and does not encourage user interaction.

Another form of advertising on the Internet is the creation of WEB "pages" or sites by advertisers. One variant of the use of a WEB page displays advertisements in a portion of the viewing area. A second variant, often used by corporate or other advertisers, is the use of WEB sites which employ attractive graphics in the hope of having the user interact with various advertising schemes. In addition, product ordering is usually available from these WEB sites. In most cases, users access these WEB sites by one of the following methods: knowing the Internet address; keyword searching; linking from a different WEB site; through an electronic shopping mall type site; through other advertisements on the Internet; or through the use of programs known as search browsers.

Current advertisers have attempted to improve the attractiveness of these WEB pages by including the use of sound, animated or rotating logos or pictures, and scrolling information. One system, Hot Java, available from Starwave Corp of Bellevue, Wash., supports the execution of small applications programs written in a specific programming language executing within the browser on the local computer. This allows the WEB pages to provide richer content, such as animation or scrolling sports scores across a user's computer display, and better interaction with users. These effects, however, are only available while the user is viewing the specific WEB page incorporating the Hot Java technology.

Despite the fast and furious growth in this advertising sector, WEB sites are still regarded as "passive" advertising used predominantly for a corporate image rather than for selling products. Specifically the following drawbacks

describe the current state of advertising on the Internet: transmitting the advertising information consumes a large amount of the bandwidth of the communications link between the user's computer and the network; access is initiated by the user rather than the advertiser; the user rather than the advertiser pays for access; accessing a site is a time consuming "hit or miss" process; and the process may improve the corporate image but creates little product demand.

U.S. Pat. No. 5,105,184 to Pirani et al. ("Pirani") discloses a system integrating commercial advertisements with computer software. The system discloses integrating commercial advertisements with different types of screens. Pirani, however, does not provide for any user input at the local computer as to the types of advertisements which are to be displayed. Thus, users would be forced to view numerous advertisements of which they are likely to have no interest. This will attenuate the users attention to the advertisements and decrease their effectiveness.

As noted above, a significant problem with current methods for advertising on computer networks is the consumption of significant portions of the bandwidth of the communications link between the user's computer and the computer network. Prior systems have attempted to utilize essentially unused time in telephone networks to deliver advertising or other information. U.S. Pat. No. 5,321,740 to Gregorek, et al. ("Gregorek") discloses a marketing system over an existing telephone network which modifies a portion of the call processing system to play an informational announcement in place of the usual ringback or busy signals. Gregorek differs from the present invention in a number of ways, including the fact that it does not provide any means for interacting with computers over a computer network. Also, Gregorek delivers the informational announcement only during a short splice of time when the user is waiting for callback information.

Current file transfer protocols, such as the File Transfer Protocol ("FTP") and the Trivial File Transfer Protocol ("TFTP"), for transferring files from a remote network, such as the Internet, via a communications link to a local computer are designed to transfer files as quickly as possible. Each computer process executing such a protocol attempts to make maximum use of the available communication resources. This leads to interference and an inevitable slowing down of other computer processes attempting to communicate over the communications link. There exists a need, therefore, for a file transfer process which is designed to behave as a background task and have a minimal impact on foreground communications.

There also exists a need to utilize the computer to display locally stored advertisements. Several software products provide "yellow pages" on CD-ROMs or other media such as floppy disks. The user may use these yellow pages to search for products or advertisers by name or description. This system of advertising is limited, however, in that it requires the user to actively search for advertisers or products and therefore do not spontaneously display products to the user.

Microsoft Windows interface provides a rudimentary form of spontaneous advertising by incorporating a Microsoft Windows logo as an option in its screen saver utility. This system, however, offers only a single advertisement in response to a user's response and therefore does not offer a variety of periodically changed advertisement content based on a user's interests.

SUMMARY OF THE INVENTION

The object of the present invention is to provide a process for transmitting an information file between a local com-

puter and a remote computer network over a communications link with minimal interference to other processes executing on the computer which are also transmitting over the communications link.

It is a further object of the present invention to provide a method and system of presenting individualized advertisements and other informational messages on a computer by allowing a user to select from a variety of advertisement or informational categories.

It is a further object of the present invention to provide a method and system of downloading and presenting individualized advertisements and other informational messages from a network to a local computer from a remote network to a local computer based on a user's selection of advertisement or informational categories.

It is a further object of the present invention to provide such a method and system of downloading and presenting individualized advertisements and other informational messages from a network to a local computer with minimal interference with other data being transmitted between the network and the local computer.

In one variant of the present invention, all advertisements or other informational messages originate on a network server which is accessed via the Internet or alternate on-line method. Select advertisements are transparently downloaded from the network server and stored locally on the user's local computer using a novel type of software referred to herein as a "Polite Agent." In a second variation, the entire advertisement database is locally stored on the local computer or a removable media such as CD-ROM. Manipulation and display of the advertising message is performed by software residing on the user's PC in accordance with preconfigured user preference information.

The advertisement is preferably displayed during idle time as a screen saver utility when the computer is not receiving keyboard input or updating the user's display. Other techniques for displaying the advertisement, such as periodic audio-only messages, screen background wallpaper, cursor modifications, and display in a window on the user's computer display are also available.

Users may enter their preferences by directly choosing categories of advertising or other informational content which most interest them or through interactive games and quizzes. Users may directly respond to advertising messages by participating in contests, requesting further product information, or ordering the advertised product. The advertisements are made attractive to the user by employing a variety of video, animation, sound or any other multimedia effects. Content may be based on an interactive theme such as a contest or special discount offers for on-line customers.

The system monitors the user's interaction with the advertisements and produces raw data on how many times a particular advertisement was accessed as well as the user's response to advertisements. All pertinent information is stored and sent back to a network server where it is made available to the advertisers. User requests for additional information may be directed to the advertiser itself or to the advertiser's WEB site on the network.

The system further comprises the use of a background software process, the Polite Agent, for transferring information between the network and the local computer. The Polite Agent monitors the communications link between the network and the local computer and transfers small portions of the information when the communications link utilization rate is low. In this manner the Polite Agent avoids significant interference with other communications applications trans-

mitting over the communications link. The Polite Agent may also be utilized to transmit other types of information content, such as news, weather, stock quotes, sports scores, software updates or trip reservation information.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, reference is made to the following Detailed Description taken in conjunction with the accompanying drawings in which:

FIG. 1 is a functional block diagram of a system architecture in accordance with the present invention;

FIG. 2 is a functional block diagram of a local computer and its related components in accordance with the present invention;

FIG. 3 is a functional block diagram of the software architecture components of the advertising system network server in accordance with the present invention;

FIG. 4 is a functional block diagram of the software architecture components of the local computer in accordance with the present invention

FIG. 5 is a schematic representation of an advertisement file in accordance with the present invention;

FIG. 6 is a flowchart illustrating a method for a polite agent for communicating information with a remote network in accordance with the present invention;

FIG. 7 is a schematic representation of a job for transmitting data in accordance with the present invention;

FIG. 8 is a flowchart illustrating a method for an advertisement display manager constructed in accordance with the present invention;

FIG. 9 is a flowchart illustrating a method for an advertisement feeder for downloading advertisements from a network constructed in accordance with the present invention; and

FIG. 10 is a functional block diagram of an advertising system for displaying a local database of advertisements constructed in accordance with an alternative embodiment of the present invention.

DETAILED DESCRIPTION

Preferred embodiments of the present invention will now be described with continued reference to the drawings.

System Architecture

1. Network Architecture

FIG. 1 shows an overall view of a preferred embodiment of the system architecture. The Local Computer 500 is physically connected to the Network Service Provider 701 via a Communications Link 703. The Network Service Provider 701 provides access to the Network 700. Advertising System Server 600 is one of the nodes on the Network 700.

a. Local Computer:

As shown in FIG. 2, the Local Computer 500 preferably includes a Central Processor 510, a Main Memory 511, an Input/Output Controller 512, a Display Device 513, input devices such as a Keyboard 514 and a Pointing Device 515 (e.g., mouse, track ball, pen, slide pointer or similar device), and a Mass Storage Device 516. These components communicate through a system bus or similar architecture.

Additionally, the Local Computer 500 is preferably connected to an internal or external Modem 520 or like device for communication with the Network 700. Alternatively, the Local Computer 500 may be connected via an ISDN adapter

and an ISDN line for communications with the Network 700. The Modem 520 optionally allows for the establishment of voice calls through software control.

The Local Computer 500 preferably also includes a Sound Device 530. The Local Computer 500 may also include a Local Storage Device 540 such as a floppy disk, CD-ROM or like device for local storage of the Local Advertisement Database 550.

The Local Computer 500 is preferably under the control of a multi-process operating system including a TCP/IP interface, and most preferably operated under the Microsoft® Windows-95 platform available from Microsoft Corporation of Redmond, Wash. However, the present invention may be embodied on a variety of different platforms, including Macintosh, UNIX, NextStep, MS-DOS, and the like.

b. Network

The Network 700 is preferably the World-Wide Internet. The World-Wide Internet ("Internet") is a world-wide network connecting thousands of computer networks. The dominant protocol used for transmitting information between computers on the Internet is the TCP/IP Network Protocol. Computers connect to the Internet use either a fixed connection, in which case they become a "permanent" node on the Internet, or a dial-up connection, in which case then act as a node on the network as long as the connection is active. Internet addresses are the numbering system used in TCP/IP communications to specify a particular network or computer on the network with which to communicate.

The invention may also be practiced with commercial on-line services such as America Online, available from America Online Inc., CompuServe, available from H&R Block Inc., Prodigy, available from Prodigy Services, Microsoft Network, available from Microsoft Corp., as well as other like services from a variety of companies such as AT&T Corporation and MCI Communications Corp.

c. Network Service Provider

The Network Service Provider 701 provides access to the Network 700. Commercial providers include: BBN, Netcom, and Prarienet.

d. Advertising System Server

The system preferably includes at least one Advertising System Server 600. The main roles of the Advertising System Server 600 are to store Advertisements 50, transfer the Advertisements 50 to the Local Computer 500, and collect user feedback. The Local Computer 500 will initiate communication with the Advertising System Server 600. Each user is assigned a unique user-ID which can not be changed by the user. This user-ID is used by the Advertising System Server 600 to track each user's activity, including which Advertisements 50 have been downloaded to the user.

When the Local Computer 500 connects to the Advertising System Server 600, the Local Computer 500 will upload the user's user-ID and the configuration and user preference information to the Advertising System Server 600. The Advertising System Server 600 uses this information to select the next Advertisement 50 to be downloaded. The Local Computer 500 also may directly request a specific advertisement through the use of a unique Advertisement-ID 55 assigned to each Advertisement 50. If feedback information has been collected it also will be uploaded when the Local Computer 500 connects to the Advertising System Service.

In an alternate embodiment of the present invention, the selected advertisement may be stored on any one of the plurality of advertising system servers connected to the Network 700. In this embodiment, the Local Computer 500

initiates communication with a predetermined advertising system server. The predetermined advertising system server will select the next Advertisement 50 to be downloaded and transmit the network address of the advertising system server storing the Advertisement 50. The Local Computer 500 uses the transmitted network address to request the selected Advertisement 50 from the appropriate advertising system server.

The Advertisements 50 stored on the Advertising System Server 600 may be translatable to one or more natural languages. The Advertising System Server 600 will use each user's native language-ID to transmit the appropriate natural language version of the Advertisement 50.

2. Advertising System Server Software Architecture
a. Software Modules

FIG. 3 is a functional block diagram of a software architecture of the Advertising System Server 600 embodying the invention.

The Local Computer 500 initially connects to the Connection Manager 710 which is responsible for allocating an available Job Manager 720 and returning its address to the agent. The Job Manager 720 then handles all further conversation with the Local Computer 500. As noted above, the architecture optionally allows for a plurality of advertising system servers each running a plurality of Job Managers 720. Thus, the Job Manager address returned to the Local Computer 500 may contain both the Internet address of the server and the process identification. The Job Manager 720 identifies and authenticates the user-ID against the Server Database 730.

The Job Manager 720 creates a Network Job 725 for each user it communicates with. Each Network Job 725 communicates with the Local Computer 500 to select and download Advertisements 50; collect feedback from the Local Computer 500; check the user's participation and any awards from contests, etc.; and upgrades and installs the Local Computer 500 software versions. The Network Job 725 is responsible for selecting the next downloaded Advertisement 50 based on user configuration and preference data as described herein.

Under most current network models, including the current implementation of the Internet, users are typically charged based on the amount of time they are connected to specific resources on the network. Thus, the current system of downloading advertisements and other information in the background does not increase the cost to the user, as transmission occurs in background mode while the user is already connected to the network Service Provider 701. Future implementations of these networks, however, may charge users based on the amount of information, or number of network "packets" or other units of data, the user has received. On such networks, the system should be able to track the amount of information transmitted, such as by counting the number advertisements, advertisement resources, or network packets (also known as "datagrams"), frames, segments or other units of network data containing advertisements. The Network Service Provider 701 may use this information to charge the system generated transmissions to the advertisers rather than the users. The Advertisement System Server 600, and most preferably the Network Job 725, will be responsible for tracking the amount of information transmitted by the system. Alternatively, the TCP/IP Polite Agent 280 or other software process on the Local Computer 500 will track this information.

The Server Database 730 contains the system information, including: the Advertisements 50 or other informational content; listings of users; listings of advertis-

ers; listings of network service providers; billing information; audit logs and statistics. The Server Database **730** also maintains information on active connections and their activity. The Server Database **730** is accessed through the Database Abstraction **740** interface which provides a layer of interface for all modules on the Advertising System Server **700**.

In addition to providing Advertisements **50** and other informational content to local computers, the Advertising System Server **600** also provides various management services, such as billing information, viewing and gathering statistics on feedback information, and advertisement display audit-logs which may be sorted according to various filters such as advertiser or advertising category. The Advertising System Server **700** includes various other software administration tools for maintaining the system, including: a Database Feeder **754** for modifying the Server Database **730**; Monitoring Tools **752** for viewing the activity of the system; Configuration Tools **751** for modifying the behavior of the system; and Reporting Tools **753** for creating reports concerning the system.

b. Advertisement Records

FIG. **5** shows a schematic representation of an Advertisement **50**. Each Advertisement **50** in the Server Database **730** comprises an Advertisement Information Record **51** and a Resource List **52**. The Advertisement Information Record **51** contains information identifying the advertisement (including the Advertisement-ID **55**), its category, its size, and the hardware required to display the advertisement, such as sound boards, screen resolution and multimedia requirements. The Resource List **52** contains a list of resources (bitmaps, animations, digitized audio segments, executable code, etc.) that must exist on the Local Computer **500** or associated local LAN in order to present the advertisement. The Resource List **52** includes a unique resource-ID, a resource type, and a resource pointer. The resource pointer identifies a file, a database record, a block of data, or other means of identifying the resource. In this manner, resources can be shared by various Advertisements **50**.

3. Local Computer Software Modules

a. Software Architecture

FIG. **4** is a functional block diagram of the software modules and processes of the software architecture for a preferred embodiment of the invention on the Local Computer **500**, including a User Interface Setup Process **201**, Advertisements Display Manager **210**, Screen Saver Subsystem **220**, User Preference and Advertisements Database **230**, Feedback Manager **240**, Advertisements Feeder **250**, Internet Feedback **260**, Advertisement Killer **275**, Scheduler **265**, Internet Feeder **270**, TCP/IP Polite Agent **280**, TCP/IP Protocol Stack **290**, and PPP-TCP/IP Over Modem Protocol **295**.

The User Interface Setup Process **201** allows the user to configure the behavior of the system on their desktops. The Advertising Display Manager **210** is responsible for selecting and displaying Advertisements **50** from the User Preference and Advertisements Database **230**. The Advertisements Feeder **250** adds new Advertisements **50** to the User Preference and Advertisements Database **230**, while the Advertisement Killer **275** purges old Advertisements **50**. The Scheduler **265** schedules the display of time dependent Advertisements **50**, such as background wallpaper, sound only advertisements, and cursor advertisements. The Feedback Manager **240** sends user preferences, user statistics and feedback information to the Advertising System Server **600**. The TCP/IP Protocol Stack **290** and PPP-TCP/IP Over Modem Protocol Subsystem **295** handle the lower level

details of transferring information to the Network **700**. The TCP/IP Polite Agent **280** is responsible for monitoring the communications line utilization rate and transmitting data during times of low communications line utilization.

b. Platform Implemented Functions

The Screen Saver Subsystem **220** tracks user interaction with the system. When the Screen Saver Subsystem **220** detects that the system has been idle, for example, when there has been no user interaction with the computer (through the use of the keyboard, mouse, pointing device or other user input device), for a preconfigured time, it activates the Advertisements Display Manager **210** which will select an advertisement and display it.

In prior systems, screen savers are graphically oriented displays which appear after the computer has been idle for a predetermined period of time. This change of display is primarily designed to prevent screen 'burn in' on the monitor. Screen savers of the prior art are limited to one or more predefined content themes. By utilizing on-line communications, the screen saver of the present invention provides a wide variety of potential content themes which may be personalized and modified on a timely basis in accordance with user preferences.

The general mechanisms for implementing the Screen Saver Subsystem **220** are known in the art. In the preferred embodiment, the Windows-95 operating system monitors user input and calls a preconfigured module when the user has not entered input for a predetermined period. In the preferred embodiment, this module is the Advertisements Display Manager **210**. The Screen Saver Subsystem **220** is further described in Microsoft's *Windows-95 Software Development Kit*, including: WIN 32 Overview and reference manuals (chapter 79—Screen Saver Library), available from Microsoft Corp., which is hereby incorporated by reference. On other platforms which do not provide similar functionality, the Advertisements Display Manager **210** itself must monitor for idle time.

The general mechanisms and protocols for communicating with a network, such as the Internet, or on-line service, are known in the art. See, e.g., Stallings, W., *Data and Computer Communications*, Second Edition, Macmillan Publishing Co., (1988). The preferred embodiment utilizes the TCP/IP protocol (Transport Control Protocol/Internet Protocol) which is also well known in the art. See, e.g., Martin J., *TCP/IP Networking*, PTR Prentice Hall (1994). The disclosure of each of the foregoing is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

The TCP/IP Protocol Stack **290** is a set of programs that mediate between application programs and the physical communication line. The TCP/IP Protocol Stack **290** provides application programs with a standard set of function calls for communicating with other application programs connected to the Network **700**. Thus the application programs, here the TCP/IP Polite Agent **280**, do not have to account for the nature of the physical communication line or error correction.

The PPP-TCP/IP Over Modem Protocol **295** module provides the ability to use the TCP/IP protocol over a specific type of physical communication line, i.e. a pair of modems connected over a telephone line. In the preferred embodiment, the functions of the TCP/IP Protocol Stack **290** and PPP-TCP/IP Over Modem Protocol **295** are implemented within the Windows-95 platform and are accessed from the TCP/IP Polite Agent **280** via system calls. See Microsoft Windows Socket Specifications (rev. 1.1), the

disclosure of which is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

c. User Preference and Advertisement Database

The User Preference and Advertisement Database 230 contains various information needed by the system. The primary data stored is the advertisement information (including executable code modules, bitmaps, video clips and sound clips). The database also stores display statistics, configuration information and user preference data.

Typically, the User Preference and Advertisement Database 230 is located on the Mass Storage Device 516, however, in LAN installations, the User Preference and Advertisement Database 230 may be stored on a LAN server. This optimizes storage for the system, since an Advertisement 50 needs to be loaded on the LAN only once and is available for display by each workstation on the LAN.

The User Preference and Advertisement Database 230 is preferably accessed through a well-defined Application Programmer's Interface (API), as is known in the art. In the preferred embodiment, this may be an OLE2 compound file or other database means supplied by third-party software vendors. Thus, the location of the User Preference and Advertisement Database 230 will be transparent to the other processes.

d. User Interface Setup

The User Interface Setup Process 201 allows the user to configure the behavior of the system. This process allows the user to input and view preferences as to advertising categories, as well as local computer configuration data.

Local configuration data typically includes:

- a) The Advertising System Server 600 Internet name or Internet address.
- b) The amount of disk space on the Local Computer 200 or LAN which may be allocated to the advertising system.
- c) The length of time an advertisement should be stored before it is deleted from the Local Computer 200.
- d) The overhead which may be caused by the advertisement transfer process, including the communications line threshold.
- e) Screen saver delay time.
- f) Whether feedback information may be sent to the Network 700.

User preference information typically includes:

- a) Listings of advertisement categories which are to be given high priority, and those categories which are to be banned from being downloaded or displayed. Typical advertisement categories are "SCUBA diving equipment," "Fast food vendors," "Toys for ages 8-14," or "Cigarettes" and the like. The actual list of categories will be provided by the Advertising System Server 600.
- b) Time periods during which sound-only advertisement are to be played.
- c) Whether wallpaper or cursor advertisements are allowed.
- d) Whether animation is allowed.
- e) Time periods and types of foreground activities during which advertisements and feedback information may be transmitted.
- f) Identification of the user's natural language.

Additionally the User Preference and Advertisement Database 230 stores information on the Local Computer's 200 platform capabilities, such as sound boards, server resolution and multi-media capabilities.

The Advertisement Feeder 250 will incorporate the advertising preference information, the Local Computer's platform capabilities, disk space limitations, and other configuration data into its request for new advertisements. The

Advertising System Server 600 uses this information in selecting the next advertisement to be transmitted. The user preference and configuration data may alternatively be stored on the Advertising System Server 600.

The Job Manager 720 on the Network Server 600 matches the user preferences and configuration data against the category information for the available Advertisements 50. Advertisements 50 matching the user's high priority categories and platform capabilities are selected for downloading. Lower priority matches are also selected occasionally on a pseudo-random basis, such that Advertisements 50 within the user's high priority categories will have higher probabilities of being downloaded. The system also allows for increasing the probability of downloading particular advertisements 50. This allows advertisers to pay increased rates to distribute advertisements faster and to a wider range of users.

User's preferably enter their preferences using whatever interfaces are most natural for the underlying platform. In the preferred embodiment, the user enters information through standard Windows-92 dialog boxes.

In one embodiment of the invention, the Advertising Display Manager 210 allows users to respond to Advertisements 50 being presented by selecting a user grading box which allows users to judge the Advertisements 50 on a scale from "do not show me this advertisement again" to "excellent." This information may be used to modify the user preferences, and may also be incorporated into user feedback information and sent to the Advertising System Server 600 for use by the advertisers.

The User Interface Setup Process 201 also allows users to browse through Advertisements 50 stored on their local system (hard disk or local LAN network), as well as those Advertisements 50 available from the Network 700. For Advertisements 50 stored locally, whether downloaded from the Network 700 to the User Preference and Advertisement Database 230 or available on a Local Advertisement Database 550 stored on floppy disk, CD-ROM or like device, the User Interface Setup Process 201 displays a menu with the available Advertisements 50. Each Advertisement 50 stored on the Network 700 or Local Advertisement Database 550 may optionally include a preview segment. The user may have the system present the preview of the Advertisement 50 such as a still image, a short animation sequence, or a sound clip. The user may then select the particular advertisement to be displayed. For Advertisements stored on the Network 700, the Advertising System Server 600 will transmit a list of available Advertisements 50. Users may also disable specific advertisements, or all advertisements of a specific advertiser from being transmitted to the Local Computer 500 or displayed.

e. Advertisement Display Manager

The Advertisement Display Manager 210 selects and displays Advertisements 50 from the User Preference and Advertisements Database 230. The Advertisement Display Manager 210 is typically activated by the Screen Saver Subsystem 220 when the user has not entered input for a predetermined time, or from the Scheduler 265. Alternatively, the user may directly access the Advertisement Display Manager 210 from the platform, such as through selecting an icon or other common method.

The Advertisement Display Manager 210 will display the collection of bitmaps, animation, and sound clips associated with the Advertisement 50. FIG. 8 shows a flowchart of a preferred method of an Advertisement Display Manager 210 in accordance with the principles of the present invention. The Advertisement Display Manager 210 is typically called

by either the Screen Saver Subsystem **220** to display a screen saver type advertisement after the system has been idle for a predefined period or by the Scheduler **265** to modify the background wallpaper or present a sound-only type advertisement on a periodic basis. The Advertisement Display Manager **210** selects and presents the next Advertisement **50** of the specified type to be presented from the User Preference and Advertisements Database **230**.

In the preferred embodiment, the display and other presentation capabilities for each advertisement are self-contained within the Advertisement **50** itself. In this manner the Advertisement Display Manager **210** can support a virtually unlimited number of presentation techniques. The code needed for presenting the advertisement such as a digital sound or video decoder or animation file player, is a resource available from the Resource List **52** within the Advertisement structure **50**. The resource may exist in a number of forms such as executable or interpreted code or scripting code such as that used in Hot Java, available from Starwave Corp. When the resource consists of interpreted or scripted code, the interpreter itself becomes an additional resource which must be made available to the Local Computer. If necessary, the Advertisement Feeder **250** will download this resource to the Local Computer **500**, using the same techniques as used to download other advertisement data.

Many platforms, including the preferred Windows-95 platform, include a multi-media subsystem that provides APIs for playing animation, sound clips, video clips, etc. See Win32 Programmer's Reference Manual, hereby incorporated by reference. Alternatively, there are a wide variety of stand-alone tools suitable for providing such functions on Windows, Macintosh and other platforms.

In the preferred embodiment, each Advertisement **50** will include a small .DLL with an entry point with a pre-defined name. This entry point will be called by the Advertisement Display Manager **210** in order to display the Advertisement **50**. The advertisement entry point is specific for each Advertisement **50**. When the advertisement entry point is called, the particular code needed to present the given Advertisement **50** will be executed.

User interaction with the Advertisement Display Manager **210** is preferably initiated by pressing a predesignated key, for example F10. When the Advertisement Display Manager **210** is active, all user input is routed directly to the Advertisement Display Manager **210**, thus allowing for user interaction with Advertisements **50**. The Advertisement Display Manager **210** selectively forwards certain keys to the default operating system routine, which will typically terminate the Advertisement Display Manager **210**. The user may interact with the Advertisement Display Manager **210** through a number of ways, including answering questioners, initiating a WEB browser to connect directly to an advertiser WEB page on the Network **700**, or automatically initiating a voice connection through the Modem **520** to the advertiser.

Additional aspects of the present invention utilize a variety of techniques for presenting the Advertisements **50**. These techniques include displaying advertising as the background "wallpaper" of the display or modifying the cursor to include an advertiser's logo or other symbol. Additionally, small advertising logos or other advertising content may be placed on the Display Device **513** either at a fixed location on the Display Device **513** or fixed relative to user display windows such that when the user display window is moved on the display the advertisement will move with the window. In the preferred Windows-95 environment, these functions are performed through system calls as described in the Win32 Programmer's Reference Manual, available from Microsoft.

An additional presentation technique is the use of sound-only advertising. The Advertisement Display Manager **210** will make use of a Sound Device **530** on the Local Computer **200**. Any sound devices supported by the platform are suitable. In the preferred embodiment, this includes the Sound Blaster card, available from Creative Labs.

f. Scheduler

The Scheduler **265** keeps track of the list of timing-dependent operations. When the time comes to execute a timing-dependent Advertisement **50**, as for example changing the wallpaper or playing a sound-only Advertisement **50**, the Scheduler **265** notifies the Advertising Display Manager **210**, which performs the required action.

g. Advertisements Feeder

The Advertisement Feeder **250**, is responsible for adding new Advertisements **50** to the User Preference and Advertisement Database **230**. Advertisements **50** preferably are provided from the Internet through the Internet Feeder **270**, however, the Advertisements Feeder **250** is not dependent on the type of advertisement source and may receive Advertisements **50** from other sources, such as commercial on-line services, via other feeder mechanisms and other types of polite agents, as shown by references **271** and **272**, respectively, in FIG. 4.

FIG. 9 shows a flow chart of a preferred embodiment of an Advertisement Feeder **250** constructed in accordance with the invention.

To download a new Advertisement **50**, the Advertisement Feeder **250** first creates a Polite Agent Job **285** to request the Advertising System Server **600** to select the next advertisement for downloading (step **251**). The Advertising System Server **600** selects the next Advertisement **50** to be transferred based on the individual user's preferences and configuration and pricing parameters attached to each Advertisement **50**. The Advertising System Server **600** sends the Resource List **52**, such as executable code modules, bitmaps, animation, sound clips, scripting systems, etc., that the Advertisement **50** needs in order to be presented. The Advertisement Feeder **250** queries the User Preference and Advertisement Database **230** and determines which resources are already available locally, i.e. on the user's PC or LAN. The Advertisement Feeder **250** creates a Polite Agent Job **285** for each resource not in the User Preference and Advertisement Database **230**, requesting the Advertising System Server **600** to download only the necessary resources (steps **252**, **254**). Once the resources have been downloaded, the Advertisement Feeder **250** adds the Advertisement **50** to the User Preference and Advertisement Database **230** (step **253**).

An important part of the functionality of the client system is the ability to resume the transfer of an Advertisement **50** which had been only partially transferred during the previous connection., i.e. the client system is preferably able to re-establish transmission of a file after a break in the Communications Link **703**. Preferably, the client system will resume transmission from the point in the file at which communications was broken off. In the preferred embodiment, this functionality is implemented within the TCP/IP Polite Agent **280** and each Polite Agent Job **285**.

h. Advertisement Killer

The Advertisement Killer **275** periodically scans the User Preference and Advertisements Database **230**, and purges Advertisements **50** that satisfy its purge criteria. Typical criterion include the total time the advertisement has been stored and the number of times displayed. Additionally, Advertisements **50** are purged on user demand through user interaction with the Advertisements **50** or the User Interface Setup Process **201**.

i. Feedback Manager

The Feedback Manager **220** is responsible for sending feedback information to the Advertising System Server **600**. This information includes statistics on displayed Advertisements **50**, including user ratings of specific advertisements and the time and length an advertisement was displayed. The Feedback Manager **220** also transmits information which was gathered from the user during interaction with the Advertisements **50**, such as through games and questionnaires. This feedback information may be used as a basis for calculating the advertiser's charge.

j. Polite Agent Technology

The system incorporates a type of intelligent software agent technology referred to herein as a "Polite Agent." The role of the Polite Agent is to perform communication tasks in the background without imposing a noticeable overhead on the user. FIG. 6 illustrates the preferred embodiment of the TCP/IP Polite Agent **280** utilizing the TCP/IP protocol. The TCP/IP Polite Agent **280** transmits information during periods of low line utilization without causing a noticeable slowdown in the data transfer rate of other processes communicating over the Communications Link **703**. The TCP/IP Polite Agent **280** constantly monitors communications status and determines periods of low communication line utilization. It then uses the TCP/IP communications resources, available on the platform, to transfer a portion of the data. Preferably, the agent does not initiate the communication itself, but rather takes advantage of communications resources once the initial Communications Link **703** with the Network Service Provider **701** has been established, thus avoiding additional user charges.

If the communications resource utilization remains low and ample resources are available the software agent performs its designated data transfer task. Alternatively, if communications resource utilization becomes high due to other applications executing on the Local Computer **500** or the Communications Link **703** is disconnected (e.g., the line goes down), the TCP/IP Polite Agent **280** temporarily suspends its data transfer operation until ample resources are available once again. At that point, the TCP/IP Polite Agent **280** recovers the data transfer process from the point where the transfer was suspended, thereby avoiding the need to retransmit data.

Low line utilization occurs when the communications line is busy no more than a predetermined percentage of time. This threshold may be fixed (typically at 30%) user-configurable, or dynamic. When dynamically determined, the threshold may vary with a number of parameters such as the length of time the TCP/IP Polite Agent **280** has been waiting to transmit, the number or type of Polite Agent Jobs **285** on the Polite Agent Queue **286**, the amount of data which the TCP/IP Polite Agent wishes to transfer, and the type of data being transferred.

Neither the Advertisement Feeder **250** nor the Feedback Manager **240**, directly perform data transfer. Instead, they place Polite Agent Jobs **285** in the Polite Agent Queue **286** which will be called by the TCP/IP Polite Agent **280** when appropriate. The Polite Agent Jobs **285** perform the actual data transfer.

In the preferred embodiment of the invention, the target operating system will be Microsoft Windows-95 utilizing a TCP/IP protocol. Extension of these operations for different protocols or operating systems will be apparent to those of ordinary skill in the art.

In step A (**41**) of the TCP/IP Polite Agent process **280**, a check is made to see if the Communication Link **703** has been established. This can be done in various ways known

to those skilled in the art. A preferred method is to "ping" (send a packet to and receive a response from) the Advertising System Server **600**. See, e.g., J. Martin, *TCP/IP Networking*, PTR Prentice Hall Inc. (1994) (pages 147-48), the disclosure of which is hereby incorporated by reference. An alternative method is to "ping" the Network Service Provider **701**.

In step B (**42**), the line utilization threshold is calculated. As noted above, this calculation may vary in different embodiments of the present invention. Thus, the line utilization may be fixed, user-configurable or dynamic. The threshold calculation also preferably takes into account the load caused by communication generated by the Polite Agent Jobs **285** themselves. This prevents the TCP/IP Polite Agent **280** from not transmitting when the Communications Link **703** is busy primarily due to its own communications.

In step C (**43**), the current communication line utilization is obtained. For TCP/IP under Windows-95, statistical information regarding the communication line utilization is available from the operating system, including such information as bytes/second. In the preferred embodiment, this sampling does not impose a significant overhead on the system and therefore does not cause any noticeable degradation of foreground processes.

In step D (**44**), the current communications line utilization is compared to the calculated threshold. If the current utilization is higher than the calculated threshold, the TCP/IP Polite Agent **280** will not perform communication and will return to step A. At this point the TCP/IP Polite Agent **280** may be temporarily suspended by the operating system.

In step E (**45**), the next Polite Agent Job **285** to be executed is selected. Several Polite Agent Jobs **285** can be pending on the Polite Agent Queue **286**. The TCP/IP Polite Agent **280** will alternate between the Polite Agent Jobs **285** preferably on a round-robin schedule allowing all of the Polite Agent Jobs **285** to execute in turn.

In step F (**46**), the Polite Agent Job **285** is executed. The Polite Agent Jobs **285** are designed in such way that they generate a small amount of communication, for example sending 1K of information, each time they are executed.

FIG. 7 shows a flow chart for a Polite Agent Job **285** embodying the present invention for transmitting data to the Network **700**. The Polite Agent Job **285** first checks (step A) to see whether this is the first time the job has been called by the TCP/IP Polite Agent **280**. If it is the first time the job has been called, the current file position is initialized to the beginning of the file (step B). The file to be transferred is then opened (step C). If the file open is unsuccessful, the Polite Agent Job **285** returns an ERROR flag to the calling TCP/IP Polite Agent **280** which then terminates the Polite Agent Job **285**, removes it from the Polite Agent Queue **286** and marks the job as terminated with error (steps D, E). If the file opens without error, the Polite Agent Job **285** seeks to the current file position in the transferring file and reads the next block of data (steps F, G, H). If no data is left to be read (end of file condition), the Polite Agent Job **285** closes the file and returns a DONE flag to the TCP/IP Polite Agent **280** which terminates the job and marks the Polite Agent Job **285** as completed successfully (steps I, J). If data was successfully read from the transferring file, the Polite Agent Job **285** transmits the name of the file, file position and file block contents to the Network **700** via the TCP/IP Protocol Stack **290** (steps I, K). The Polite Agent Job **285** then updates the current file position and stores it on persistent storage, such as the Local Computer's Mass Storage Device **516** (step L). Finally, the Polite Agent Job **285** closes the transfer file and returns a flag asking the TCP/IP Polite Agent **280** to again schedule and execute this job again.

It will be obvious to one of ordinary skill in the art how to modify the above type of Polite Agent Job 285 for other types of tasks, such as receiving files. The TCP/IP Polite Agent 280 is capable of transferring any type of information file including, executable code, digitized audio or video, and text, for a variety of types of content such as advertisements, news, or weather, etc. The information file transmitted by the Polite Agent Job 285 may consist of a true file type supported by the platform, or, alternatively, any block of data such as a database record.

In one variation of the present invention, the Polite Agent Job 285 receives as an input the current communication line utilization and a line utilization threshold value. The Polite Agent Job 285 uses this threshold to calibrate its operation by calculating how many network packets, bytes, or other units of data may be transferred without increasing the load beyond the line utilization threshold value.

One of the most common types of Polite Agent Jobs 285, is a file-transfer task. Each time the file-transfer task is run it will send (or receive) a small portion of the file, typically between 0.5-K.

Part of the information collected by the present invention may be sensitive in nature, for example, a user's responses to advertising contests may need to be authenticated to prevent fraudulent responses. The system will use known methods such as public key encryption and digital signatures in order to authenticate the information sent by the Local Computer 500. These methods are well known to those in the art.

The technology of the Polite Agent is general and may be applied in other systems and to the transmission of other types of information content such as news and weather, sports scores and stock quotes, software support information and executable updates, and airlines reservations information.

4. Locally Stored Advertisement Database

FIG. 10 shows a variation of the present invention in which a plurality of Advertisements 50 are stored locally to the Local Computer 500 in a Local Advertisement Database 550 on the Local Storage Device 540 or LAN. In this embodiment, the Advertisements 50 are not downloaded from the Advertising System Server 600, but rather are selected from the Local Advertisement Database 550. The Advertisements Feeder 250 selects Advertisements 50 for presentation on the Local Computer 500 from the Local Advertisement Database 550. In this embodiment, the Advertisements Feeder 250 includes the selection functionality described for the Network Job 725 above. Selected Advertisements 50 are loaded into the User Preference and Advertisement Database 230 for display as described above.

It is understood that various other modifications will be apparent to and can be readily made by those skilled in the art without departing from the scope and spirit of the present invention. Accordingly, it is not intended that the scope of the claims be limited to the description or illustrations set forth herein, but rather that the claims be construed as encompassing all features of patentable novelty that reside in the present invention, including all features that would be treated as equivalents by those skilled in the art.

What is claimed is:

1. A method for transmitting data between a computer and a computer network coupled by a communications link, said computer and said computer network including Internet Protocol (IP) communication resources for providing an IP connection between said computer and said computer network, said IP connection configured to provide for non-blocking virtually simultaneous transmission of data for at

least one higher priority processes whereby transmission of data by said higher priority processes results in slower throughput for each of said higher priority processes transmitting at the same time, said method for transmitting data comprising the steps of:

- (a) determining a current communication line utilization rate for the IP connection until the current communication line utilization rate is below a line utilization threshold, the threshold providing for a predetermined maximum amount of delay to said higher priority processes transmitting data over said IP connection, said communication line utilization rate being a function of the rate of data being transferred in at least one direction across said IP connection;
- (b) transmitting a portion of data between the computer network and the computer via the IP connection;
- (c) tracking the remaining untransmitted portion of the data; and
- (d) repeating steps (a)–(c) until the data has been transferred.

2. The method for transmitting data of claim 1 wherein the communications line utilization rate of step (a) is a function of the amount of data being transmitted by said higher priority processes, data transmitted by said method for transmitting data does not affect said communication line utilization rate.

3. The method for transmitting data of claim 2 wherein the line utilization threshold of step (a) is set to zero.

4. The method for transmitting data of claim 1 wherein the line utilization threshold of step (a) is varied.

5. The method for transmitting data of claim 1 wherein the line utilization threshold of step (a) is varied in accordance with the length of time which has elapsed while waiting for the current communication line utilization to fall below the line utilization threshold.

6. A method for transmitting data between a computer and a computer network coupled by a communications link, said computer and said computer network including Internet Protocol (IP) communication resources for providing an IP connection between said computer and said computer network, said IP connection configured to provide for non-blocking virtually simultaneous transmission of data for at least one higher priority processes whereby transmission of data by said higher priority processes results in slower throughput for each of said higher priority processes transmitting at the same time, said method for transmitting data comprising the steps of:

- (a) determining a current communication line utilization rate for the IP connection until the current communication line utilization rate is below a line utilization threshold, the threshold providing for a predetermined maximum amount of delay to said higher priority processes transmitting data over said IP connection, said communication line utilization rate being a function of the rate of data being transferred in at least one direction across said IP connection;
- (b) transmitting a portion of data between the computer network and the computer via the IP connection;
- (c) tracking the remaining untransmitted portion of the data;
- (d) repeating steps (a)–(c) until the data has been transferred; and
- (e) tracking the amount of information transmitted in step (b), the tracked amount of information transmitted is used to adjust a user's charge for accessing the computer network.

7. A process for transmitting data between a computer and a computer network coupled by a communications link, said communications link further configured to transmit data for one or more higher priority processes, said process for transmitting data comprising the steps of:

- (a) determining a current communication line utilization rate for data transmitted by said higher priority processes over the communications link until the current communication line utilization rate is below a line utilization threshold;
- (b) transmitting a portion of data between the computer network and the computer;
- (c) tracking a remaining untransmitted portion of the data; and
- (d) repeating steps (a)–(c) until the data has been transferred.

8. A system for downloading advertisement items from a remote network to a local computer, said system comprising: one or more primary applications processes communicating between said local computer and said remote network;

- a two-way communications link coupling said remote network and said local computer, said communications link configured to transmit data between said remote network and said local computer for a plurality of processes, including said one or more primary applications processes, said communications link having a line utilization rate being a function of the amount of data currently transmitted by said one or more primary applications processes;

a database containing user preference information;

an IP network service provider coupling said local computer to said remote network, said network service provider providing two-way communication between said remote network and said local computer;

an advertisement system server coupled to said remote network, said advertisement system server containing a plurality of advertisement items; and

a software agent, said software agent initiating the transfer of said advertisement items from said advertisement system server to said local computer while said communications link line utilization is below a threshold value.

9. The system for downloading advertisement items of claim 8 further comprising:

a means for tracking the amount of information transmitted between said advertisement system server and said local computer,

said amount of information transmitted is used to adjust a user's charge for accessing the remote network.

10. The system for downloading advertisement items of claim 8 further comprising:

a means for inputting user preference information; and
a means for selecting a plurality of selected advertisement items from said plurality of advertisement items stored on said advertisement system server, said means for selecting a plurality of selected advertisement items utilizing said input user preference information,
said software agent configured to transmit said selected plurality of selected advertisement items to said local computer.

11. The system for downloading advertisement items of claim 8 wherein said means for selecting a plurality of selected advertisement items is a software process executing on said advertisement system server.

12. A process for transmitting data between a client computer and a server computer coupled by a communications link on a computer network, said communications link configured to transmit data for one or more higher priority processes, said process for transmitting data comprising the steps of:

- (a) monitoring the amount of data transmitted by said higher priority processes to calculate a current communication line utilization rate for said communications link, said calculation excluding data being transmitted by said processes for transmitting data;
- (b) comparing said communication line utilization rate to one or more preestablished values;
- (c) calibrating the amount of data to be transmitted based on said comparison of said communication line utilization to said one or more preestablished values;
- (d) transmitting said calibrated amount of data; and
- (e) repeating steps (a)–(d) until the data has been transferred.

13. The process of claim 12 further comprising the step of: monitoring the communication link until the communication link coupling the first computer and second computer has been established.

14. The process of claim 12 wherein said data comprises executable code.

15. The process of claim 12 wherein said step (a) of monitoring the communications line utilization comprises the step of sampling the line utilization.

16. The method for transmitting data of claim 12 wherein said one or more preestablished values of steps (b) and (c) are varied with the length of time the process has been transmitting.

17. A process for transmitting data between a client computer and a server computer coupled by a communications link, said communications link being configured to transmit data for one or more higher priority processes, said process comprising the steps of:

- (a) monitoring data transmitted by said high priority processes to determine a communication line utilization rate for said communications link;
- (b) utilizing said monitored line utilization rate to determine whether to transmit data in a current iteration of said process;
- (c) if said determination of step (b) indicates data should be transmitted in the current iteration of said process, transmitting a portion of the data; and
- (d) repeating steps (a)–(c) for a new iteration until the data has been transmitted.

18. A process for transmitting data between a server computer and a local computer, said local computer coupled to said server computer by a communications link on a computer network, said communication link further configured to transmit data for one or more high priority processes, said communication link providing a plurality of logical connections allowing said process and said high priority processes to contemporaneously transmit information, said process comprising the steps of:

- (a) monitoring whether said one or more high priority processes are transmitting data over said plurality of logical connections;
- (b) if no high priority process is transmitting data over said plurality of logical connections, transmitting a portion of said data via said communication link; and
- (c) repeating steps (a) and (b) until said data has been transferred.

19

19. A method for transmitting a file between a local computer and a remote computer, said local computer and remote computer coupled by a communication link having a current communication link usage rate, the method comprising the steps of:

- (a) determining a communications status for said local computer, said communications status related to said current communications link usage rate;
- (b) transmitting a portion of said file over said communication link when the said communications status is beyond a threshold level;
- (c) tracking a remaining untransmitted portion of the file, said tracking providing tracking information for said remaining untransmitted portion of the file;
- (d) storing said tracking information indicating the last transmitted portion of said file; and
- (e) repeating steps (a)–(d) to transmit data from the last transmitted portion of said file after any intervening breaks in said communication link or breaks in the availability of said local computer until the file has been transferred.

20. A method for controlling the transmission of data between a local computer and a remote computer, the local computer and remote computer coupled by a communication link having a current utilization rate, the method comprising:

- (a) transmitting a portion of said file over said communication link when the said communications link current utilization rate is below a threshold level;
- (b) tracking the remaining untransmitted portion of the file, said tracking providing tracking information for any remaining untransmitted portion of the file;
- (c) storing said tracking information indicating the last transmitted portion of said file; and
- (d) repeating steps (a)–(c) to transmit data from the last transmitted portion of said file after any intervening breaks in said communications link or breaks in the availability of said local computer until the file has been transferred.

21. A method for controlling the transmission of data between a local computer and a remote computer, the local computer and remote computer coupled by a communication

20

link, the communication link having a current utilization rate, the method comprising:

- providing an indicator signal having at least a first state and a second state;
- allowing the transmission of data on said communication link when said indicator signal is in the first state;
- preventing the transmission of data on said communication link when said indicator signal is in the second state; and
- moving the indicator signal between said states in response the predetermined conditions associated with said communications link, said indicator signal being in said second state when the current utilization rate for the communication link is above a predetermined threshold.

22. A process for transmitting a file between a local computer and a remotely located computer network coupled by a communication link, said process for transmitting a file comprising the steps of:

- (a) monitoring the communication link until the communication link coupling the local computer and the remote network has been established;
- (b) determining a current communication line utilization rate for the communication link until the current communication line utilization rate is below a preestablished line utilization threshold;
- (c) transmitting a portion of the remaining file between the network and the local computer in response to the current communication line utilization rate below the preestablished threshold;
- (d) tracking the remaining untransmitted portion of the file, said tracking providing tracking information for any remaining untransmitted portion of the file;
- (e) storing said tracking information indicating the last transmitted portion of said file, said tracking information being stored in persistent memory; and
- (f) repeating steps (a)–(e) until the file has been transferred.

* * * * *

EXHIBIT C

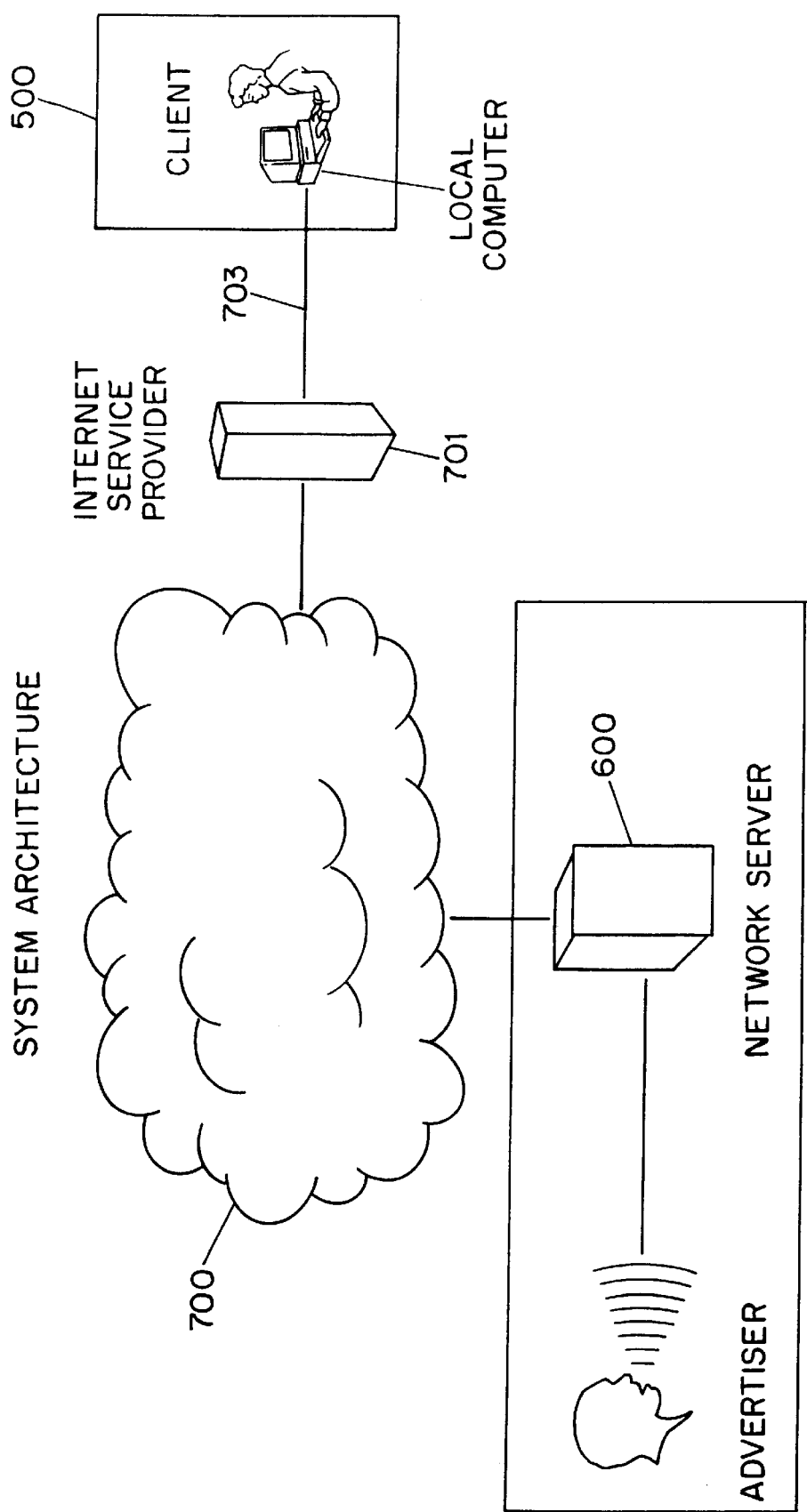


FIG. 1

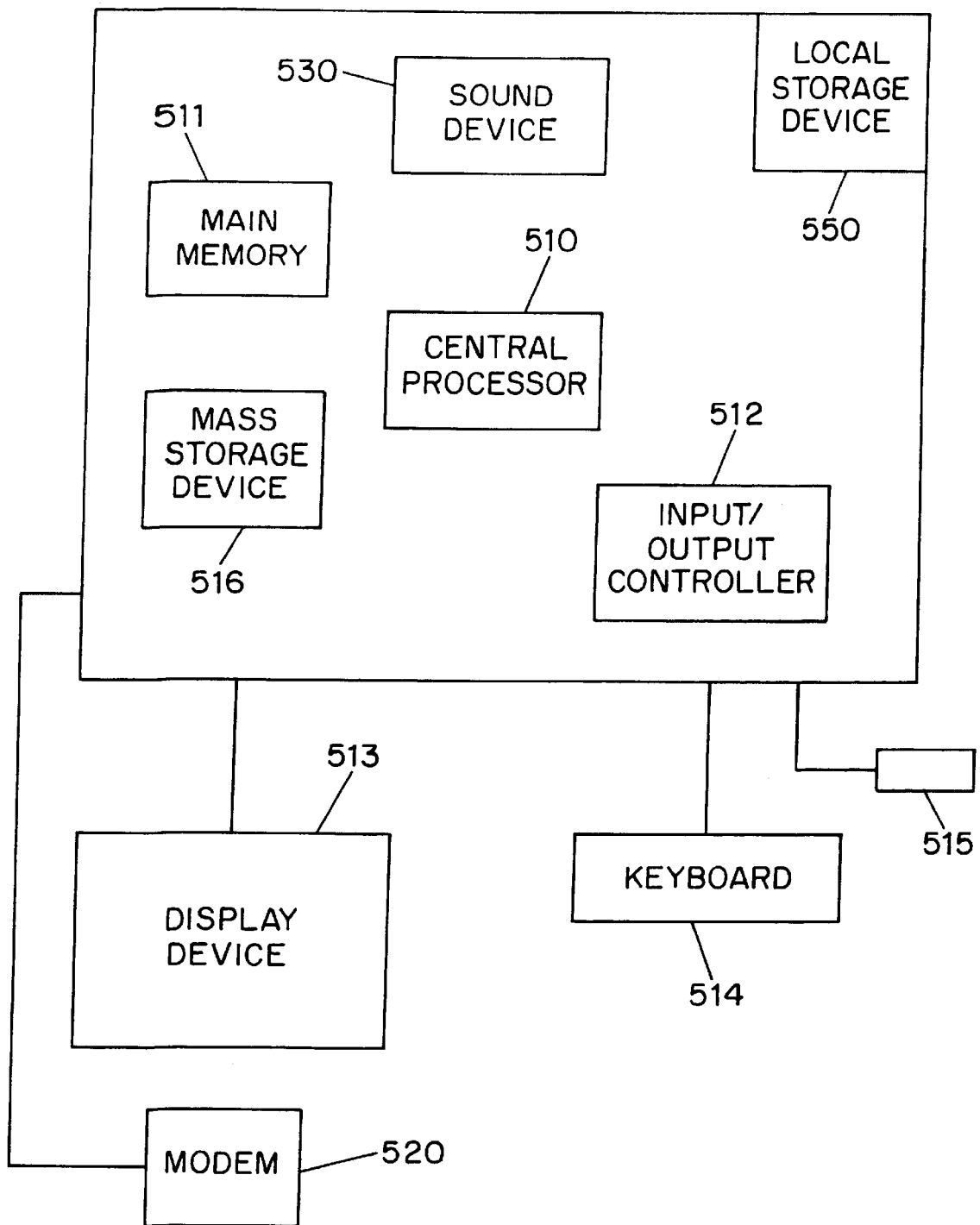


FIG. 2

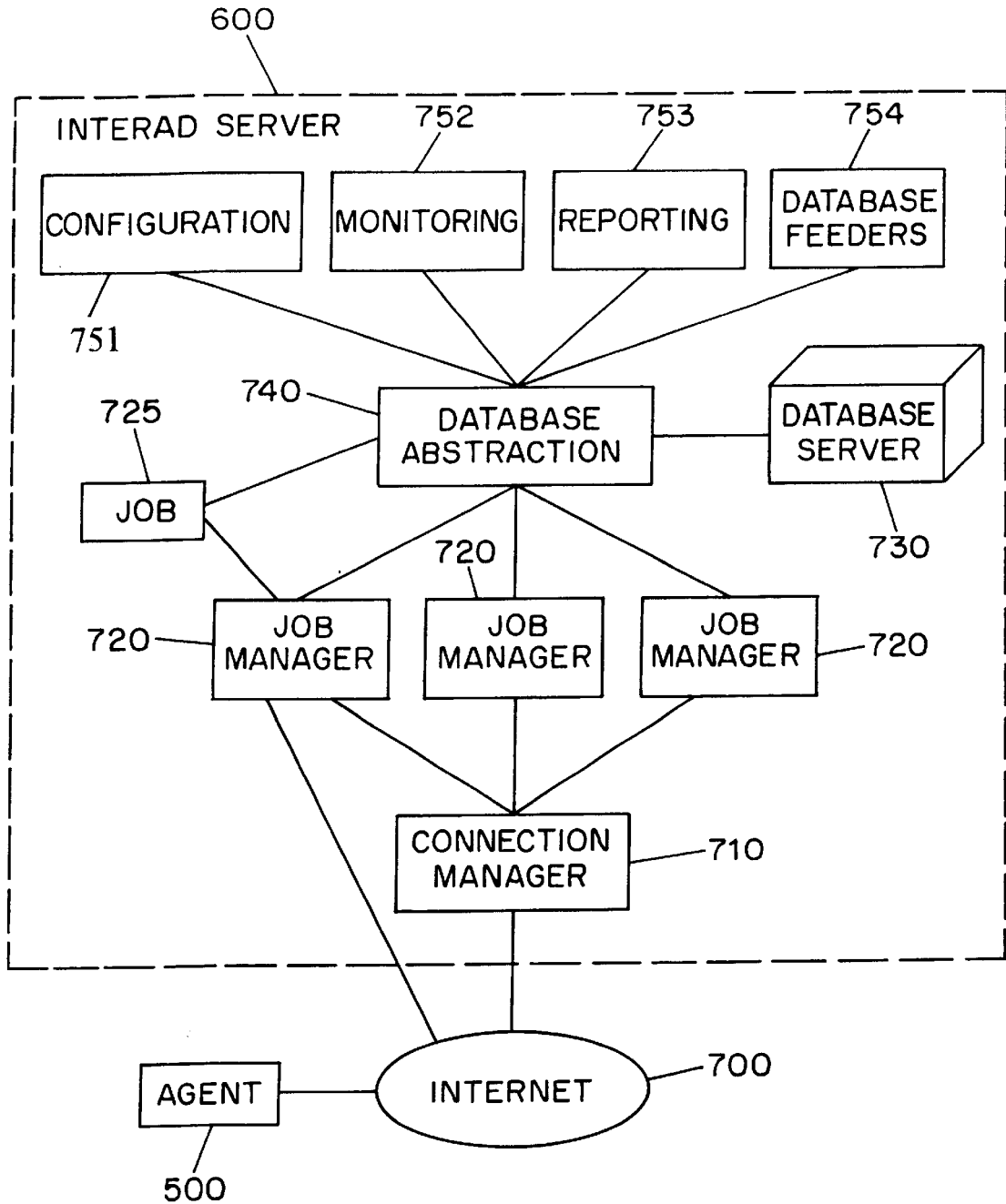
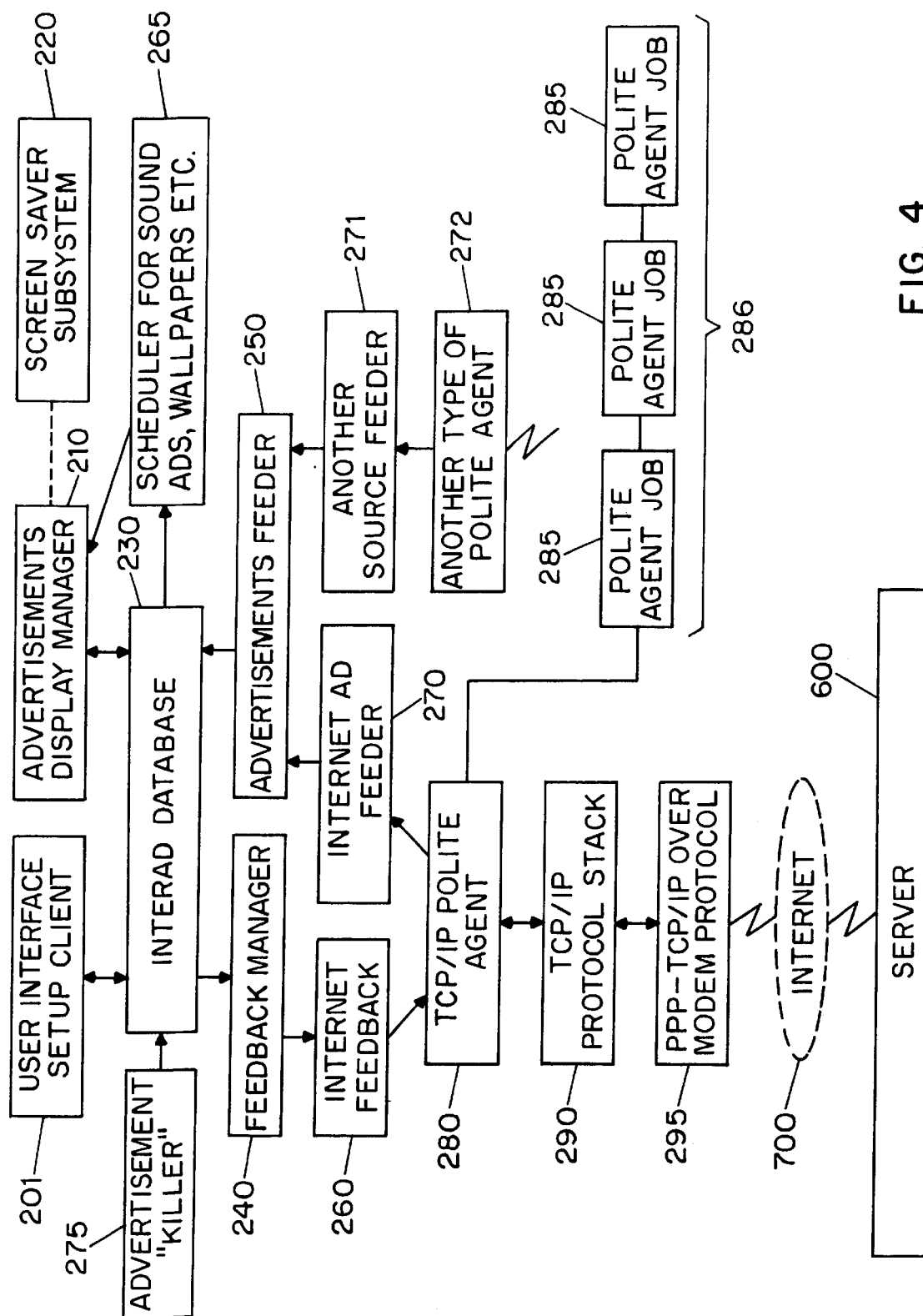


FIG. 3



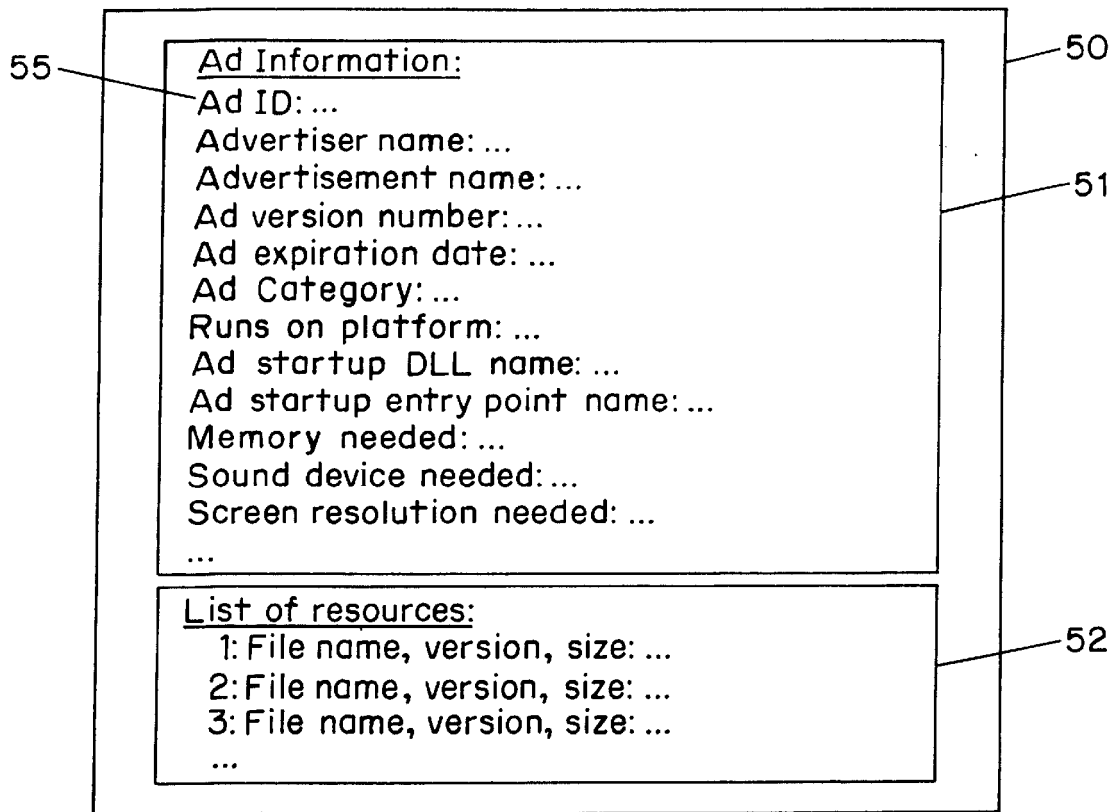


FIG. 5

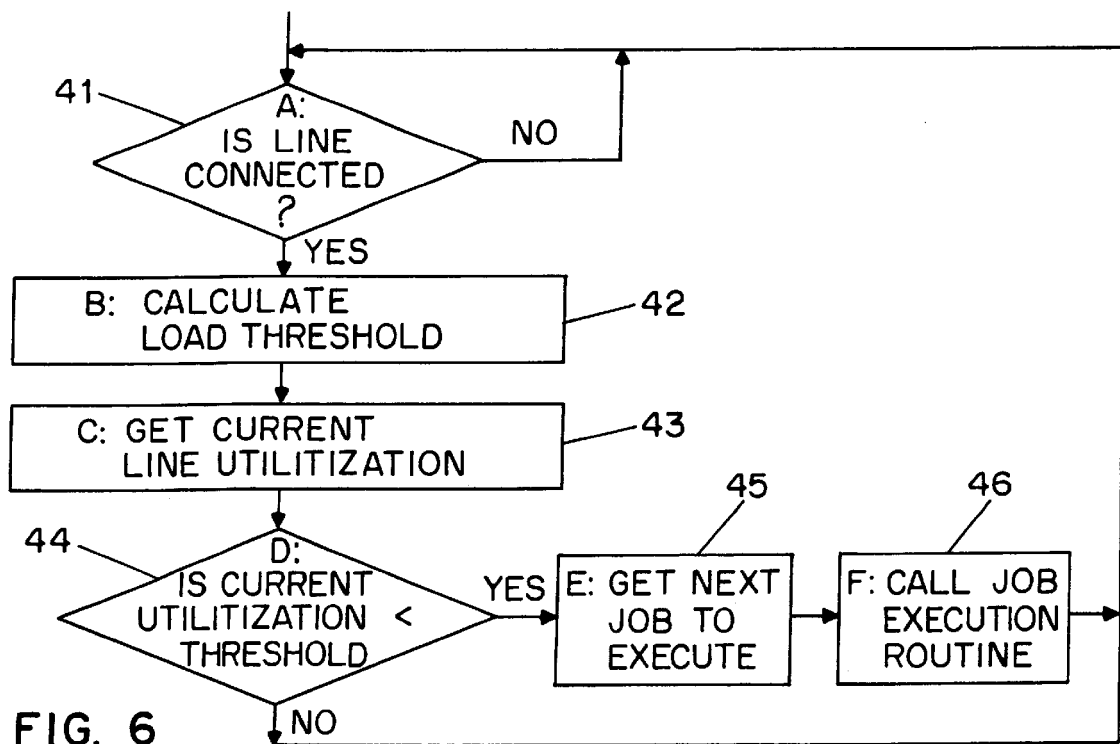


FIG. 6

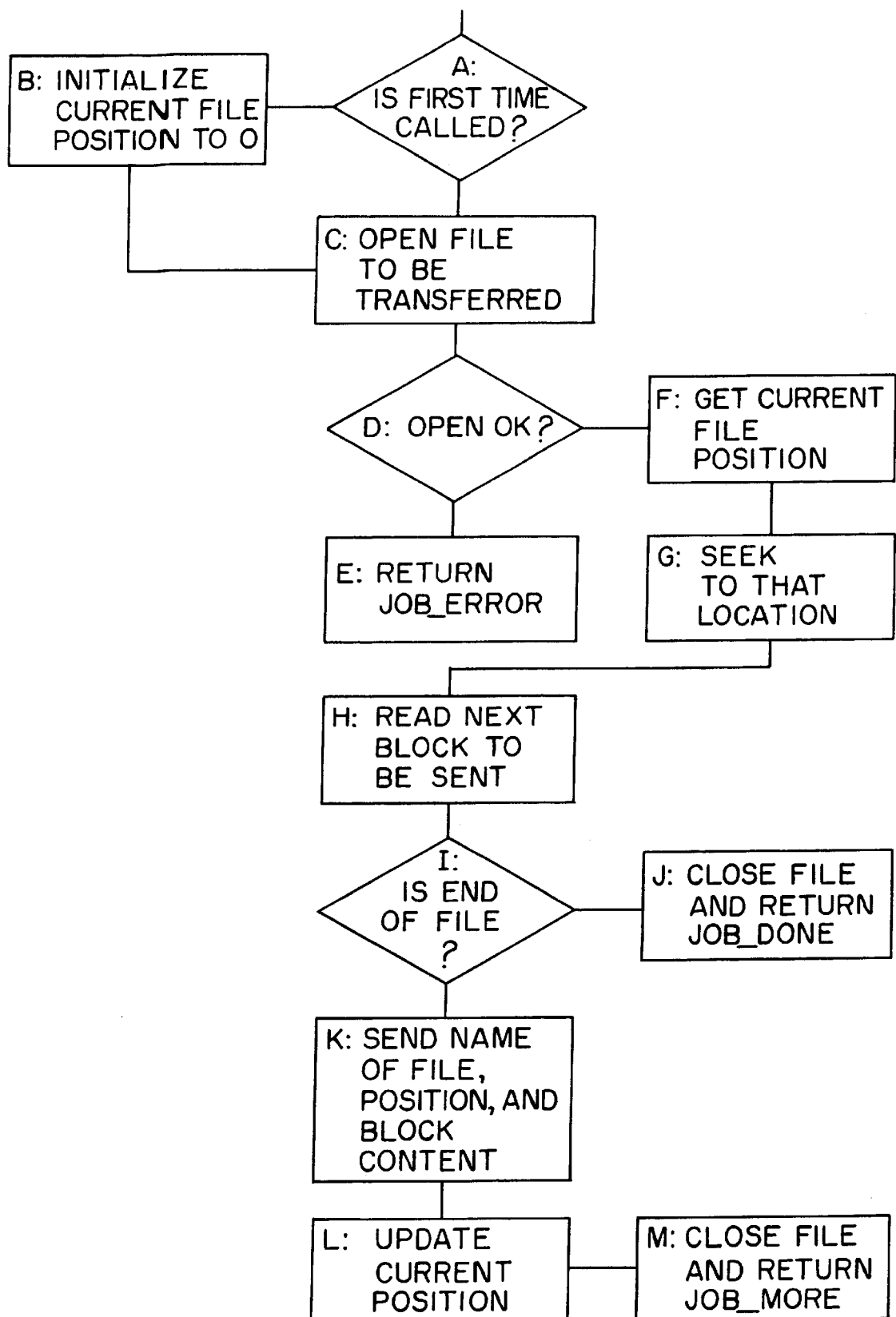


FIG. 7

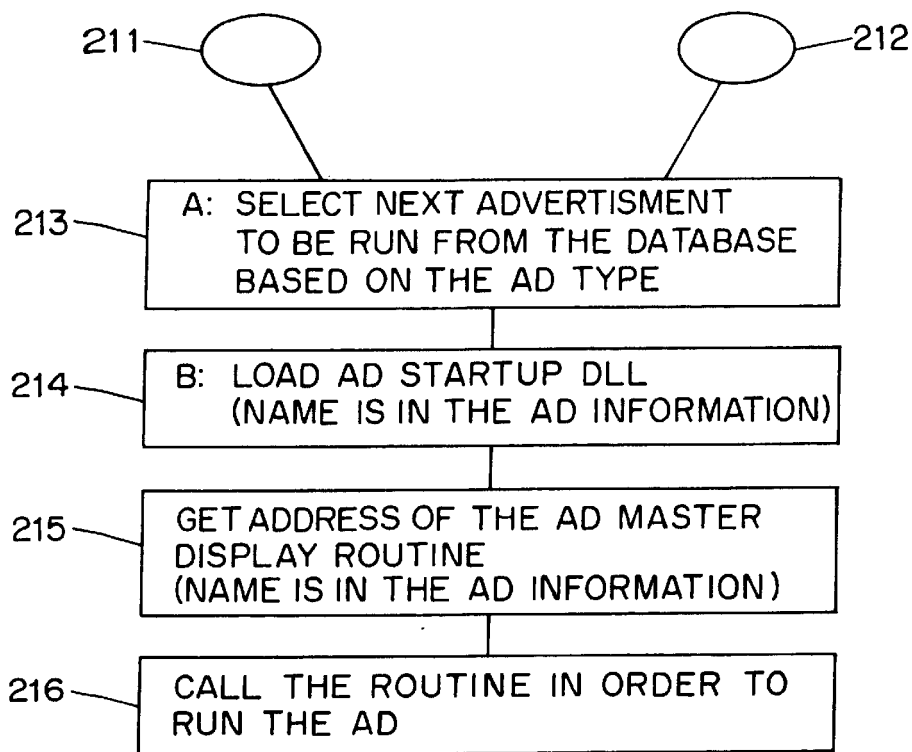


FIG. 8

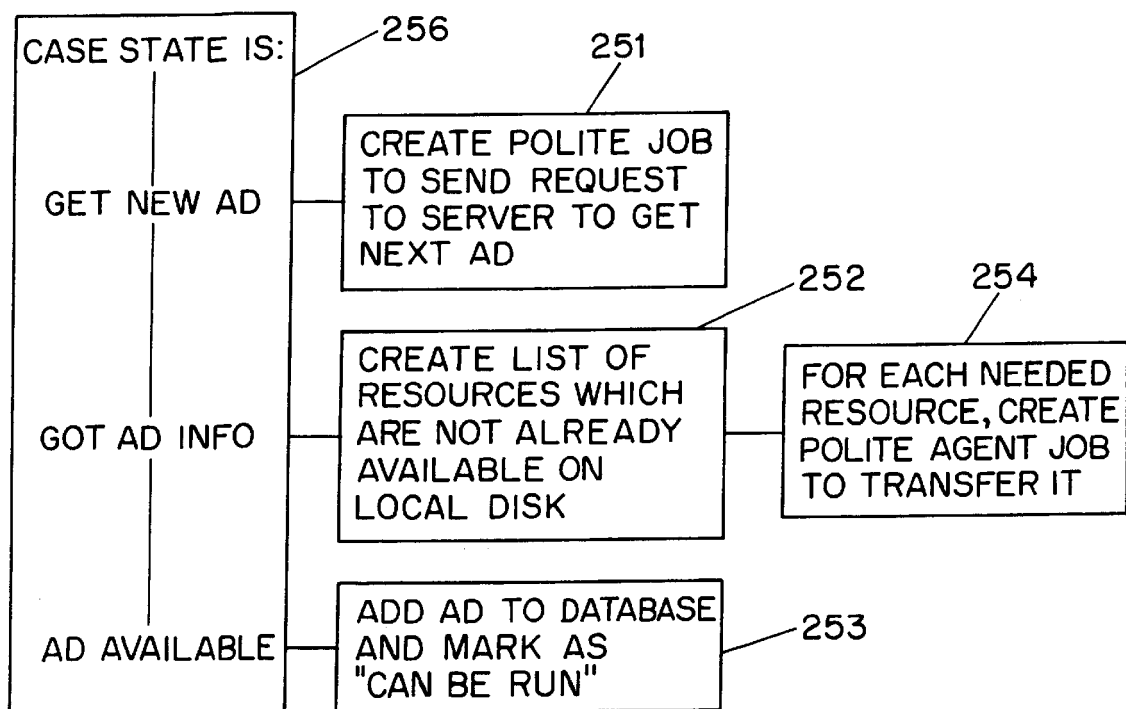


FIG. 9

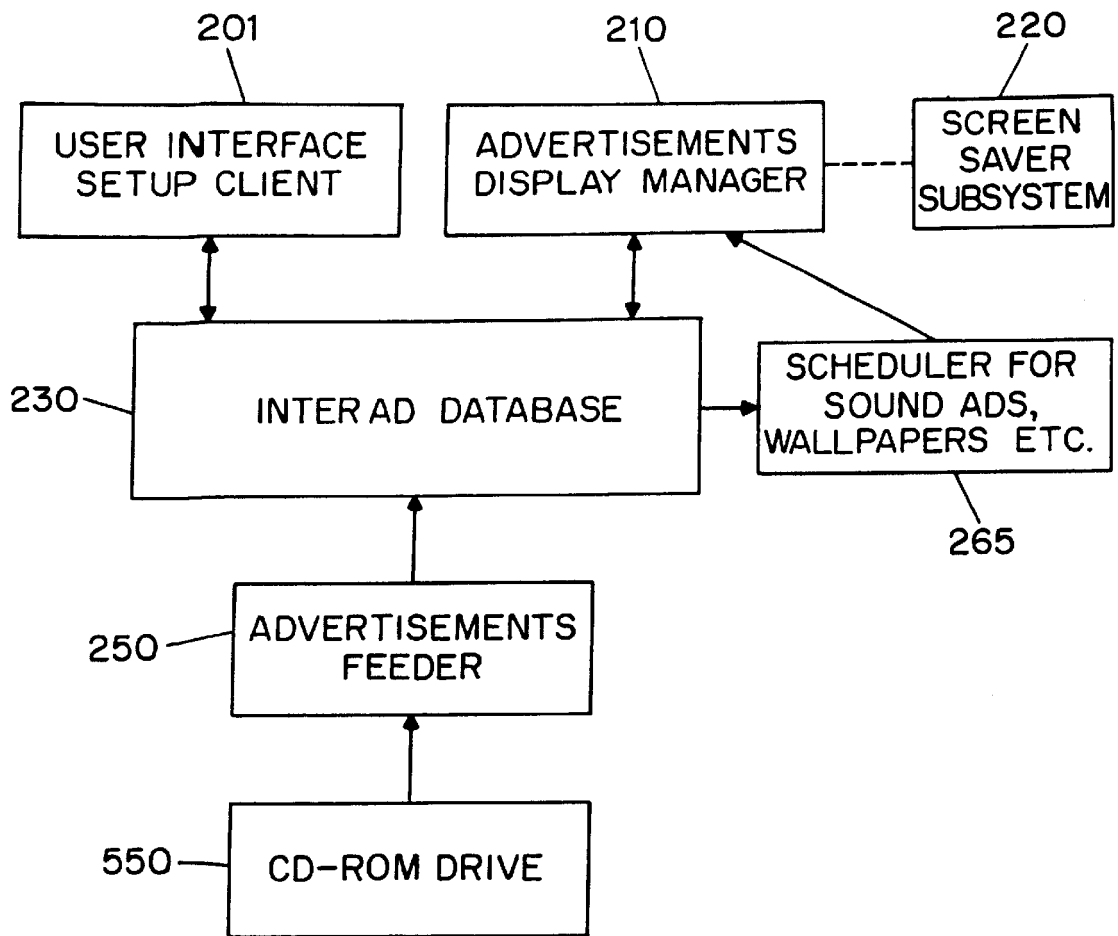


FIG. 10

METHOD AND APPARATUS FOR TRANSMITTING AND DISPLAYING INFORMATION BETWEEN A REMOTE NETWORK AND A LOCAL COMPUTER

CROSS REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. application Ser. No. 09/274,612, filed on Mar. 23, 1999, now U.S. Pat. No. 6,317,789; which is a continuation of U.S. application Ser. No. 08/517,666, filed on Aug. 22, 1995, now U.S. Pat. No. 5,913,040.

FIELD OF THE INVENTION

This invention relates generally to advertisement computer display systems and more particularly to a method and system for displaying advertisements and other information on a computer based on general user selected criteria and transmitting such information from a remote network to the local computer.

BACKGROUND OF THE PRESENT INVENTION

There are two major forms of advertising which are currently being employed on the Internet and commercial on-line services. One form is the use of a small advertisement on WEB pages which are commonly accessed. For example, a portion of the screen display for WEB pages used to access Internet searches may include a corporate logo or other advertisement material. Typical of this style of advertising is the Netscape™ Internet Browser software available from Netscape Communications Corporation of Mountain View Calif., which presents a box containing logos for various corporations on the computer display when the user performs a search. This form of advertising, however, is not very sophisticated and does not encourage user interaction.

Another form of advertising on the Internet is the creation of WEB "pages" or sites by advertisers. One variant of the use of a WEB page displays advertisements in a portion of the viewing area. A second variant, often used by corporate or other advertisers, is the use of WEB sites which employ attractive graphics in the hope of having the user interact with various advertising schemes. In addition, product ordering is usually available from these WEB sites. In most cases, users access these WEB sites by one of the following methods: knowing the Internet address; keyword searching; linking from a different WEB site; through an electronic shopping mall type site; through other advertisements on the Internet; or through the use of programs known as search browsers.

Current advertisers have attempted to improve the attractiveness of these WEB pages by including the use of sound, animated or rotating logos or pictures, and scrolling information. One system, Hot Java, available from Starwave Corp of Bellevue, Wash., supports the execution of small applications programs written in a specific programming language executing within the browser on the local computer. This allows the WEB pages to provide richer content, such as animation or scrolling sports scores across a user's computer display, and better interaction with users. These effects, however, are only available while the user is viewing the specific WEB page incorporating the Hot Java technology.

Despite the fast and furious growth in this advertising sector, WEB sites are still regarded as "passive" advertising

used predominantly for a corporate image rather than for selling products. Specifically the following drawbacks describe the current state of advertising on the Internet: transmitting the advertising information consumes a large amount of the bandwidth of the communications link between the user's computer and the network; access is initiated by the user rather than the advertiser; the user rather than the advertiser pays for access; accessing a site is a time consuming "hit or miss" process; and the process may improve the corporate image but creates little product demand.

U.S. Pat. No. 5,105,184 to Pirani et al. ("Pirani") discloses a system integrating commercial advertisements with computer software. The system discloses integrating commercial advertisements with different types of screens. Pirani, however, does not provide for any user input at the local computer as to the types of advertisements which are to be displayed. Thus, users would be forced to view numerous advertisements of which they are likely to have no interest. This will attenuate the users attention to the advertisements and decrease their effectiveness.

As noted above, a significant problem with current methods for advertising on computer networks is the consumption of significant portions of the bandwidth of the communications link between the user's computer and the computer network. Prior systems have attempted to utilize essentially unused time in telephone networks to deliver advertising or other information. U.S. Pat. No. 5,321,740 to Gregorek, et al. ("Gregorek") discloses a marketing system over an existing telephone network which modifies a portion of the call processing system to play an informational announcement in place of the usual ringback or busy signals. Gregorek differs from the present invention in a number of ways, including the fact that it does not provide any means for interacting with computers over a computer network. Also, Gregorek delivers the informational announcement only during a short splice of time when the user is waiting for callback information.

Current file transfer protocols, such as the File Transfer Protocol ("FTP") and the Trivial File Transfer Protocol ("TFTP"), for transferring files from a remote network, such as the Internet, via a communications link to a local computer are designed to transfer files as quickly as possible. Each computer process executing such a protocol attempts to make maximum use of the available communication resources. This leads to interference and an inevitable slowing down of other computer processes attempting to communicate over the communications link. There exists a need, therefore, for a file transfer process which is designed to behave as a background task and have a minimal impact on foreground communications.

There also exists a need to utilize the computer to display locally stored advertisements. Several software products provide "yellow pages" on CD-ROMs or other media such as floppy disks. The user may use these yellow pages to search for products or advertisers by name or description. This system of advertising is limited, however, in that it requires the user to actively search for advertisers or products and therefore do not spontaneously display products to the user.

Microsoft Windows interface provides a rudimentary form of spontaneous advertising by incorporating a Microsoft Windows logo as an option in its screen saver utility. This system, however, offers only a single advertisement in response to a user's response and therefore does not offer a variety of periodically changed advertisement content based on a user's interests.

SUMMARY OF THE INVENTION

The object of the present invention is to provide a process for transmitting an information file between a local computer and a remote computer network over a communications link with minimal interference to other processes executing on the computer which are also transmitting over the communications link.

It is a further object of the present invention to provide a method and system of presenting individualized advertisements and other informational messages on a computer by allowing a user to select from a variety of advertisement or informational categories.

It is a further object of the present invention to provide a method and system of downloading and presenting individualized advertisements and other informational messages from a network to a local computer from a remote network to a local computer based on a user's selection of advertisement or informational categories.

It is a further object of the present invention to provide such a method and system of downloading and presenting individualized advertisements and other informational messages from a network to a local computer with minimal interference with other data being transmitted between the network and the local computer.

In one variant of the present invention, all advertisements or other informational messages originate on a network server which is accessed via the Internet or alternate on-line method. Select advertisements are transparently downloaded from the network server and stored locally on the user's local computer using a novel type of software referred to herein as a "Polite Agent." In a second variation, the entire advertisement database is locally stored on the local computer or a removable media such as CD-ROM. Manipulation and display of the advertising message is performed by software residing on the user's PC in accordance with preconfigured user preference information.

The advertisement is preferably displayed during idle time as a screen saver utility when the computer is not receiving keyboard input or updating the user's display. Other techniques for displaying the advertisement, such as periodic audio-only messages, screen background wallpaper, cursor modifications, and display in a window on the user's computer display are also available.

Users may enter their preferences by directly choosing categories of advertising or other informational content which most interest them or through interactive games and quizzes. Users may directly respond to advertising messages by participating in contests, requesting further product information, or ordering the advertised product. The advertisements are made attractive to the user by employing a variety of video, animation, sound or any other multimedia effects. Content may be based on an interactive theme such as a contest or special discount offers for on-line customers.

The system monitors the user's interaction with the advertisements and produces raw data on how many times a particular advertisement was accessed as well as the user's response to advertisements. All pertinent information is stored and sent back to a network server where it is made available to the advertisers. User requests for additional information may be directed to the advertiser itself or to the advertiser's WEB site on the network.

The system further comprises the use of a background software process, the Polite Agent, for transferring information between the network and the local computer. The Polite Agent monitors the communications link between the net-

work and the local computer and transfers small portions of the information when the communications link utilization rate is low. In this manner the Polite Agent avoids significant interference with other communications applications transmitting over the communications link. The Polite Agent may also be utilized to transmit other types of information content, such as news, weather, stock quotes, sports scores, software updates or trip reservation information.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, reference is made to the following Detailed Description taken in conjunction with the accompanying drawings in which:

FIG. 1 is a functional block diagram of a system architecture in accordance with the present invention;

FIG. 2 is a functional block diagram of a local computer and its related components in accordance with the present invention;

FIG. 3 is a functional block diagram of the software architecture components of the advertising system network server in accordance with the present invention;

FIG. 4 is a functional block diagram of the software architecture components of the local computer in accordance with the present invention

FIG. 5 is a schematic representation of an advertisement file in accordance with the present invention;

FIG. 6 is a flowchart illustrating a method for a polite agent for communicating information with a remote network in accordance with the present invention;

FIG. 7 is a schematic representation of a job for transmitting data in accordance with the present invention;

FIG. 8 is a flowchart illustrating a method for an advertisement display manager constructed in accordance with the present invention;

FIG. 9 is a flowchart illustrating a method for an advertisement feeder for downloading advertisements from a network constructed in accordance with the present invention; and

FIG. 10 is a functional block diagram of an advertising system for displaying a local database of advertisements constructed in accordance with an alternative embodiment of the present invention.

DETAILED DESCRIPTION

Preferred embodiments of the present invention will now be described with continued reference to the drawings.

System Architecture

1. Network Architecture

FIG. 1 shows an overall view of a preferred embodiment of the system architecture. The Local Computer 500 is physically connected to the Network Service Provider 701 via a Communications Link 703. The Network Service Provider 701 provides access to the Network 700. Advertising System Server 600 is one of the nodes on the Network 700.

a. Local Computer:

As shown in FIG. 2, the Local Computer 500 preferably includes a Central Processor 510, a Main Memory 511, an Input/Output Controller 512, a Display Device 513, input devices such as a Keyboard 514 and a Pointing Device 515 (e.g., mouse, track ball, pen, slide pointer or similar device), and a Mass Storage Device 516. These components communicate through a system bus or similar architecture.

Additionally, the Local Computer 500 is preferably connected to an internal or external Modem 520 or like device for communication with the Network 700. Alternatively, the Local Computer 500 may be connected via an ISDN adapter and an ISDN line for communications with the Network 700. The Modem 520 optionally allows for the establishment of voice calls through software control.

The Local Computer 500 preferably also includes a Sound Device 530. The Local Computer 500 may also include a Local Storage Device 540 such as a floppy disk, CD-ROM or like device for local storage of the Local Advertisement Database 550.

The Local Computer 500 is preferably under the control of a multi-process operating system including a TCP/IP interface, and most preferably operated under the Microsoft® Windows-95 platform available from Microsoft Corporation of Redmond, Wash. However, the present invention may be embodied on a variety of different platforms, including Macintosh, UNIX, NextStep, MS-DOS, and the like.

b. Network

The Network 700 is preferably the World-Wide Internet. The World-Wide Internet ("Internet") is a world-wide network connecting thousands of computer networks. The dominant protocol used for transmitting information between computers on the Internet is the TCP/IP Network Protocol. Computers connect to the Internet use either a fixed connection, in which case they become a "permanent" node on the Internet, or a dial-up connection, in which case then act as a node on the network as long as the connection is active. Internet addresses are the numbering system used in TCP/IP communications to specify a particular network or computer on the network with which to communicate.

The invention may also be practiced with commercial on-line services such as America Online, available from America Online Inc., CompuServe, available from H&R Block Inc., Prodigy, available from Prodigy Services, Microsoft Network, available from Microsoft Corp., as well as other like services from a variety of companies such as AT&T Corporation and MCI Communications Corp.

c. Network Service Provider

The Network Service Provider 701 provides access to the Network 700. Commercial providers include: BBN, Netcom, and Prarienet.

d. Advertising System Server

The system preferably includes at least one Advertising System Server 600. The main roles of the Advertising System Server 600 are to store Advertisements 50, transfer the Advertisements 50 to the Local Computer 500, and collect user feedback. The Local Computer 500 will initiate communication with the Advertising System Server 600. Each user is assigned a unique user-ID which can not be changed by the user. This user-ID is used by the Advertising System Server 600 to track each user's activity, including which Advertisements 50 have been downloaded to the user.

When the Local Computer 500 connects to the Advertising System Server 600, the Local Computer 500 will upload the user's user-ID and the configuration and user preference information to the Advertising System Server 600. The Advertising System Server 600 uses this information to select the next Advertisement 50 to be downloaded. The Local Computer 500 also may directly request a specific advertisement through the use of a unique Advertisement-ID 55 assigned to each Advertisement 50. If feedback information has been collected it also will be uploaded when the Local Computer 500 connects to the Advertising System Service.

In an alternate embodiment of the present invention, the selected advertisement may be stored on any one of the plurality of advertising system servers connected to the Network 700. In this embodiment, the Local Computer 500 initiates communication with a predetermined advertising system server. The predetermined advertising system server will select the next Advertisement 50 to be downloaded and transmit the network address of the advertising system server storing the Advertisement 50. The Local Computer 500 uses the transmitted network address to request the selected Advertisement 50 from the appropriate advertising system server.

The Advertisements 50 stored on the Advertising System Server 600 may be translatable to one or more natural languages. The Advertising System Server 600 will use each user's native language-ID to transmit the appropriate natural language version of the Advertisement 50.

2. Advertising System Server Software Architecture

a. Software Modules

FIG. 3 is a functional block diagram of a software architecture of the Advertising System Server 600 embodying the invention.

The Local Computer 500 initially connects to the Connection Manager 710 which is responsible for allocating an available Job Manager 720 and returning its address to the agent. The Job Manager 720 then handles all further conversation with the Local Computer 500. As noted above, the architecture optionally allows for a plurality of advertising system servers each running a plurality of Job Managers 720. Thus, the Job Manager address returned to the Local Computer 500 may contain both the Internet address of the server and the process identification. The Job Manager 720 identifies and authenticates the user-ID against the Server Database 730.

The Job Manager 720 creates a Network Job 725 for each user it communicates with. Each Network Job 725 communicates with the Local Computer 500 to select and download Advertisements 50; collect feedback from the Local Computer 500; check the user's participation and any awards from contests, etc.; and upgrades and installs the Local Computer 500 software versions. The Network Job 725 is responsible for selecting the next downloaded Advertisement 50 based on user configuration and preference data as described herein.

Under most current network models, including the current implementation of the Internet, users are typically charged based on the amount of time they are connected to specific resources on the network. Thus, the current system of downloading advertisements and other information in the background does not increase the cost to the user, as transmission occurs in background mode while the user is already connected to the network Service Provider 701. Future implementations of these networks, however, may charge users based on the amount of information, or number of network "packets" or other units of data, the user has received. On such networks, the system should be able to track the amount of information transmitted, such as by counting the number advertisements, advertisement resources, or network packets (also known as "datagrams"), frames, segments or other units of network data containing advertisements. The Network Service Provider 701 may use this information to charge the system generated transmissions to the advertisers rather than the users. The Advertisement System Server 600, and most preferably the Network Job 725, will be responsible for tracking the amount of information transmitted by the system. Alternatively, the TCP/IP Polite Agent 280 or other software process on the Local Computer 500 will track this information.

The Server Database 730 contains the system information, including: the Advertisements 50 or other informational content; listings of users; listings of advertisers;

listings of network service providers; billing information; audit logs and statistics. The Server Database 730 also maintains information on active connections and their activity. The Server Database 730 is accessed through the Database Abstraction 740 interface which provides a layer of interface for all modules on the Advertising System Server 700.

In addition to providing Advertisements 50 and other informational content to local computers, the Advertising System Server 600 also provides various management services, such as billing information, viewing and gathering statistics on feedback information, and advertisement display audit-logs which may be sorted according to various filters such as advertiser or advertising category. The Advertising System Server 700 includes various other software administration tools for maintaining the system, including: a Database Feeder 754 for modifying the Server Database 730; Monitoring Tools 752 for viewing the activity of the system; Configuration Tools 751 for modifying the behavior of the system; and Reporting Tools 753 for creating reports concerning the system.

b. Advertisement Records

FIG. 5 shows a schematic representation of an Advertisement 50. Each Advertisement 50 in the Server Database 730 comprises an Advertisement Information Record 51 and a Resource List 52. The Advertisement Information Record 51 contains information identifying the advertisement (including the Advertisement-ID 55), its category, its size, and the hardware required to display the advertisement, such as sound boards, screen resolution and multimedia requirements. The Resource List 52 contains a list of resources (bitmaps, animations, digitized audio segments, executable code, etc.) that must exist on the Local Computer 500 or associated local LAN in order to present the advertisement. The Resource List 52 includes a unique resource-ID, a resource type, and a resource pointer. The resource pointer identifies a file, a database record, a block of data, or other means of identifying the resource. In this manner, resources can be shared by various Advertisements 50.

3. Local Computer Software Modules

a. Software Architecture

FIG. 4 is a functional block diagram of the software modules and processes of the software architecture for a preferred embodiment of the invention on the Local Computer 500, including a User Interface Setup Process 201, Advertisements Display Manager 210, Screen Saver Subsystem 220, User Preference and Advertisements Database 230, Feedback Manager 240, Advertisements Feeder 250, Internet Feedback 260, Advertisement Killer 275, Scheduler 265, Internet Feeder 270, TCP/IP Polite Agent 280, TCP/IP Protocol Stack 290, and PPP-TCP/IP Over Modem Protocol 295.

The User Interface Setup Process 201 allows the user to configure the behavior of the system on their desktops. The Advertising Display Manager 210 is responsible for selecting and displaying Advertisements 50 from the User Preference and Advertisements Database 230. The Advertisements Feeder 250 adds new Advertisements 50 to the User Preference and Advertisements Database 230, while the Advertisement Killer 275 purges old Advertisements 50. The Scheduler 265 schedules the display of time dependent Advertisements 50, such as background wallpaper, sound only advertisements, and cursor advertisements. The Feed-

back Manager 240 sends user preferences, user statistics and feedback information to the Advertising System Server 600. The TCP/IP Protocol Stack 290 and PPP-TCP/IP Over Modem Protocol Subsystem 295 handle the lower level details of transferring information to the Network 700. The TCP/IP Polite Agent 280 is responsible for monitoring the communications line utilization rate and transmitting data during times of low communications line utilization.

b. Platform Implemented Functions

The Screen Saver Subsystem 220 tracks user interaction with the system. When the Screen Saver Subsystem 220 detects that the system has been idle, for example, when there has been no user interaction with the computer (through the use of the keyboard, mouse, pointing device or other user input device), for a preconfigured time, it activates the Advertisements Display Manager 210 which will select an advertisement and display it.

In prior systems, screen savers are graphically oriented displays which appear after the computer has been idle for a predetermined period of time. This change of display is primarily designed to prevent screen 'bum in' on the monitor. Screen savers of the prior art are limited to one or more predefined content themes. By utilizing on-line communications, the screen saver of the present invention provides a wide variety of potential content themes which may be personalized and modified on a timely basis in accordance with user preferences.

The general mechanisms for implementing the Screen Saver Subsystem 220 are known in the art. In the preferred embodiment, the Windows-95 operating system monitors user input and calls a preconfigured module when the user has not entered input for a predetermined period. In the preferred embodiment, this module is the Advertisements Display Manager 210. The Screen Saver Subsystem 220 is further described in Microsoft's *Windows-95 Software Development Kit*, including: WIN 32 Overview and reference manuals (chapter 79—Screen Saver Library), available from Microsoft Corp., which is hereby incorporated by reference. On other platforms which do not provide similar functionality, the Advertisements Display Manager 210 itself must monitor for idle time.

The general mechanisms and protocols for communicating with a network, such as the Internet, or on-line service, are known in the art. See, e.g., Stallings, W., *Data and Computer Communications*, Second Edition, Macmillan Publishing Co., (1988). The preferred embodiment utilizes the TCP/IP protocol (Transport Control Protocol/Internet Protocol) which is also well known in the art. See, e.g., Martin J., *TCP/IP Networking*, PTR Prentice Hall (1994). The disclosure of each of the foregoing is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

The TCP/IP Protocol Stack 290 is a set of programs that mediate between application programs and the physical communication line. The TCP/IP Protocol Stack 290 provides application programs with a standard set of function calls for communicating with other application programs connected to the Network 700. Thus the application programs, here the TCP/IP Polite Agent 280, do not have to account for the nature of the physical communication line or error correction.

The PPP-TCP/IP Over Modem Protocol 295 module provides the ability to use the TCP/IP protocol over a specific type of physical communication line, i.e. a pair of modems connected over a telephone line. In the preferred embodiment, the functions of the TCP/IP Protocol Stack 290

and PPP-TCP/IP Over Modem Protocol **295** are implemented within the Windows-95 platform and are accessed from the TCP/IP Polite Agent **280** via system calls. See Microsoft Windows Socket Specifications (rev. 1.1), the disclosure of which is hereby incorporated by reference. Methods of implementing these functions on other platforms are known to those of ordinary skill in the art.

c. User Preference and Advertisement Database

The User Preference and Advertisement Database **230** contains various information needed by the system. The primary data stored is the advertisement information (including executable code modules, bitmaps, video clips and sound clips). The database also stores display statistics, configuration information and user preference data.

Typically, the User Preference and Advertisement Database **230** is located on the Mass Storage Device **516**, however, in LAN installations, the User Preference and Advertisement Database **230** may be stored on a LAN server. This optimizes storage for the system, since an Advertisement **50** needs to be loaded on the LAN only once and is available for display by each workstation on the LAN.

The User Preference and Advertisement Database **230** is preferably accessed through a well-defined Application Programmer's Interface (API), as is known in the art. In the preferred embodiment, this may be an OLE2 compound file or other database means supplied by third-party software vendors. Thus, the location of the User Preference and Advertisement Database **230** will be transparent to the other processes.

d. User Interface Setup

The User Interface Setup Process **201** allows the user to configure the behavior of the system. This process allows the user to input and view preferences as to advertising categories, as well as local computer configuration data.

Local configuration data typically includes:

- a) The Advertising System Server **600** Internet name or Internet address.
- b) The amount of disk space on the Local Computer **200** or LAN which may be allocated to the advertising system.
- c) The length of time an advertisement should be stored before it is deleted from the Local Computer **200**.
- d) The overhead which may be caused by the advertisement transfer process, including the communications line threshold.
- e) Screen saver delay time.
- f) Whether feedback information may be sent to the Network **700**. User preference information typically includes:
 - a) Listings of advertisement categories which are to be given high priority, and those categories which are to be banned from being downloaded or displayed. Typical advertisement categories are "SCUBA diving equipment," "Fast food vendors," "Toys for ages 8-14," or "Cigarettes" and the like. The actual list of categories will be provided by the Advertising System Server **600**.
 - b) Time periods during which sound-only advertisement are to be played.
 - c) Whether wallpaper or cursor advertisements are allowed.
 - d) Whether animation is allowed.

- e) Time periods and types of foreground activities during which advertisements and feedback information may be transmitted.
- f) Identification of the user's natural language.

Additionally the User Preference and Advertisement Database **230** stores information on the Local Computer's **200** platform capabilities, such as sound boards, server resolution and multi-media capabilities.

The Advertisement Feeder **250** will incorporate the advertising preference information, the Local Computer's platform capabilities, disk space limitations, and other configuration data into its request for new advertisements. The Advertising System Server **600** uses this information in selecting the next advertisement to be transmitted. The user preference and configuration data may alternatively be stored on the Advertising System Server **600**.

The Job Manager **720** on the Network Server **600** matches the user preferences and configuration data against the category information for the available Advertisements **50**. Advertisements **50** matching the user's high priority categories and platform capabilities are selected for downloading. Lower priority matches are also selected occasionally on a pseudo-random basis, such that Advertisements **50** within the user's high priority categories will have higher probabilities of being downloaded. The system also allows for increasing the probability of downloading particular advertisements **50**. This allows advertisers to pay increased rates to distribute advertisements faster and to a wider range of users.

User's preferably enter their preferences using whatever interfaces are most natural for the underlying platform. In the preferred embodiment, the user enters information through standard Windows-92 dialog boxes.

In one embodiment of the invention, the Advertising Display Manager **210** allows users to respond to Advertisements **50** being presented by selecting a user grading box which allows users to judge the Advertisements **50** on a scale from "do not show me this advertisement again" to "excellent." This information may be used to modify the user preferences, and may also be incorporated into user feedback information and sent to the Advertising System Server **600** for use by the advertisers.

The User Interface Setup Process **201** also allows users to browse through Advertisements **50** stored on their local system (hard disk or local LAN network), as well as those Advertisements **50** available from the Network **700**. For Advertisements **50** stored locally, whether downloaded from the Network **700** to the User Preference and Advertisement Database **230** or available on a Local Advertisement Database **550** stored on floppy disk, CD-ROM or like device, the User Interface Setup Process **201** displays a menu with the available Advertisements **50**. Each Advertisement **50** stored on the Network **700** or Local Advertisement Database **550** may optionally include a preview segment. The user may have the system present the preview of the Advertisement **50** such as a still image, a short animation sequence, or a sound clip. The user may then select the particular advertisement to be displayed. For Advertisements stored on the Network **700**, the Advertising System Server **600** will transmit a list of available Advertisements **50**. Users may also disable specific advertisements, or all advertisements of a specific advertiser from being transmitted to the Local Computer **500** or displayed.

e. Advertisement Display Manager

The Advertisement Display Manager **210** selects and displays Advertisements **50** from the User Preference and Advertisements Database **230**. The Advertisement Display Manager **210** is typically activated by the Screen Saver Subsystem **220** when the user has not entered input for a predetermined time, or from the Scheduler **265**.

Alternatively, the user may directly access the Advertisement Display Manager 210 from the platform, such as through selecting an icon or other common method.

The Advertisement Display Manager 210 will display the collection of bitmaps, animation, and sound clips associated with the Advertisement 50. FIG. 8 shows a flowchart of a preferred method of an Advertisement Display Manager 210 in accordance with the principles of the present invention. The Advertisement Display Manager 210 is typically called by either the Screen Saver Subsystem 220 to display a screen saver type advertisement after the system has been idle for a predefined period or by the Scheduler 265 to modify the background wallpaper or present a sound-only type advertisement on a periodic basis. The Advertisement Display Manager 210 selects and presents the next Advertisement 50 of the specified type to be presented from the User Preference and Advertisements Database 230.

In the preferred embodiment, the display and other presentation capabilities for each advertisement are self-contained within the Advertisement 50 itself. In this manner the Advertisement Display Manager 210 can support a virtually unlimited number of presentation techniques. The code needed for presenting the advertisement such as a digital sound or video decoder or animation file player, is a resource available from the Resource List 52 within the Advertisement structure 50. The resource may exist in a number of forms such as executable or interpreted code or scripting code such as that used in Hot Java, available from Starwave Corp. When the resource consists of interpreted or scripted code, the interpreter itself becomes an additional resource which must be made available to the Local Computer. If necessary, the Advertisement Feeder 250 will download this resource to the Local Computer 500, using the same techniques as used to download other advertisement data.

Many platforms, including the preferred Windows-95 platform, include a multi-media subsystem that provides APIs for playing animation, sound clips, video clips, etc. See Win32 Programmer's Reference Manual, hereby incorporated by reference. Alternatively, there are a wide variety of stand-alone tools suitable for providing such functions on Windows, Macintosh and other platforms.

In the preferred embodiment, each Advertisement 50 will include a small DLL with an entry point with a pre-defined name. This entry point will be called by the Advertisement Display Manager 210 in order to display the Advertisement 50. The advertisement entry point is specific for each Advertisement 50. When the advertisement entry point is called, the particular code needed to present the given Advertisement 50 will be executed.

User interaction with the Advertisement Display Manager 210 is preferably initiated by pressing a predesignated key, for example F10. When the Advertisement Display Manager 210 is active, all user input is routed directly to the Advertisement Display Manager 210, thus allowing for user interaction with Advertisements 50. The Advertisement Display Manager 210 selectively forwards certain keys to the default operating system routine, which will typically terminate the Advertisement Display Manager 210. The user may interact with the Advertisement Display Manager 210 through a number of ways, including answering questioners, initiating a WEB browser to connect directly to an advertiser WEB page on the Network 700, or automatically initiating a voice connection through the Modem 520 to the advertiser.

Additional aspects of the present invention utilize a variety of techniques for presenting the Advertisements 50. These techniques include displaying advertising as the back-

ground "wallpaper" of the display or modifying the cursor to include an advertiser's logo or other symbol. Additionally, small advertising logos or other advertising content may be placed on the Display Device 513 either at a fixed location on the Display Device 513 or fixed relative to user display windows such that when the user display window is moved on the display the advertisement will move with the window. In the preferred Windows-95 environment, these functions are performed through system calls as described in the Win32 Programmer's Reference Manual, available from Microsoft.

An additional presentation technique is the use of sound-only advertising. The Advertisement Display Manager 210 will make use of a Sound Device 530 on the Local Computer 200. Any sound devices supported by the platform are suitable. In the preferred embodiment, this includes the Sound Blaster card, available from Creative Labs.

f. Scheduler

The Scheduler 265 keeps track of the list of timing-dependent operations. When the time comes to execute a timing-dependent Advertisement 50, as for example changing the wallpaper or playing a sound-only Advertisement 50, the Scheduler 265 notifies the Advertising Display Manager 210, which performs the required action.

g. Advertisements Feeder

The Advertisement Feeder 250, is responsible for adding new Advertisements 50 to the User Preference and Advertisement Database 230. Advertisements 50 preferably are provided from the Internet through the Internet Feeder 270, however, the Advertisements Feeder 250 is not dependent on the type of advertisement source and may receive Advertisements 50 from other sources, such as commercial on-line services, via other feeder mechanisms and other types of polite agents, as shown by references 271 and 272, respectively, in FIG. 4.

FIG. 9 shows a flow chart of a preferred embodiment of an Advertisement Feeder 250 constructed in accordance with the invention.

To download a new Advertisement 50, the Advertisement Feeder 250 first creates a Polite Agent Job 285 to request the Advertising System Server 600 to select the next advertisement for downloading (step 251). The Advertising System Server 600 selects the next Advertisement 50 to be transferred based on the individual user's preferences and configuration and pricing parameters attached to each Advertisement 50. The Advertising System Server 600 sends the Resource List 52, such as executable code modules, bitmaps, animation, sound clips, scripting systems, etc., that the Advertisement 50 needs in order to be presented. The Advertisement Feeder 250 queries the User Preference and Advertisement Database 230 and determines which resources are already available locally, i.e. on the user's PC or LAN. The Advertisement Feeder 250 creates a Polite Agent Job 285 for each resource not in the User Preference and Advertisement Database 230, requesting the Advertising System Server 600 to download only the necessary resources (steps 252, 254). Once the resources have been downloaded, the Advertisement Feeder 250 adds the Advertisement 50 to the User Preference and Advertisement Database 230 (step 253).

An important part of the functionality of the client system is the ability to resume the transfer of an Advertisement 50 which had been only partially transferred during the previous connection., i.e. the client system is preferably able to re-establish transmission of a file after a break in the Communications Link 703. Preferably, the client system will resume transmission from the point in the file at which

communications was broken off. In the preferred embodiment, this functionality is implemented within the TCP/IP Polite Agent **280** and each Polite Agent Job **285**.

h. Advertisement Killer

The Advertisement Killer **275** periodically scans the User Preference and Advertisements Database **230**, and purges Advertisements **50** that satisfy its purge criteria. Typical criterion include the total time the advertisement has been stored and the number of times displayed. Additionally, Advertisements **50** are purged on user demand through user interaction with the Advertisements **50** or the User Interface Setup Process **201**.

i. Feedback Manager

The Feedback Manager **220** is responsible for sending feedback information to the Advertising System Server **600**. This information includes statistics on displayed Advertisements **50**, including user ratings of specific advertisements and the time and length an advertisement was displayed. The Feedback Manager **220** also transmits information which was gathered from the user during interaction with the Advertisements **50**, such as through games and questionnaires. This feedback information may be used as a basis for calculating the advertiser's charge.

j. Polite Agent Technology

The system incorporates a type of intelligent software agent technology referred to herein as a "Polite Agent." The role of the Polite Agent is to perform communication tasks in the background without imposing a noticeable overhead on the user. FIG. 6 illustrates the preferred embodiment of the TCP/IP Polite Agent **280** utilizing the TCP/IP protocol. The TCP/IP Polite Agent **280** transmits information during periods of low line utilization without causing a noticeable slowdown in the data transfer rate of other processes communicating over the Communications Link **703**. The TCP/IP Polite Agent **280** constantly monitors communications status and determines periods of low communication line utilization. It then uses the TCP/IP communications resources, available on the platform, to transfer a portion of the data. Preferably, the agent does not initiate the communication itself, but rather takes advantage of communications resources once the initial Communications Link **703** with the Network Service Provider **701** has been established, thus avoiding additional user charges.

If the communications resource utilization remains low and ample resources are available the software agent performs its designated data transfer task. Alternatively, if communications resource utilization becomes high due to other applications executing on the Local Computer **500** or the Communications Link **703** is disconnected (e.g., the line goes down), the TCP/IP Polite Agent **280** temporarily suspends its data transfer operation until ample resources are available once again. At that point, the TCP/IP Polite Agent **280** recovers the data transfer process from the point where the transfer was suspended, thereby avoiding the need to retransmit data.

Low line utilization occurs when the communications line is busy no more than a predetermined percentage of time. This threshold may be fixed (typically at 30%) user-configurable, or dynamic. When dynamically determined, the threshold may vary with a is number of parameters such as the length of time the TCP/IP Polite Agent **280** has been waiting to transmit, the number or type of Polite Agent Jobs **285** on the Polite Agent Queue **286**, the amount of data which the TCP/IP Polite Agent wishes to transfer, and the type of data being transferred.

Neither the Advertisement Feeder **250** nor the Feedback Manager **240**, directly perform data transfer. Instead, they

place Polite Agent Jobs **285** in the Polite Agent Queue **286** which will be called by the TCP/IP Polite Agent **280** when appropriate. The Polite Agent Jobs **285** perform the actual data transfer.

In the preferred embodiment of the invention, the target operating system will be Microsoft Windows-95 utilizing a TCP/IP protocol. Extension of these operations for different protocols or operating systems will be apparent to those of ordinary skill in the art.

In step A (**41**) of the TCP/IP Polite Agent process **280**, a check is made to see if the Communication Link **703** has been established. This can be done in various ways known to those skilled in the art. A preferred method is to "ping" (send a packet to and receive a response from) the Advertising System Server **600**. See, e.g., J. Martin, *TCP/IP Networking*, PTR Prentice Hall Inc. (1994) (pages 147-48), the disclosure of which is hereby incorporated by reference. An alternative method is to "ping" the Network Service Provider **701**.

In step B (**42**), the line utilization threshold is calculated. As noted above, this calculation may vary in different embodiments of the present invention. Thus, the line **10** utilization may be fixed, user-configurable or dynamic. The threshold calculation also preferably takes into account the load caused by communication generated by the Polite Agent Jobs **285** themselves. This prevents the TCP/IP Polite Agent from not transmitting when the Communications Link **703** is busy primarily due to its own communications.

In step C (**43**), the current communication line utilization is obtained. For TCP/IP under Windows-95, statistical information regarding the communication line utilization is available from the operating system, including such information as bytes/second. In the preferred embodiment, this sampling does not impose a significant overhead on the system and therefore does not cause any noticeable degradation of fore-ground processes.

In step D (**44**), the current communications line utilization is compared to the calculated threshold. If the current utilization is higher than the calculated threshold, the TCP/IP Polite Agent **280** will not perform communication and will return to step A. At this point the TCP/IP Polite Agent **280** may be temporarily suspended by the operating system.

In step E (**45**), the next Polite Agent Job **285** to be executed is selected. Several Polite Agent Jobs **285** can be pending on the Polite Agent Queue **286**. The TCP/IP Polite Agent **280** will alternate between the Polite Agent Jobs **285** preferably on a round-robin schedule allowing all of the Polite Agent Jobs **285** to execute in turn.

In step F (**46**), the Polite Agent Job **285** is executed. The Polite Agent Jobs **285** are designed in such way that they generate a small amount of communication, for example sending I K of information, each time they are executed.

FIG. 7 shows a flow chart for a Polite Agent Job **285** embodying the present invention for transmitting data to the Network **700**. The Polite Agent Job **285** first checks (step A) to see whether this is the first time the job has been called by the TCP/IP Polite Agent **280**. If it is the first time the job has been called, the current file position is initialized **1 0** to the beginning of the file (step B). The file to be transferred is then opened (step C). If the file open is unsuccessful, the Polite Agent Job **285** returns an ERROR flag to the calling TCP/IP Polite Agent **280** which then terminates the Polite Agent Job **285**, removes it from the Polite Agent Queue **286** and marks the job as terminated with error (steps D, E). If the file opens without error, the Polite Agent Job **285** seeks to the current file position in the transferring file and reads the next block of data (steps F, G, H). If no data is left to be

read (end of file condition), the Polite Agent Job 285 closes the file and returns a DONE flag to the TCP/IP Polite Agent 280 which terminates the job and marks the Polite Agent Job 285 as completed successfully (steps I, J). If data was successfully read from the transferring file, the Polite Agent Job 285 transmits the name of the file, file position and file block contents to the Network 700 via the TCP/IP Protocol Stack 290 (steps I, K). The Polite Agent Job 285 then updates the current file position and stores it on persistent storage, such as the Local Computer's Mass Storage Device 516 (step L). Finally, the Polite Agent Job 285 closes the transfer file and returns a flag asking the TCP/IP Polite Agent 280 to again schedule and execute this job again.

It will be obvious to one of ordinary skill in the art how to modify the above type of Polite Agent Job 285 for other types of tasks, such as receiving files. The TCP/IP Polite Agent 280 is capable of transferring any type of information file including, executable code, digitized audio or video, and text, for a variety of types of content such as advertisements, news, or weather, etc. The information file transmitted by the Polite Agent Job 285 may consist of a true file type supported by the platform, or, alternatively, any block of data such as a database record.

In one variation of the present invention, the Polite Agent Job 285 receives as an input the current communication line utilization and a line utilization threshold value. The Polite Agent Job 285 uses this threshold to calibrate its operation by calculating how many network packets, bytes, or other units of data may be transferred without increasing the load beyond the line utilization threshold value.

One of the most common types of Polite Agent Jobs 285, is a file-transfer task. Each time the file-transfer task is run it will send (or receive) a small portion of the file, typically between 0.5-K.

Part of the information collected by the present invention may be sensitive in nature, for example, a user's responses to advertising contests may need to be authenticated to prevent fraudulent responses. The system will use known methods such as public key encryption and digital signatures in order to authenticate the information sent by the Local Computer 500. These methods are well known to those in the art.

The technology of the Polite Agent is general and may be applied in other systems and to the transmission of other types of information content such as news and weather, sports scores and stock quotes, software support information and executable updates, and airlines reservations information.

4. Locally Stored Advertisement Database

FIG. 10 shows a variation of the present invention in which a plurality of Advertisements 50 are stored locally to the Local Computer 500 in a Local Advertisement Database 550 on the Local Storage Device 540 or LAN. In this embodiment, the Advertisements 50 are not downloaded from the Advertising System Server 600, but rather are selected from the Local Advertisement Database 550. The Advertisements Feeder 250 selects Advertisements 50 for presentation on the Local Computer 500 from the Local Advertisement Database 550. In this embodiment, the Advertisements Feeder 250 includes the selection functionality described for the Network Job 725 above. Selected Advertisements 50 are loaded into the User Preference and Advertisement Database 230 for display as described above.

It is understood that various other modifications will be apparent to and can be readily made by those skilled in the art without departing from the scope and spirit of the present invention. Accordingly, it is not intended that the scope of

the claims be limited to the description or illustrations set forth herein, but rather that the claims be construed as encompassing all features of patentable novelty that reside in the present invention, including all features that would be treated as equivalents by those skilled in the art.

What is claimed is:

1. A method for transmitting an information file to a local computer, the information file is stored on a server system, the local computer is coupled to the server system by a network connection, the network connection facilitating a plurality of logical communication links, the method comprising:

- (i) selecting an information file for transmission to the local computer;
- (ii) monitoring a current communication line utilization rate for a primary logical communications link over the network connection;
- (iii) determining whether to transmit data in a current iteration based on the monitored communication line utilization rate;
- (iv) if the determining of step (iii) indicated data may be transmitted, transmitting data of the information file from the server system by employing a secondary logical communications link over the network connection;
- (v) tracking remaining untransmitted data of the information file, the tracking providing tracking information for any remaining untransmitted data of the information file; and
- (vi) repeating steps (ii) to (v) until all data of the information file has been transmitted to the local computer.

2. The method of claim 1, further comprising prioritizing information files for down-loading to the local computer.

3. The method of claim 1, wherein the information file is a software update.

4. The method of claim 1, wherein the primary communication link is a TCP/IP process and the secondary communications link is a TCP/IP process.

5. A method for updating software on a local computer when updated software is stored on a server system, the local computer is coupled to the server system by a network connection, the network connection facilitating a plurality of logical communication links, the method comprising:

- (i) providing an information file adapted to facilitate updating the software;
- (ii) monitoring a current communication line utilization rate for a primary logical communications link;
- (iii) determining whether to transmit data in a current iteration based on the monitored communication line utilization rate;
- (iv) if the determining of step (iii) indicated data may be transmitted, transmitting data of the information file from the server system by employing a secondary logical communications link over the network connection;
- (v) tracking remaining untransmitted data of the information file, the tracking providing tracking information for any remaining untransmitted data of the information file; and
- (vi) repeating steps (ii) to (v) until all data of the information file has been transmitted to the local computer.

6. The method of claim 5, further comprising identifying a version identifier for the software on the local computer.

EXHIBIT D



US006374289B2

(12) **United States Patent**
Delaney et al.

(10) **Patent No.:** **US 6,374,289 B2**
(45) **Date of Patent:** **Apr. 16, 2002**

(54) **DISTRIBUTED CLIENT-BASED DATA CACHING SYSTEM**

(75) Inventors: **Hubert Delaney**, Wilton, CT (US); **Adi Ruppin**, Ramat Gan (IL); **Lior Hass**, Tel Aviv (IL); **Ofer Faigon**, Jerusalem (IL)

(73) Assignee: **Backweb Technologies, Ltd.**, Ramat Gan (IL)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/817,953**

(22) Filed: **Mar. 26, 2001**

Related U.S. Application Data

(63) Continuation of application No. 09/166,686, filed on Oct. 5, 1998, now abandoned.

(51) **Int. Cl.**⁷ **G06F 13/00**

(52) **U.S. Cl.** **709/203; 709/238**

(58) **Field of Search** **709/200–253**

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,151,989 A 9/1992 Johnson et al. 707/10

5,261,069 A	11/1993	Wilkinson et al.	711/145
5,689,708 A	11/1997	Regnier et al.	709/229
5,729,689 A	3/1998	Allard et al.	709/228
5,835,720 A	11/1998	Nelson et al.	709/224
5,864,854 A	1/1999	Boyle	707/10
5,884,046 A	3/1999	Antonov	709/238
6,002,852 A	12/1999	Birdwell et al.	395/200.363
6,003,087 A	12/1999	Housel, III et al.	709/229
6,018,766 A	1/2000	Samuel et al.	709/218
6,021,426 A	2/2000	Douglis et al.	709/200

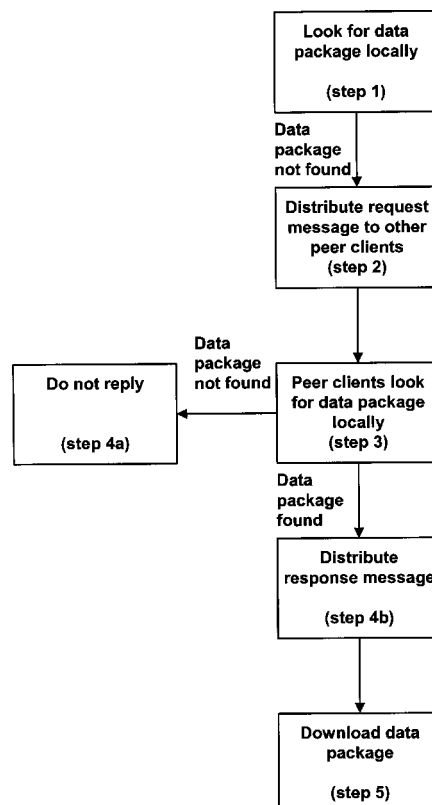
Primary Examiner—David Wiley

(74) *Attorney, Agent, or Firm*—Fliesler, Dubb, Meyer & Lovejoy LLP

(57) **ABSTRACT**

A system and method for enabling data package distribution to be performed by a plurality of peer clients connected to each other through a network, such as a LAN (local area network). Each peer client can obtain data packages from each other or from an external server. However, each peer client preferably obtains data packages from other peer clients, rather than obtaining data packages from the external server.

23 Claims, 9 Drawing Sheets



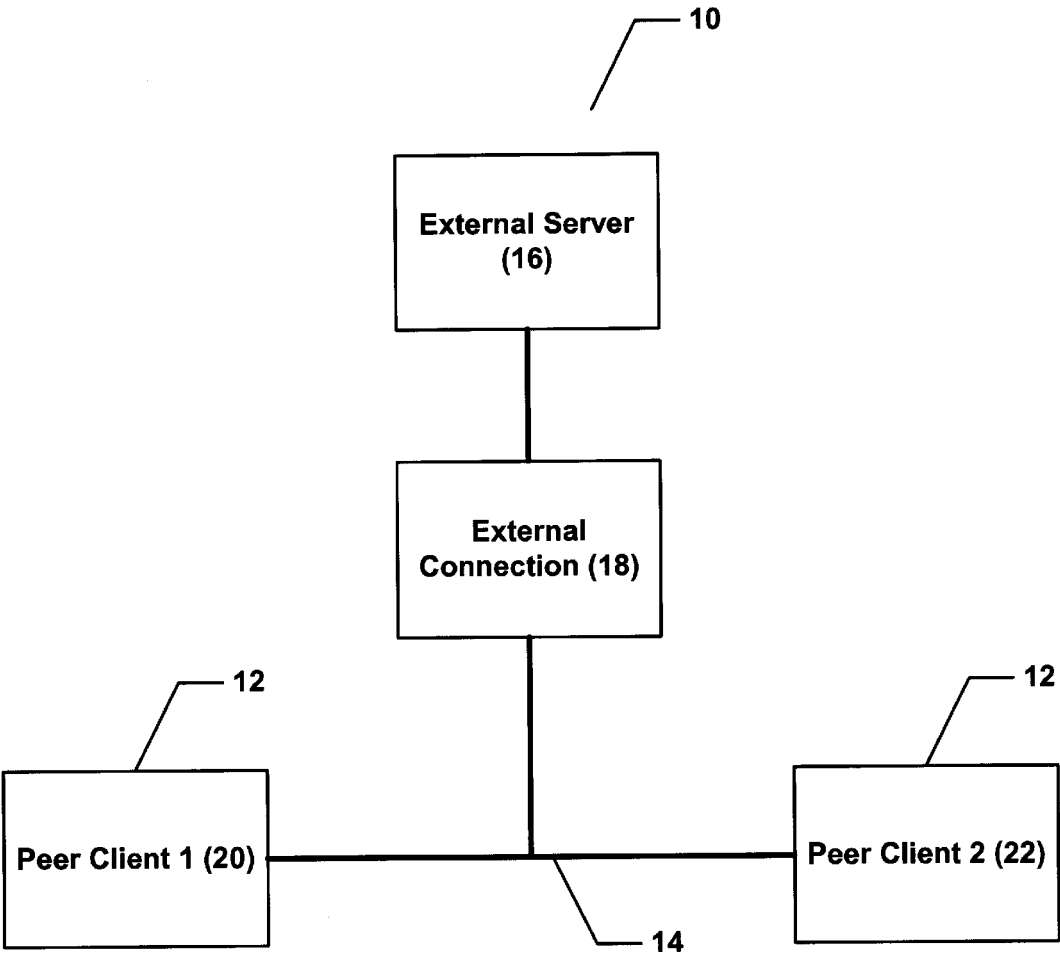
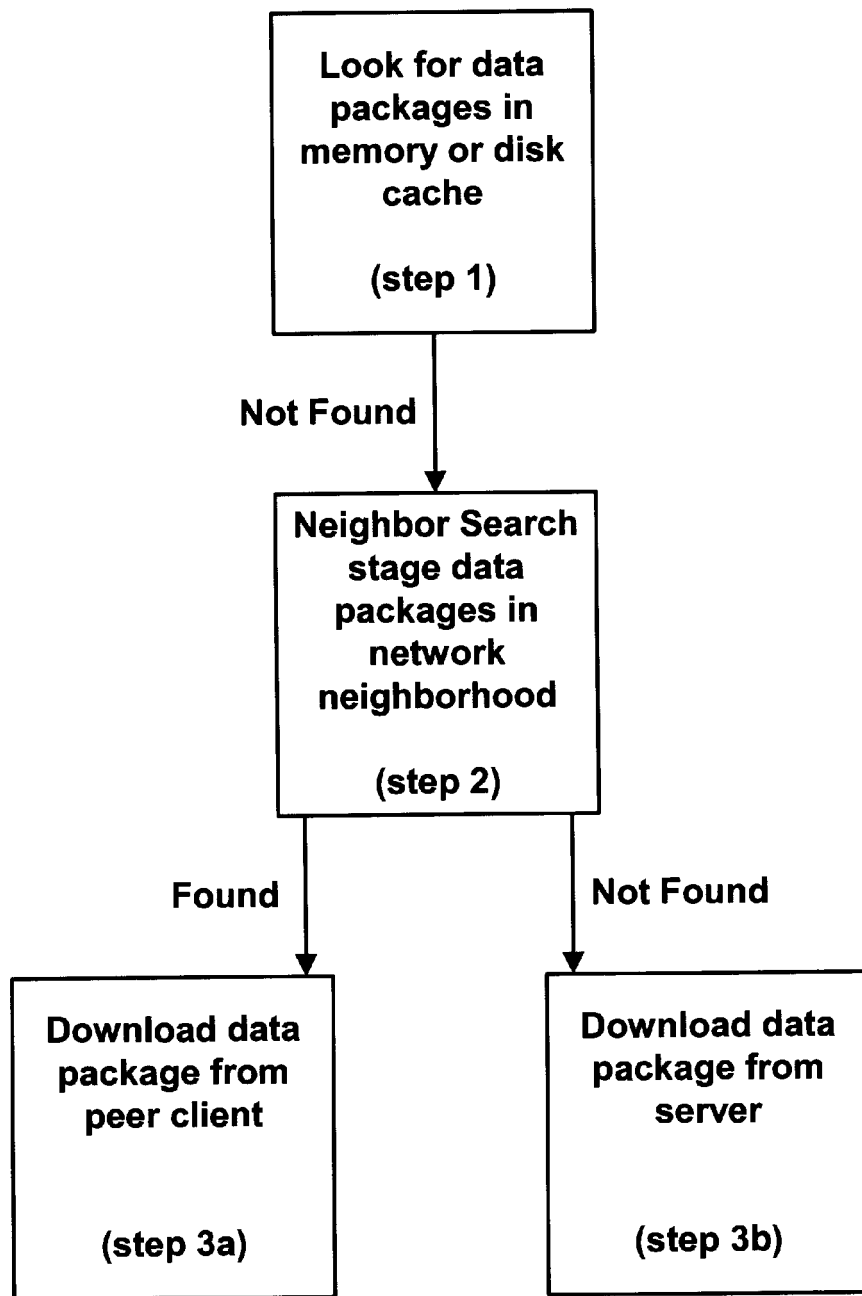


FIGURE 1A

**FIGURE 1B: Downloading Process**

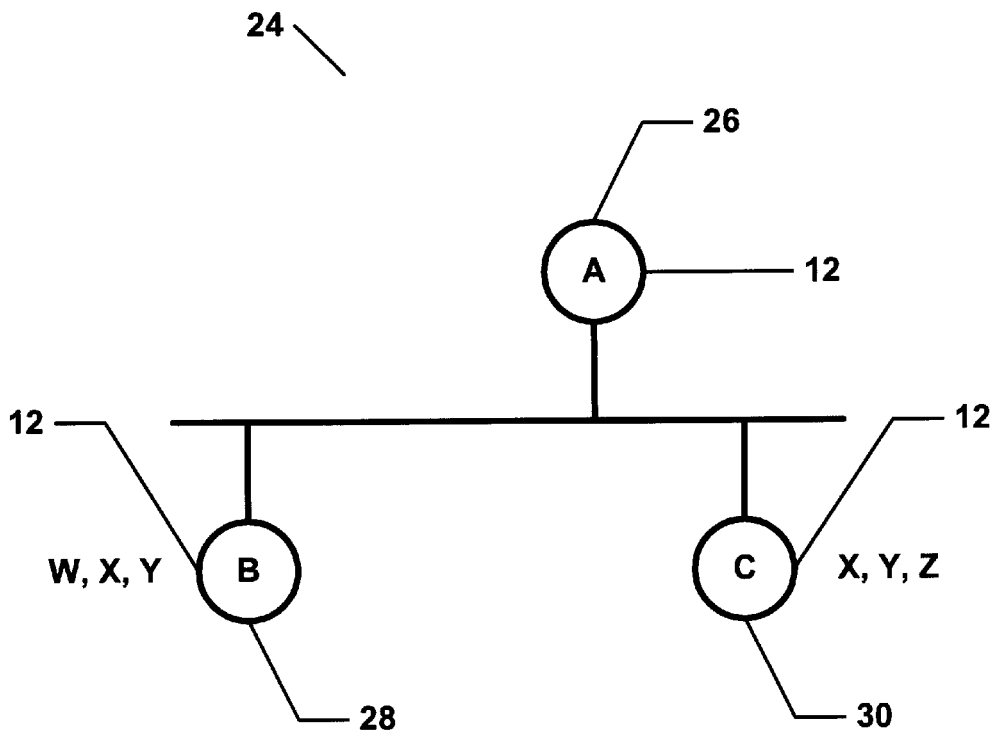


FIGURE 2A: Request-response example

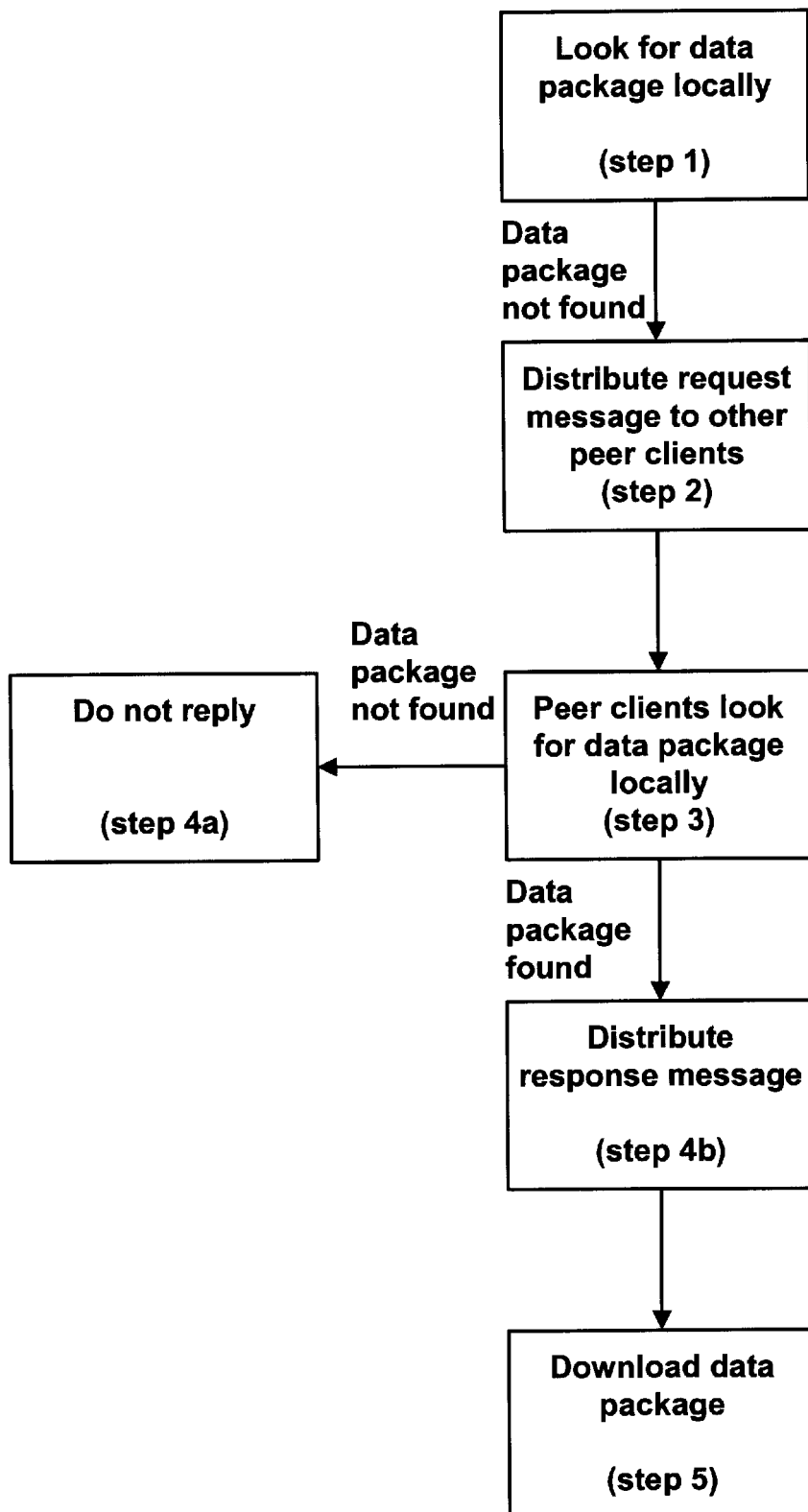


FIGURE 2B



FIGURE 2C: Request message



FIGURE 2D: Response message

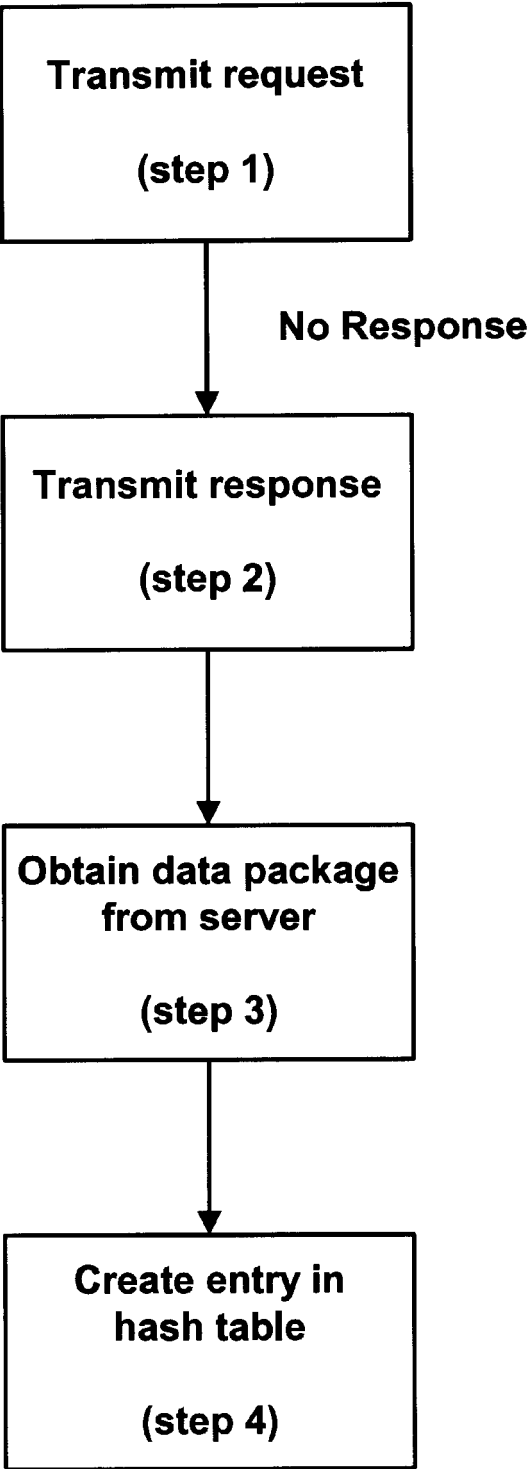


FIGURE 2E

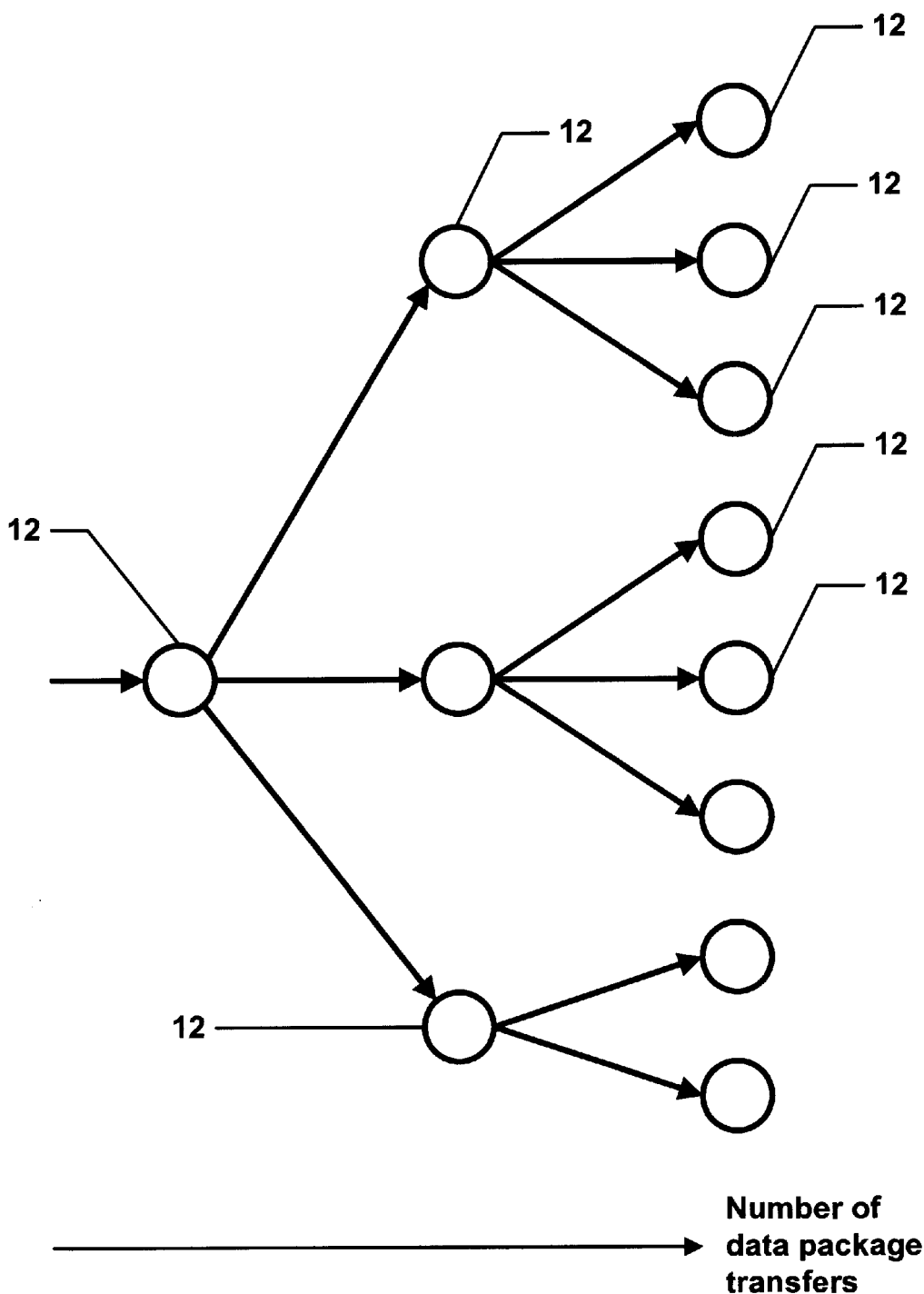


FIGURE 3: Tree-shaped data-flow (optimal)

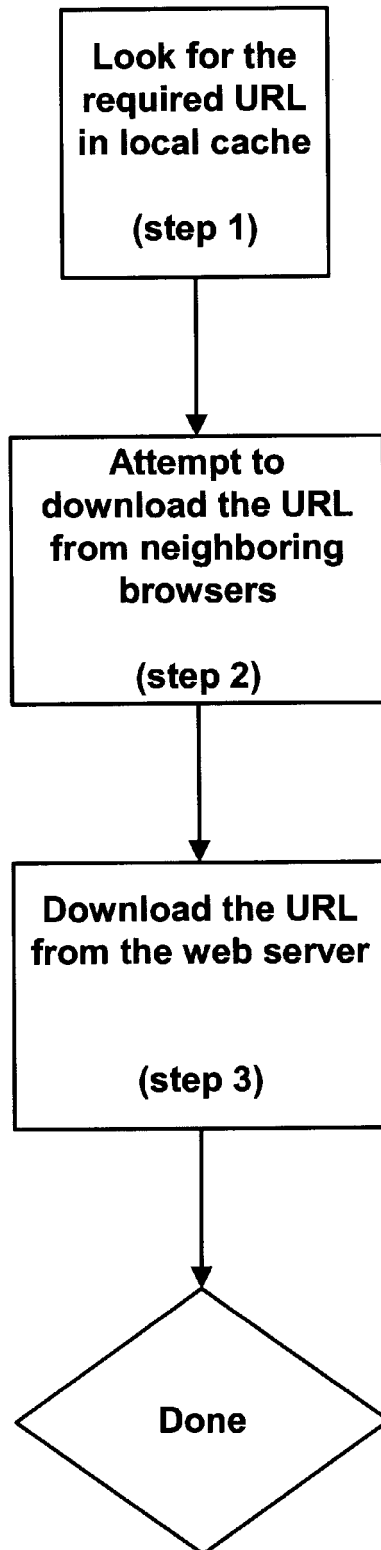


FIGURE 4

PVN	REQ	URL ₁ , URL ₂ , URL ₃ ,...
-----	-----	---

FIGURE 5A

PVN	RSP	URL ₁ , URL ₂ , URL ₃ ,...
-----	-----	---

FIGURE 5B

DISTRIBUTED CLIENT-BASED DATA CACHING SYSTEM

This application is a continuation of Ser. No. 09/166,686, filed Oct. 5, 1998, now abandoned.

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to a distributed client-based data caching system. Specifically, the system of the present invention enables data packages to be served to a client through a flexible, non-deterministic distributed system of peer clients which cache the data packages, in order to maximize efficiency and speed for serving the data package to the client.

Networks which connect two or more computers, such as the Internet or intranets, enable client computers to obtain data packages, such as documents, images, messages, data packages or other types of data, from remote storage media which are not installed on the client computer itself. Instead, these remote storage media are managed and operated through a remote computer, known as a server computer or simply as a "server" (in the same vein, the client computer is also often termed only a "client"). The advantage of such a system is that the client computer can potentially obtain data from any server on the network. The disadvantage of the system is the requirement for sufficient bandwidth on the network to enable data to be transmitted from the server to the client. Furthermore, if the load is not evenly distributed between servers on the network, one server may become overwhelmed with requests, thereby decreasing the speed and efficiency of retrieval. Thus, currently many networks cannot provide rapid and efficient data retrieval due to the heavy demands placed upon the available bandwidth.

Proxy servers are often installed to conserve bandwidth on an Internet connection or on connections to other LANs (local area networks). These proxy servers cache frequently accessed data, thereby reducing the load on the main server, and distributing demand for bandwidth more evenly across the network. Unfortunately, such proxy servers are typically expensive to maintain. Furthermore, proxy servers require dedicated computers to be installed and configured. Each computer on the LAN has to be separately configured in order to communicate with the proxy server. Such configuration is deterministic, such that each client must be configured to communicate with each proxy server separately. Thus, proxy servers have many drawbacks.

A more useful solution would enable Intranets to reap the benefits of the proxy server, without requiring dedicated machines and without requiring any special installation or configuration. Furthermore, such a solution would not be deterministic, such that each client could communicate with more than one server according to the load on each server, rather than according to the configuration of the client itself. Unfortunately, such a solution is not currently available.

Therefore, there is an unmet need for, and it would be highly useful to have, a distributed client-based data caching system which enables data to be stored and retrieved from a plurality of peer clients, or "caching entities", yet which does not require any special configuration or installation of separate servers.

SUMMARY OF THE INVENTION

The present invention is of a distributed client-based data caching system, which enables data to be served to a client through a flexible, non-deterministic distributed system of

caching entities, in order to maximize efficiency and speed for serving the document to the client. The caching entities are peer clients which serve the data to each other, thereby reducing the amount of bandwidth required to obtain data from an external server.

According to the present invention, there is provided a method for distributing data packages across a network, the network featuring an external server for serving at least one data package, the external server being a dedicated server, the steps of the method being performed by a data processor, the method comprising the steps of: (a) providing a plurality of peer clients attached to the network and a list of data packages being stored by each of the plurality of peer clients, each data package on the list of data packages having an entry, the entry indicating a unique identifier for the data package and a location of the data package in at least one of the plurality of peer clients; (b) examining the list of data packages by a first peer client to find an entry for a data package; and (c) if the entry for the data package is present on the list of data packages of the first peer client, retrieving the data package from the location at another of the plurality of peer clients according to the entry for the data package.

Alternatively, the list of data packages is stored on the external server.

According to preferred embodiments of the present invention, the list of data packages is stored on at least the first peer client. Preferably, if alternatively the entry for the data package is absent from the list of data packages of the first peer client, the method further comprises the steps of: (d) sending a request message for the data package by the first peer client to at least one other peer client; and (e) if a response message is received by the first peer client from the at least one other peer client, retrieving the data package from the at least one other peer client by the first peer client.

Preferably, the request message and the response message are transmitted to the plurality of peer clients by broadcasting. Alternatively, the request message and the response message are transmitted to the plurality of peer clients by multicasting. Also alternatively, the request message and the response message are transmitted to the plurality of peer clients by polling each peer client individually.

Also alternatively and preferably, if the response message is not received from the at least one other peer client by the first peer client, the method further comprises the step of: (f) obtaining the data package by the first peer client from the external server. Preferably, the method further comprises the step of sending a response message by the first peer client to the at least one other peer client substantially before the first peer client obtains the data package from the external server. More preferably, the list of data packages is stored on each of the plurality of peer clients, and the method further comprises the steps of: (g) receiving the response message from the first peer client by the at least one other peer client; and (h) altering the list of data packages being stored by the at least one other peer client for indicating the location of the data package according to the response message.

Alternatively, the list of data packages is stored on each of the plurality of peer clients, and the method further comprises the steps of: (g) receiving the response message from the first peer client by the at least one other peer client; and (h) altering the list of data packages being stored by the at least one other peer client for indicating the location of the data package according to a probabilistic function.

Preferably, the probabilistic function is performed according to a set of equations:

$$\text{New location} = \begin{cases} \text{Old location} & Po(x) = 1 / (\text{generation} + 1) \\ \text{New location} & Pn(x) = 1 - 1 / (\text{generation} + 1) \end{cases}$$

wherein Pn(x) is a probability that the new location is substituted for the old location, Po(x) is a probability that the old location is retained, and "generation" indicates how many times the location had been previously changed.

Also preferably, an upper limit is predetermined for a number of the plurality of peer clients served substantially simultaneously by the at least one other peer client, such that if a number of the plurality of peer clients served substantially simultaneously by the at least one other peer client is greater than the upper limit, the method further comprises the step of: (d) sending a busy message from the at least one other peer client to the first peer client.

Preferably, the external server is a Web server, and the plurality of peer clients is a plurality of Web browsers.

Also preferably, the external server is a BackWeb™ server, and the plurality of peer clients is a plurality of BackWeb™ clients.

Preferably, the unique identifier for the data package is an MD5 digest of the data package.

According to still other preferred embodiments of the present invention, the step of retrieving the data package is performed according to a protocol based on TCP/IP. Preferably, the protocol is HTTP. Alternatively and preferably, the protocol is FTP.

Hereinafter, the term "protocol based on TCP/IP" includes any such protocol, including but not limited to the HTTP (hypertext transfer protocol) and FTP (file transfer protocol) protocols.

Hereinafter, the term "data package" refers to any discrete, identifiable unit of data, including but not limited to documents, images, messages, data packages or any other type of data.

Hereinafter, the term "computing platform" refers to a particular computer hardware system or to a particular software operating system. Examples of such hardware systems include, but are not limited to, personal computers (PC), Apple Macintosh™ computers, mainframes, minicomputers and workstations, which are also non-limiting examples of data processors for operating a software application under an operating system. Examples of such software operating systems include, but are not limited to, UNIX, VMS, Linux, MacOS™, DOS, one of the Windows™ operating systems by Microsoft Inc. (Seattle, Wash., USA), including Windows NT™, Windows 3.x™ (in which "x" is a version number, such as "Windows 3.1™"), Windows95™ and Windows98™.

For the present invention, a software application could be written in a substantially suitable programming language, which could easily be selected by one of ordinary skill in the art. The programming language chosen should be compatible with the operating system according to which the software application is executed. Examples of suitable programming languages include, but are not limited to, C, C++ and Java.

Hereinafter, the term "broadcast" may also include "multicast" as well.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

FIGS. 1A and 1B are schematic block diagrams of an exemplary basic system and method according to the present invention;

FIGS. 2A–2E are schematic block diagrams of an exemplary request/response protocol and method according to the present invention;

FIG. 3 is a schematic block diagram of an exemplary preferred data-flow diagram according to the present invention;

FIG. 4 is a flowchart of a method for operating the system of the present invention with Web browsers; and

FIGS. 5A and 5B are exemplary request and response messages according to the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention is of a distributed client-based data caching system, which enables data to be served to a client through a flexible, non-deterministic distributed system of caching entities, in order to maximize efficiency and speed for serving the data to the client. The caching entities are peer clients which serve the data to each other, thereby reducing the amount of bandwidth required to obtain data from an external server.

The system of the present invention enables clients to share data packages among themselves across their local network neighborhood, for example within a LAN, thereby eliminating the need for a specialized proxy server. Furthermore, the network traffic is not significantly affected, since modern network architectures are well suited for peer-to-peer communications. Most currently operating networks have a star topology, using switching hubs, in which communication between two peers does not affect simultaneous communication among other nodes on the network. Thus, the system of the present invention overcomes the drawbacks of a proxy server, yet does not add significant loads to the traffic on the network itself.

For currently available client-server software applications known in the art, whenever a client requires a data package, the following algorithm is performed. First, the software application attempts to locate the data package locally on the memory or on the disk or disks of the client. Then, if the data package is not found locally, the software application retrieves the data package from the appropriate server.

By contrast, the operation of the system of the present invention adds an intermediate step. For the present invention, if the data package is not found locally, an attempt is made to retrieve the data package from a peer client on the local network "neighborhood" before resorting to retrieving the data package from the server.

Thus, for the system of the present invention, every client actually functions as a caching proxy. Once a client requires a data package, it queries all the hosts, which are actually peer clients, on the local network for that data package. If no neighboring peer client has the data package, the client retrieves the data package from the external server as usual. However, if a neighboring client already has the required data package, the requesting client will download this data package from the peer client rather than from the external server.

The principles and operation of the distributed client-based data caching system according to the present invention may be better understood with reference to the drawings and the accompanying description.

FIG. 1A is a schematic block diagram of an exemplary system according to the present invention, while FIG. 1B is

a flowchart of the operation of the system of FIG. 1A. FIG. 1A shows a system 10 which includes a plurality of peer clients 12 connected by a local network 14 of some type, for example a LAN, indicated by the heavier line in FIG. 1A. Two peer clients 12, labeled as "peer client 1" 20 and "peer client 2" 22, are shown for the purposes of illustration only and without intending to be limiting in any way. Each peer client 12 is also connected to an external server 16 of some type by an external connection 18. Although only one external server 16 is shown, a plurality of external servers could also be implemented. External server 16 is a dedicated server, in the sense that this server has a primary or at least a substantially significant role as a server for data packages. As shown for the purposes of illustration, external connection 18 only connects to local network 14 at one point, although multiple such external connections could also be implemented (not shown). In addition, external connection 18 could also optionally connect each peer client 12 directly to server 16 (not shown).

The operation of system 10 according to the present invention is illustrated with reference also to FIG. 1B. In step 1, peer client 12, such as peer client 12 looks for a data package in the local memory or disk cache of that particular peer client 12. If the desired data package is not found on the local disk cache, then in step 2, peer client 12 queries any other peer client(s) 12 on local network 14 to determine whether any other peer client 12 has a particular data package. For example, peer client 20 could query peer client 22, to determine whether peer client 22 has the desired data package. In step 3a, if peer client 22 has the desired data package, then peer client 20 obtains the data package from peer client 22. Alternatively, as shown in step 3b, if peer client 22 does not have the desired data package, then peer client 20 obtains the data package from server 16 through external connection 18. Thus, every peer client 12 is also potentially a server which is internal to local network 14, and hence could be described as an "internal server" to distinguish peer client 12 from external server 16.

Each peer client 12 could also be described as a "caching entity" and the data stored by each client for serving to other peer clients 12 as "cached data" or "cached data packages".

A number of different possible embodiments of the system of the present invention can be implemented, of which two illustrative embodiments are shown with reference to the Figures below. Briefly, FIGS. 2A–2D illustrate an exemplary embodiment of the system of the present invention for implementation with the software application of BackWeb™ (BackWeb Technologies Ltd., Ramat Gan, Israel) on a local area network (LAN). FIGS. 4 and 5A–5B illustrate an exemplary embodiment of the system of the present invention for implementation with a Web browser software application on the Internet.

FIG. 2A shows an exemplary local network 24 which features a plurality of peer clients 12 of which three are shown for the purposes of discussion only and without intending to be limiting in any way. For the purposes of discussion only, suppose a peer client 26, labeled "A", wishes to obtain four data packages "W", "X", "Y" and "Z". None of these data packages are local to peer client 26, which must therefore obtain these data packages from either another peer client 12 as an internal server, or from an external server (not shown). Local area network 24 features two other peer clients 12: peer client 28, labeled "B", and peer client 30, labeled "C". Peer client 26 must therefore first communicate a request to peer client 28 and peer client 30 to see if the desired data packages are available at either location, and then peer client 26 must obtain these data

packages from peer client 28 or peer client 30 if the desired data packages are available.

Preferably, two protocols are used for communication between peer clients on a local area network (LAN), a data package-exchange protocol and a control protocol. Specifically, the data package exchange protocol is used to transfer data packages between peer clients, once the desired data package has been located, and is described in greater detail with respect to FIG. 2B below. The control protocol enables each peer client to efficiently build and maintain tables which describe the location of available data packages across the local area network, by exchanging messages.

Each peer client maintains two hash-tables which contain information about data package location: a local-data packages table and a network-data packages table. The local-data packages table is a hash-table of data packages which reside on the storage medium or media of the peer client itself. The network-data packages table is a hash-table of data packages which reside on the storage medium or media of other clients on the local network. This table contains the local area network address of the peer client on which each data package is being stored. The size of this hash-table is preferably limited in order to reduce memory consumption. More preferably, each entry in the table has a time-stamp, such that older entries are purged when the size of the table exceeds the upper permissible limit.

In order to effectively identify the desired data package, preferably each data package has a unique identifier or "fingerprint" associated with it. More preferably, this unique identifier is an MD5 digest of the content of the data package (for a description of the MD5 specification, which is an industry standard and would therefore be obvious to one of ordinary skill in the art, see "RFC 1321" at <http://ds.internic.net/rfc/rfc1321.txt>).

Once any peer client knows both the unique identifier and the location of the data package on the local network, that client can then proceed to download the data package. However, the peer client may not know the location of the desired data package, in which case the client must follow a control protocol according to the present invention in order to determine the location of the desired data package and to enable the client to build these hash tables with respect to future attempts to locate a data package.

The control protocol is used to provide each client with knowledge about the locations of data packages across the local network. In the preferred implementation illustrated with respect to FIGS. 2A–2D, control messages are preferably sent and received as broadcast or multicast packets. Local area networks such as Ethernet networks support broadcast or multicast packets such that all peer clients on a local area network receive the broadcast or multicast packets. Effectively, a single packet can be sent to all peer clients by using broadcast or multicast, thereby reducing the amount of traffic on the network required as a result of transmitting the request message (see for example Chapter 12, "Broadcasting and Multicasting", of *TCP/IP Illustrated Volume*, by W. Richard Stevens, Addison-Wesley, 1994). However, optionally the system of the present invention could poll each peer client individually with a control message for that peer client, although this is not preferred since such individually addressed messages would consume excessive amounts of available bandwidth. In such a situation, preferably polling would be restricted to a certain group of peer clients as internal servers, in order to reduce the amount of traffic on the local area network.

For the preferred implementation in which broadcast or multicast is used, more preferably, the decision to select

either IP multicast or broadcast is made according to the configuration set by the network administrator for the local area network. IP multicast is preferable in terms of load on the clients of the local network, but may not be supported on all platforms (operating systems). More preferably, the TTL or Time to Live may be configured. The TTL specifies the number of routers a packet can cross before being dropped. Configuring the TTL enables data package sharing to be expanded across subnet boundaries.

As shown with respect to FIG. 2B, the control protocol of the present invention preferably operates as follows. In step 1, peer client "A" from FIG. 2A looks for a data package on the local storage medium or media. In step 2, since the data package was not found locally on the medium or media of peer client "A", peer client "A" must download the data package and therefore preferably multicasts (or alternatively broadcasts) a request message. A request message preferably contains a protocol identifying version number (PVN) for the control protocol of the present invention and a list of MD5 digests of the needed data packages, as shown in FIG. 2C.

Optionally and preferably, if more than one data package is desired, a list of requested data packages is included in the request message rather than a single MD5 digest, in order to reduce the total number of request messages on the network.

In step 3, the neighboring clients, shown as peer clients "B" and "C" in FIG. 2A, receive this request message and search for the requested data package in their local-data packages hash-table. A peer client which does not find the data package locally does not reply, as shown in step 4a. Otherwise, in step 4b the peer client sends a response message, preferably after waiting a short random time interval to determine whether another peer client will respond first. More preferably, the peer client does not distribute the response message if another client responded previously, in order to reduce unnecessary traffic on the local area network. Also more preferably, the peer client distributes the response message by broadcast or multicast.

For example, as shown in FIG. 2A, if peer client "A" requests a data package "W", peer client "B" would reply with the response message, since peer client "B" has the data package stored locally. By contrast, peer client "C" would not reply with a response message, since peer client "C" does not have data package "W" stored locally. On the other hand, if peer client "A" requests a data package "X", both peer client "B" and peer client "C" could respond. In this situation, preferably only peer client "B" or peer client "C" would respond, depending on which peer client had the shorter random interval for waiting before sending the response message.

More preferably, responses are sent only for data packages with yet unknown locations. For example, suppose client "A" requests data packages "W", "X", "Y" and "Z". Client "B" has data packages "W", "X" and "Y", and is the first to r, with a reply message indicating possession of data packages "W", "X" and "Y". Suppose another client, "C" has data packages "X", "Y" and "Z". Since it replied after client "B", the response message from client "C" will only indicate possession of data package "Z" because this is the only data package with an as yet unknown location.

A response message optionally contains the identifying PVN, the list of MD5 digests of data packages that were found and a TCP port number, as shown in FIG. 2D. The port number identifies on which TCP port the responding peer client is waiting for data package requests. Alternatively, the response message optionally contains other indicators which

enable the requesting client to retrieve one or more data packages from the responding peer. Preferably, response messages are also be broadcast for data packages which are currently being downloaded from an external server, for reasons described in greater detail below.

In step 5, the peer client downloads the data package or data packages. In principle, according to a relatively simple embodiment of the present invention, at this stage the requesting client either receives a reply and downloads the data packages from the client that replied; or, if a reply is not received within a certain period of time, proceeds to download these data packages from an external server. If the peer client is downloading a data package from another peer client as an internal server, the data package-exchange protocol is used to obtain the data package. The data package-exchange protocol is based on some appropriate peer-to-peer communication protocol, including but not limited to the HTTP protocol (see RFC-2068, "Hypertext Transfer Protocol-HTTP/1.1", available from <http://ds.internic.net/rfc/rfc2068.txt> as of Sep. 23, 1998).

Preferably, a more complex implementation is employed, since such a simple implementation may cause multiple clients to fetch the same data packages from the external server simultaneously. This situation would arise if several peer clients need to download the same data packages at approximately the same time, which is a very probable scenario for push clients for which content delivery is triggered by an external server, since none of these clients would receive a response to its request. Instead, the other clients would still be downloading the data package when the new client request is broadcast, such that none of them would be ready to serve these data packages. Thus, many or even all of the clients would attempt to retrieve the data package from the external server and not from another peer client, thereby increasing the amount of traffic on the network and reducing the efficiency of operation of the system of the present invention.

Preferably, the problem is solved by notifying other clients when a first client is downloading the data package from the external server, even if the process of retrieving the data package is not yet complete. In this preferred embodiment, the first client which requires the data package obtains the data package from the external server. Other clients which require the data package will then download it from the first client, even if the first client is still in the process of retrieving the data package from the external server. The preferred embodiment of the method of the present invention is described in greater detail with regard to FIG. 2E.

In step 1, the requesting client again transmits the request, again preferably by broadcasting or multicasting, and then waits for a response. If no response is received within a certain period of time, in step 2 the client transmits a response message as if replying to its own request, indicating that this client either has the data package, or in this case, that the client is retrieving the data package. In step 3, the client retrieves the data package from the external server.

In step 4, other clients create an entry in their network data packages hash table, indicating the location of the client which will be serving the data package. Thus, preferably only a single client accesses the external server for any given data package.

If a request is sent for multiple data packages, but a response is received indicating the location of only some of the data packages at a neighboring peer client or clients, the client first obtains these data packages from the neighboring

peer client or clients. Next, the client then transmits the response message for the rest of the data packages, and proceeds to obtain the remaining data package or data packages from the external server. Thus, the client only obtains the data package or data packages from the external server which are not available locally, rather than obtaining all of the data packages from the external server, thereby reducing network traffic.

According to preferred embodiments of the present invention, preferably the process of downloading data package from peer clients is optimized to reduce the amount of time required for downloading, the load on each individual client and the overall network traffic. Such optimization is performed as follows.

First, preferably the exit degree of each client is bound, such that each client is only able to serve a fixed, limited number of other clients simultaneously. More preferably, the default limit is three other clients, for example, or some another appropriate number which is preferably configured by the user or by the network administrator. If an additional client attempts to download a data package from a client which is already serving the maximum number of other clients will receive a "busy" message. This feature limits the load on each individual client.

Also preferably, the present invention is able to optimize the selection of the best client from which the data package should be obtained. For example, if client "A" had already downloaded a larger portion of the required data package than client "B", transferring the data package from client "A" is more optimal. Such clients are preferentially selected to serve data packages, since these clients will be able to serve the data package after a shorter time period has elapsed. Such preferential selection occurs by shortening the time period for waiting before these clients respond, thereby increasing the likelihood that they will serve the data packages. For this reason, the client preferably calculates the random delay before responding such that the delay is inversely proportional to the percentage of the data package which has been already downloaded. In addition, the random delay is preferably proportional to the number of clients being served at the moment, in order to decrease the likelihood of overloading already busy clients.

In addition, according to other preferred embodiments of the present invention, preferably the entries of the locations of data packages in the network data packages table are updated according to a probabilistic function. Such a function is preferred in order to prevent all of the clients from registering a single client as the server for any particular data package, for example. When different clients respond, usually at different times, indicating they have a specific data package, the remaining clients listening across the network update the entry for this data package in their network data packages table, by adding the IP address, or some other type of address according to the addressing system employed by the network, of the client which can serve the data package to this table. In a simple implementation, the clients would store only the last advertised location of each data package, and therefore many or all clients might attempt to obtain the data package from a single client as the internal server, thereby overloading that client.

To avoid this situation, preferably the following probabilistic algorithm is used to determine the particular client address which is stored in the network data packages table. Each time a new client transmits a response message, indicating that this client is able to serve particular data package, the probability that the new IP address of the new

client is substituted for the old IP address is calculated according to the following equations:

5 New IP address = $\begin{cases} \text{Old IP address} & Po(x) = 1 / (\text{generation} + 1) \\ \text{New IP address} & Pn(x) = 1 - 1 / (\text{generation} + 1) \end{cases}$

wherein Pn(x) is the probability that a new IP address is substituted for the old IP address, Po(x) is the probability that the old IP address is retained, and "generation" is a number indicating how many times this address had been previously changed.

For example, if client "A" responds indicating it has data package "X", then initially all other peer clients store the IP address of client "A" as the location of data package "X". If client "B" then broadcasts a response also indicating that client "B" has data package "X", then the probability that any one client now changes the IP address for the location of data package "X" is 50%. In other words, about half of the clients should now point to client "A" and about half should point to client "B".

Such a substantially even distribution of load across multiple clients should produce data-flow with a tree-shaped topology, as shown in FIG. 3, rather than a random topology, thus optimizing the average download time and the load on the serving clients.

Furthermore, if any client requests a particular data package during the period required by client "A" for downloading that package, preferably client "A" sends a broadcast or multicast message indicating that the package is in the process of being downloaded. Therefore, preferably only a single client "B" polls client "A" for each data package, for example. Other clients preferably automatically receive any responses from that polling action through the broadcast or multicast transmission, and thus will not be forced to poll for themselves.

The polling (request/response) traffic is optimized since there is usually no need to transmit both a request and a response for each data package needed by each client. Such optimization is possible since each client preferably receives substantially all of the request/response communication of all the other clients and "remembers" the location of the data packages in the network-data packages table.

As previously described, the actual process for receiving a data package from an internal server is performed according to the data package exchange protocol, by using the HTTP protocol or some other suitable peer-to-peer communication protocol. The data package transfer software application of the present invention preferably features a timer, for detection of an aborted transfer or a very slow data package transfer, for example. The timer determines when such a transfer has timed out. If a time-out occurs, the requesting client preferably repeats the whole process. If the transfer remains unsuccessful after a plurality of attempts, the client preferably ceases to attempt to transfer the data package from the peer client as the internal server, and instead transfers the data package or data packages directly from the external server.

Again, as described previously, if a requested data package has not yet finished being downloaded by a peer client, the requesting client receives a message indicating that the data package is not ready, as well as an indication of the fraction of the data package already downloaded. The requesting client continues polling the serving client until the data package download is complete. If the download becomes substantially slower or is otherwise interrupted or terminated for a long period of time, the requesting client behaves as if a time-out occurred.

According to additional preferred features of the present invention, substantially automatic detection of peer clients is supported. Such automatic detection enables each peer client to detect the presence of other peer clients on the network. If such peer clients are not found, preferably the system of the present invention is disabled, since the operation of the system as described above would only prolong the time period required to download a data package if no other peer clients are available.

Preferably, the amount of bandwidth on the local area network which is consumed by each peer client serving data packages to other clients is limited, to avoid over-burdening any specific host. This limit is preferably configurable by the user or by the network administrator.

Furthermore, in order to protect peer clients from unauthorized access of local storage media through the system of the present invention, certain security features are preferably included. For example, preferably only data packages identified in the hash tables are able to be transferred from the client. Thus, transmitted data packages are preferably only data packages which were intended to be served to the peer clients, such that malicious users preferably cannot use the system of the present invention to obtain "random" data packages from the storage media of a peer client. Data packages are more preferably only referenced by their unique identifier, such as their 128-bit MD5 digest, such that a data package is only able to be downloaded from a client if the intended recipient knows this digest. Thus, the name of a data package alone is preferably not sufficient information to permit retrieval of the data package from a peer client.

According to another embodiment of the present invention, the system of the present invention is also applicable to Web browsers, FTP clients, and other software applications involving client-server data-transfer. As described with reference to FIGS. 4 and 5A-5B, another exemplary embodiment of the present invention is used for caching Web content.

In step 1 of FIG. 4, a Web browser being operated by a client computer requests a specific data package. First the Web browser looks at the local cache, as is known to one of ordinary skill in the art. If the data package is found in the local cache, then that data package is retrieved from the local cache. Otherwise, the Web browser issues a message requesting this data package, preferably by using broadcast or multicast message transmission. The data package is preferably uniquely defined by a unique identifier. More preferably, the unique identifier is the URL of the data package, or alternatively and preferably a combination of the URL of the data package and timestamp, or by any other suitable unique identifier.

For optimization, if more than one data package is required, the Web browser preferably transmits one request message containing the list of needed data packages, thereby reducing the total network traffic across the network. Such a situation may arise if, for example, the Web browser had just parsed an HTML (hypertext mark-up language) document, or Web page, which contains many links to follow. Preferably and optionally, each request message contains an identifying "magic number", which may contain the protocol version (PVN). For instance: "V1.0". As shown in FIG. 5A, the request message includes the list of URL's or other unique identifiers to identify the data package or data packages being requested, which is similar in function to the list of MD5 digests described previously for request messages, and a unique identifier identifying the request message, shown as "REQ".

In step 2, other Web browsers across the network listen to detect request messages of this type. These Web browsers, which are peer clients for this embodiment of the present invention, receive this request message and check their own cache for the requested URL. If the requested URL is found in the local cache of a Web browser, that Web browser preferably waits a random interval and then preferably transmits a response message indicating it has the required data package (or data packages). Preferably, the message is broadcast or multicast. More preferably, that Web browser does not reply if another Web browser had replied first. A reply message is preferably sent by a particular Web browser even if the requested URL is still being downloaded by that Web browser.

In step 3, if no response to an issued request message is received within a certain amount of time, for example 5 seconds, then the process is preferably timed out. In this case, the Web browser preferably no longer attempts to obtain the URL from another Web browser, and the URL is obtained from the regular Web server using regular HTTP protocol. Before starting to download the data package from the regular Web server, the Web browser preferably transmits a response message indicating that this particular Web browser is downloading the data package.

On the other hand, if a response message is received, the Web browser obtains the URL from the other Web browser which indicated that it had the URL in the local cache. Preferably, Web browsers across the network record the URLs and the address from which the response message originated for future use, such that these Web browsers would be able to download the URL at a future time without first transmitting the request message.

Once the Web browser is able to locate a data package on a neighboring Web browser, the Web browser attempts to download the data package. The downloading process is performed with a suitable data-transfer protocol, such as HTTP or FTP. If a time-out or other failure occurs during the processing of data package transfer, the receiving Web browser preferably performs substantially the entire procedure more than once. More preferably, the number of permitted attempts to retry the transfer is configurable. If the process fails after these attempts have been performed, preferably the Web browser transfers the required data package or data packages from the regular Web server.

According to preferred features of this embodiment of the present invention, data package downloading is well distributed, such that the Web browsers do not obtain a data package from only a single Web browser, but rather obtain the data package from a plurality of Web browsers. Such distribution is maintained as follows.

First, preferably the number of simultaneous data package transfers from a single Web browser is limited. If this number is exceeded, the Web browser transmits a "busy" message to other Web browsers attempting to transfer the data package. Next, preferably once a Web browser receives a message giving the location of a particular data package, the corresponding entry in the hash table for that data package is not altered every time another response message is received pertaining to this data package. The hash table is preferably altered by subsequent messages in a probabilistic manner, such that the probability that any particular entry is updated to indicate a new location of a data package is equal to $1/(\text{generation}+1)$, where 'generation' counts the number of times a response message was received for that data package.

For example, if Web browser "A" transmits a response message indicating that data package "X" is on the local

cache, then initially all of the neighboring Web browsers have an entry in the hash table indicating that Web browser “A” is the location of data package “X”. If Web browser “B” then transmits a response message for data package “X”, then each Web browser preferably now alters the entry in the hash table to indicate a new location of data package “X” with a probability of about fifty percent, such that about fifty percent of the Web browsers now have an entry indicating that the data package is available from Web browser “A” and such that about fifty percent of the Web browsers now have an entry indicating that the data package is available from Web browser “B”. Thus, a good load distribution can be achieved.

The random delay (mentioned in step 2 above) chosen by a browser is proportional to the number of currently served browsers, or the number of browsers currently downloading data packages from that browser, and inversely proportional to the amount of the data package already downloaded by it. This way the browsers more eligible to download from are more likely to be chosen by other browsers to serve these data packages.

While the invention has been described with respect to a limited number of embodiments, it will be appreciated that many variations, modifications and other applications of the invention may be made.

What is claimed is:

1. A method for distributing data packages across a network, the network featuring an external server for serving at least one data package, the external server being a dedicated server, the steps of the method being performed by a data processor, the method comprising the steps of:

- (a) providing a plurality of peer clients attached to the network and providing a list of data packages, said data packages being stored by each of said plurality of peer clients, each data package of said data packages having an entry in said list, said entry indicating a unique identifier for said data package and a location of said data package in at least one of said plurality of peer clients;
- (b) examining said list of data packages by a first peer client to find an entry for a required data package; and
- (c) if said entry for said data package is present on said list of data packages of said first peer client, retrieving said data package from said location at another of said plurality of peer clients according to said entry for said data package.

2. The method of claim 1, wherein said list of data packages is stored on the external server.

3. The method of claim 1, wherein said list of data packages is stored on at least said first peer client.

4. The method of claim 3, wherein alternatively said entry for said data package is absent from said list of data packages of said first peer client, the method further comprising the steps of:

- (d) sending a request message for said data package by said first peer client to at least one other peer client; and
- (e) if a response message is received by said first peer client from said at least one other peer client, retrieving said data package from said at least one other peer client by said first peer client.

5. The method of claim 4, the method further comprising the step of:

- (f) altering said list of data packages being stored by at least said first peer client for indicating said location of said data package according to said response message.

6. The method of claim 5, wherein said request message and said response message are transmitted to said plurality of peer clients by broadcasting.

7. The method of claim 5, wherein said request message and said response message are transmitted to said plurality of peer clients by multicasting.

8. The method of claim 5, wherein said request message and said response message are transmitted to said plurality of peer clients by polling each peer client individually.

9. The method of claim 5, wherein if said response message is not received from said at least one other peer client by said first peer client, the method further comprises the step of:

- (g) obtaining said data package by said first peer client from the external server.

10. The method of claim 9, further comprising the step of sending a response message by said first peer client to said at least one other peer client substantially before said first peer client obtains said data package from the external server.

11. The method of claim 10, wherein said list of data packages is stored on each of said plurality of peer clients, the method further comprising the steps of:

- (h) receiving said response message from said first peer client by said at least one other peer client; and
- (i) altering said list of data packages being stored by said at least one other peer client for indicating said location of said data package according to said response message.

12. The method of claim 10, wherein said list of data packages is stored on each of said plurality of peer clients, the method further comprising the steps of:

- (h) receiving said response message from said first peer client by said at least one other peer client; and
- (i) altering said list of data packages being stored by said at least one other peer client for indicating said location of said data package according to a probabilistic function.

13. The method of claim 1, wherein said probabilistic function is performed according to a set of equations:

$$\text{New location} = \begin{cases} \text{Old location} & Po(x) = 1 / (\text{generation} + 1) \\ \text{New location} & Pn(x) = 1 - 1 / (\text{generation} + 1) \end{cases}$$

wherein Pn(x) is a probability that said new location is substituted for said old location, Po(x) is a probability that said old location is retained, and “generation” indicates how many times said location had been previously changed.

14. The method of claim 10, further comprising the steps of:

- (h) receiving said response message from said first peer client by said at least one other peer client; and
- (i) retrieving said data package from said first peer client by said at least one other peer client substantially after said first peer client has obtained said data package.

15. The method of claim 1, wherein an upper limit is predetermined for a number of said plurality of peer clients served substantially simultaneously by said at least one other peer client, such that if a number of said plurality of peer clients served substantially simultaneously by said at least one other peer client is greater than said upper limit, the method further comprises the step of:

- (d) sending a busy message from said at least one other peer client to said first peer client.

16. The method of claim 1, wherein the external server is a Web server, and said plurality of peer clients is a plurality of Web browsers.

17. The method of claim 1, wherein the external server is a BackWeb™ server, and said plurality of peer clients is a plurality of BackWeb™ clients.

15

18. The method of claim 1, wherein said unique identifier for said data package is an MD5 digest of said data package.

19. The method of claim 1, wherein the step of retrieving said data package is performed according to a protocol based on TCP/IP.

20. The method of claim 19, wherein said protocol is HTTP.

21. The method of claim 19, wherein said protocol is FTP.

22. A method for distributing data packages across a network, the network featuring a server for serving at least one data package, and a plurality of peer clients for storing said data packages, the steps of the method being performed by a data processor, the method comprising the steps of:

- (a) providing a list of said data packages stored by said plurality of peer clients, each of said data packages having an entry in said list, said entry including a unique identifier for said data package and a location of said data package within at least one of said plurality of peer clients;
- (b) searching, at a first peer client, said list to find a particular entry at a second peer client for a required data package; and,
- (c) if said particular entry is found, retrieving, at said first peer client, said data package from said second peer client, according to said particular entry; and,

16

(d) if said particular entry is not found, retrieving, at said first peer client, said data package from said server.

23. A method for distributing data packages across a network, the network featuring a server for serving at least one data package, and a plurality of peer clients for storing said data packages, the steps of the method being performed by a data processor, the method comprising the steps of:

- (a) providing at each of said plurality of peer clients a local list of data packages stored by said peer client, each of said data packages having an entry in said local list;
- (b) requesting, at a first peer client, a required data package;
- (c) searching at a second peer client, said local list, to find a particular entry for said required data package;
- (d) if said particular entry is found, retrieving, at said first peer client, said data package from said second peer client, according to said particular entry; and,
- (e) if said particular entry is not found, retrieving, at said first peer client, said data package from said server.

* * * * *