(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 February 2006 (23.02.2006)

PCT

(10) International Publication Number
WO 2006/020823 A1

(51) International Patent Classification[7]: H04L 29/08, 29/14, 29/06

(21) International Application Number: PCT/US2005/028663

(22) International Filing Date: 11 August 2005 (11.08.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/601,431 13 August 2004 (13.08.2004) US

(71) Applicant *(for all designated States except US)*: **CITRIX SYSTEMS, INC.** [US/GB]; Cambourne Business Park, Cambourne, Cambridge CB3 6DW (GB).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **RAO, Goutham, P.** [US/US]; 6963 Starling Valley Drive, San Jose, CA 95120 (US). **BRUEGGEMANN, Eric** [US/US]; 5642 Stevens Creek Boulevard, #606, Cupertino, CA 95014 (US). **RODRIGUEZ, Robert** [US/US]; 5647 Wells Court, San Jose, CA 95123 (US).

(74) Agents: **LANZA, John, D.** et al.; Lahive & Cockfield, LLP, 28 State Street, Boston, MA 02109 (US).

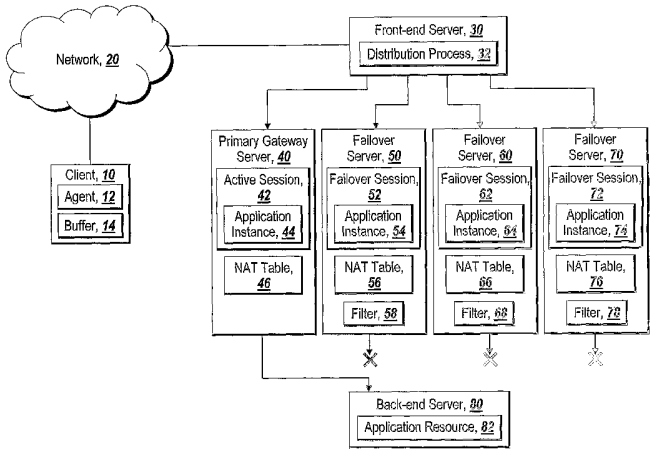
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

**Published:**
— *with international search report*

*For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.*

(54) **Title:** A METHOD FOR MAINTAINING TRANSACTION INTEGRITY ACROSS MULTIPLE REMOTE ACCESS SERVERS




(57) **Abstract:** A system for providing failover redundancy in a remote access solution that includes at least one application resource on a back-end server is discussed. The system further includes multiple gateway servers. One of the multiple gateway servers is designated as a primary gateway server while the other servers are designated as failover gateway servers. Each of the multiple gateway servers hosts a session with at least one executing application instance for the same application with each of the sessions on the failover gateway servers being maintained in the same state as the session on the primary gateway server. The primary gateway server is the only one of the gateway servers that is allowed to communicate with the application resource(s). The system further includes a client device that is in communication over a VPN with the primary gateway server. The client device receives output of the application instance executing in the session on the primary gateway server over the VPN. The client device also sends input to the primary gateway server over the VPN. The received output is displayed on a viewer by the client device.

WO 2006/020823 A1

# A METHOD FOR MAINTAINING TRANSACTION INTEGRITY ACROSS MULTIPLE REMOTE ACCESS SERVERS

## Related Application

This application claims the benefit of U.S. Provisional Patent Application Number 60/601,431, entitled "System And Method For Assuring Redundancy In Remote Access Solutions", filed August 13, 2004.

## Field of the Invention

The illustrative embodiment of the present invention relates generally to remote access servers, and more particularly to a system and method for maintaining transaction integrity in a remote access solution during failure.

## Background

Remote access solutions enable workers away from a company to securely access the company network. Through the use of IPSec VPNs or SSL VPNs remote users are able to access a company network in a secure manner. Access from a client device may be routed through one or more gateway servers which are utilized in maintaining the remote access session. With conventional remote access solutions, failure in a gateway server maintaining the remote access session results in a terminated session and lost data.

To address the issue of gateway server failure, a number of different conventional techniques have been applied in an attempt to ensure redundancy and full availability of system resources in the event of hardware or software failure. In an active/passive server arrangement an active server hosts a number of executing processes and applications. The active server may be hosting an IPSec VPN-based session or an SSL based session. One or more other servers are designated as backup or "failover" servers. The failover servers include the capability of executing the same applications and processes that are executing on the active server but the failover

server(s) do not execute the applications and processes until notified of a problem with the active server. The failover servers are known as "passive" servers in this arrangement because they may be thought of as quietly waiting to execute the applications and processes that are being executed on the active server while the backup server is operating in failover mode. Once notified of a problem with the active server, one of the failover servers is selected as the new active server, receives the last saved state/session information from the formerly "active" server and proceeds to execute in the manner in which the previous active server was executing prior to the detected failure/problem. Unfortunately, the active/passive arrangement results in a loss of data during the transition from the first active server to the newly designated active server.

It will be appreciated that the failover servers may be located on either the same or a separate physical node. An implementation that includes a failover server on the same physical node runs the risk that the failure causing the failover is associated with a physical node element that will also impact the failover server. Locating the failover server on a separate physical node that includes separate physical resources (e.g. memory, disk arrays, motherboard, etc.) lowers the risk of a single physical element causing both the active and failover servers to fail but increases the cost of the overall system in both hardware acquisition and management.

Another technique to provide redundancy amongst servers is to arrange servers in an active/active arrangement. In an active/active arrangement, both the active and passive failover servers are executing the same set of applications and processes. When the first active server goes down, the second active server allocates resources to those application instances and processes that were previously being handled by the failed server.

Unfortunately, neither the active/passive nor the active/active failover server arrangement lend themselves to preserving session state for a remote access session. Neither conventional failover implementation allows a remote session state to be mirrored in a failover server such that the failover server may be transitioned to without terminating the existing session and losing session data. It would be desirable to be able

to preserve an existing remote session and its session data in the event of a failure affecting an active server hosting the remote session.

Brief Summary

The illustrative embodiment of the present invention allows a remote session state to be synchronized between a primary gateway server and active failover servers. Incoming data sent from the client device to the primary gateway server hosting the remote session is transmitted to the active failover servers which are hosting mirror sessions. Acknowledgements of the receipt of the incoming data are not sent back to the client device from the primary gateway device until the distribution of the data to the failover servers has been confirmed. Requests to application resources from the remote session on the primary gateway server are allowed while those from the mirror sessions running on the failover servers are intercepted and discarded so as to prevent application resource conflicts. Data received in response to a request from the primary gateway server to an application resource running on back-end servers is similarly mirrored to the failover servers before being transmitted to the client device. The response is forwarded to the client device from the primary gateway server while being intercepted from the mirror sessions running on the failover servers. In the event of a failure affecting the primary gateway server, one of the failover servers hosting the mirrored sessions is elected as the primary active server and its permissions are changed to allow the newly elected primary gateway to communicate with the client device and with the application resources.

In one aspect of the present invention, a system for providing failover redundancy in a remote access solution includes at least one application resource on a back-end server. The system further includes multiple gateway servers. One of the multiple gateway servers is designated as a primary gateway server while the other servers are designated as failover gateway servers. Each of the multiple gateway servers hosts a session with at least one executing application instance for the same application with each of the sessions on the failover gateway servers being maintained in the same state as the session on the primary gateway server. The primary gateway server is the only one of the gateway servers that is allowed to communicate with the application

resource(s). The system further includes a client device that is in communication over a VPN with the primary gateway server. The client device receives output of the application instance executing in the session on the primary gateway server over the VPN. The client device also sends input to the primary gateway server over the VPN. The received output is displayed on a viewer by the client device.

In another aspect of the present invention, a method for providing failover redundancy in a remote access solution, includes the step of providing at least one application resource on a back-end server. The method further includes the step of designating one of multiple gateway servers as a primary gateway server while designating the other servers as failover gateway servers. Each of the gateway servers hosts a session with at least one executing application instance for the same application. The primary gateway server is the only one of the multiple gateway servers that is allowed to communicate with the at least one application resource. The method further includes the step of maintaining the sessions on the failover gateway servers in the same state as the session on the primary gateway server. Additionally, the method includes the step of receiving at a client device in communication over a VPN with the primary gateway server the output of the at least one application instance executing in the session on the primary gateway server with the received output being displayed on a viewer by the client device.

Brief Description of the Drawings

The invention is pointed out with particularity in the appended claims. The advantages of the invention described above, as well as further advantages of the invention, may be better understood by reference to the following description taken in conjunction with the accompanying drawings, in which:

**Figure 1** depicts an environment suitable for practicing the illustrative embodiment of the present invention utilizing a front-end server in front of multiple gateway servers;

**Figure 2A** is a block diagram depicting a typical computer useful in the present invention;

**Figure 2B** depicts an embodiment of the computer system in which the processor communicates directly with main memory via a memory port;

**Figure 3** is a flowchart of the sequence of steps followed by the illustrative embodiment of the present invention to mirror data directed to the active session on the primary gateway server to the failover sessions on the failover servers;

**Figure 4** is a flowchart of the sequence of steps followed by the illustrative embodiment of the present invention to request data from an application resource and mirror the response to the failover sessions;

**Figure 5** depicts an alternate embodiment in which application screen buffers are kept synchronized for a remote session requested by a thin client or kiosk; and

**Figure 6** is a flowchart of the sequence of steps followed by the illustrative embodiment to detect failure in a primary gateway server and transition to a new primary gateway server selected from the failover servers.

Detailed Description

The illustrative embodiment of the present invention provides the ability to mirror session state from an active primary gateway server to an active failover server. By copying all of the received input from the client and all responses received from application resources, the failover sessions are able to be maintained in the same state as the session on the primary gateway server with which the client device is communicating. In the event of failure, the existing session can be transitioned to a failover server (which becomes the primary gateway server) without disrupting the session and with minimal loss of data.

**Figure 1** depicts an environment suitable for practicing the illustrative embodiment of the present invention. A client 10 that includes an agent 12 communicates over a network 20 with a front end server 30. The front end server 30 includes a distribution process 32. The distribution process 32 copies and distributes data received from the client device to an active remote session 42 on a primary gateway server 40 and mirror failover sessions 52, 62 and 72 on failover servers 50, 60 and 70.

Application instances 44, 54, 64 and 74 running on the respective active and failover sessions 42, 52, 62 and 72 may request data from an application resource 82 hosted by a back end server 80. Responses from the application resource 82 to an executing application instance's request for data are returned to the distribution process 32 on the front-end server 30 for distribution to the active session 42 and failover sessions 52, 62 and 72.

Referring now to **Figure 1** in more detail, the client 10 may be communicating over the network 20 by establishing an IPSEC VPN or an SSL VPN to a remote session (active session 42) established on the primary gateway server 40. The client may include a buffer 14 in which the agent 12 buffers a copy of the data sent to the primary gateway server 40 until receiving an acknowledgement from the primary gateway server. The network 20 may be may be the Internet, a local area network (LAN), a wide area network (WAN), an extranet, an intranet, wireless network, satellite network, or some other type of network capable of allowing the client 10 to communicate with the active session 42 on the primary gateway server 40. The primary gateway server 40, and failover servers 50, 60 and 70 may include Network Address Translation (NAT) tables 46, 56, 66 and 76 allowing the servers to perform NAT for an IP address assigned to the client 10 during the establishment of the remote session by the primary VPN server. As noted above, active session 42, and failover sessions 52, 62 and 72 may include executing application instances 44, 54, 64 and 74 which request data from an application resource 82 on a back-end server 80. As will be explained in greater detail below, failover servers 50, 60 and 70 also include filters 56, 66 and 76 which filter and discard requests to the application resource 82 from the failover sessions 52, 62, and 72 while the failover servers are acting in failover mode. Responses form the application resource 82 to the request from an executing application instance 44 in active session 42 are routed to the distribution process 32 for distribution to the active session and failover sessions 52, 62 and 72. Those skilled in the art will recognize that the location of the distribution process 32 may vary within the scope of the present invention in that it may be located in different network accessible locations other than on the front-end server and still perform the functions described herein.

Still referring to **Figure 1**, and in more detail, in many embodiments, the client 10, front-end server 30 and the back-end server 80 are provided as personal computers or computer servers, of the sort manufactured by the Hewlett-Packard Corporation of Palo Alto, California or the Dell Corporation of Round Rock, TX. **Figures 2A and 2B** depict block diagrams of a typical computer 200 useful as the client 10, front-end server 30 and the back-end server 80 in those embodiments. As shown in **Figures 2A and 2B**, each computer 200 includes a central processing unit 202, and a main memory unit 204. Each computer 200 may also include other optional elements, such as one or more input/output devices 230a-230n (generally referred to using reference numeral 230), and a cache memory 240 in communication with the central processing unit 202.

The central processing unit 202 is any logic circuitry that responds to and processes instructions fetched from the main memory unit 204. In many embodiments, the central processing unit is provided by a microprocessor unit, such as: the 8088, the 80286, the 80386, the 80486, the Pentium, Pentium Pro, the Pentium II, the Celeron, or the Xeon processor, all of which are manufactured by Intel Corporation of Mountain View, California; the 68000, the 68010, the 68020, the 68030, the 68040, the PowerPC 601, the PowerPC604, the PowerPC604e, the MPC603e, the MPC603ei, the MPC603ev, the MPC603r, the MPC603p, the MPC740, the MPC745, the MPC750, the MPC755, the MPC7400, the MPC7410, the MPC7441, the MPC7445, the MPC7447, the MPC7450, the MPC7451, the MPC7455, the MPC7457 processor, all of which are manufactured by Motorola Corporation of Schaumburg, Illinois; the Crusoe TM5800, the Crusoe TM5600, the Crusoe TM5500, the Crusoe TM5400, the Efficeon TM8600, the Efficeon TM8300, or the Efficeon TM8620 processor, manufactured by Transmeta Corporation of Santa Clara, California; the RS/6000 processor, the RS64, the RS 64 II, the P2SC, the POWER3, the RS64 III, the POWER3-II, the RS 64 IV, the POWER4, the POWER4+, the POWER5, or the POWER6 processor, all of which are manufactured by International Business Machines of White Plains, New York; or the AMD Opteron, the AMD Athalon 64 FX, the AMD Athalon, or the AMD Duron processor, manufactured by Advanced Micro Devices of Sunnyvale, California.

In the embodiment shown in **Figure 2A**, the processor 202 communicates with main memory 204 via a system bus 220 (described in more detail below). **Figure 2B** depicts an embodiment of a computer system 200 in which the processor communicates

directly with main memory 204 via a memory port. For example, in **Figure 2B** the main memory 204 may be DRDRAM.

**Figures 2A and 2B** depict embodiments in which the main processor 202 communicates directly with cache memory 240 via a secondary bus, sometimes referred to as a "backside" bus. In other embodiments, the main processor 202 communicates with cache memory 240 using the system bus 220. Cache memory 240 typically has a faster response time than main memory 204 and is typically provided by SRAM, BSRAM, or EDRAM.

In the embodiment shown in **Figure 2A**, the processor 202 communicates with various I/O devices 230 via a local system bus 220. Various buses may be used to connect the central processing unit 202 to the I/O devices 230, including a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, or a NuBus. For embodiments in which the I/O device is a video display, the processor 202 may use an Advanced Graphics Port (AGP) to communicate with the display. **Figure 2B** depicts an embodiment of a computer system 200 in which the main processor 202 communicates directly with I/O device 230b via HyperTransport, Rapid I/O, or InfiniBand. **Figure 2B** also depicts an embodiment in which local buses and direct communication are mixed: the processor 202 communicates with I/O device 230a using a local interconnect bus while communicating with I/O device 230b directly.

A wide variety of I/O devices 230 may be present in the computer system 200. Input devices include keyboards, mice, trackpads, trackballs, microphones, and drawing tablets. Output devices include video displays, speakers, inkjet printers, laser printers, and dye-sublimation printers. An I/O device may also provide mass storage for the computer system 200 such as a hard disk drive, a floppy disk drive for receiving floppy disks such as 3.5-inch, 5.25-inch disks or ZIP disks, a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive, tape drives of various formats, and USB storage devices such as the USB Flash Drive line of devices manufactured by Twintech Industry, Inc. of Los Alamitos, California.

In further embodiments, an I/O device 230 may be a bridge between the system bus 220 and an external communication bus, such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, or a Serial Attached small computer system interface bus.

General-purpose desktop computers of the sort depicted in **Figures 2A and 2B** typically operate under the control of operating systems, which control scheduling of tasks and access to system resources. Typical operating systems include: MICROSOFT WINDOWS, manufactured by Microsoft Corp. of Redmond, Washington; MacOS, manufactured by Apple Computer of Cupertino, California; OS/2, manufactured by International Business Machines of Armonk, New York; and Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah, among others.

For embodiments in which the client 10 is a mobile device, the client device may be a JAVA-enabled cellular telephone, such as the i50sx, i55sr, i58sr, i85s, i88s, i90c, i95cl, or the im11000, all of which are manufactured by Motorola Corp. of Schaumburg, Illinois, the 6035 or the 7135, manufactured by Kyocera of Kyoto, Japan, or the i300 or i330, manufactured by Samsung Electronics Co., Ltd., of Seoul, Korea. In other embodiments in which the client 10 is mobile, it may be a personal digital assistant (PDA) operating under control of the PalmOS operating system, such as the Tungsten W, the VII, the VIIx, the i705, all of which are manufactured by palmOne, Inc. of Milpitas, California. In further embodiments, the client device 20 may be a personal digital assistant (PDA) operating under control of the PocketPC operating system, such as the iPAQ 4155, iPAQ 5555, iPAQ 1945, iPAQ 2215, and iPAQ 4255, all of which manufactured by Hewlett-Packard Corporation of Palo Alto, California, the ViewSonic V36, manufactured by ViewSonic of Walnut, California, or the Toshiba PocketPC e405, manufactured by Toshiba America, Inc. of New York, New York. In still other embodiments the client device is a combination PDA/telephone device such as the Treo 180, Treo 270 or Treo 600, all of which are manufactured by palmOne, Inc. of Milpitas, California. In still further embodiment, the client device 20 is a cellular telephone that operates under control of the PocketPC operating system, such as the MPx200, manufactured by Motorola Corp.

The functions performed by the components depicted in the architecture displayed in **Figure 1** may be further explained with reference to **Figure 3**. **Figure 3** is a flowchart of the sequence of steps followed by the present invention to mirror data directed to the active session on the primary gateway server to the failover sessions on the failover servers. Following the establishment of the initial remote active session 42 on the primary gateway server 40(following authentication of the client 10) and the establishment of the corresponding failover sessions 52, 62 and 72 on the failover servers 50, 60 and 70, the front-end server 30 receives input from the client directed to the active session (step 300). The distribution process 32 on the front-end server 30 copies and distributes the input data to the active session 42 and failover sessions 52, 62 and 72 (step 302). The application instances in the active 42 and failover sessions 52, 62 and 72 then process the received data (step 304).

The distribution process 32 then queries the active and failover sessions 42, 52, 62 and 72 to determine if the session states are synchronized (step 306). The verification of synchronization is necessary due to possible variances in processing speed amongst the servers and latency delays in the network. In alternate implementations, the primary gateway server 40 and the failover servers 40, 50 and 60 are configured to report their session state to the distribution process 32 without first being queried. In another implementation, a different process other than the distribution process has the responsibility of verifying the synchronization of the states of the sessions 42, 52, 62 and 72. If the session states are determined not to be synchronized (step 307), a delay is implemented pending a further query from the distribution process to determine the synchronization status (step 306). If the session states are synchronized ( step 307), the results of the processing performed by the application instance 44 in the active session 42 on the primary gateway server 40 are reported to the client (step 308). Attempts by application instances in the failover sessions 52, 62 and 72 are filtered by the respective filters 46, 56 and 66 on the network stack and the packets discarded.

A similar process is performed to keep the active and failover session states synchronized during a request by an application instance executing in a session for an application resource. **Figure 4** is a flowchart of the sequence of steps followed by the illustrative embodiment of the present invention to request data from an application

resource and mirror the response to the failover sessions. The sequence begins when the application instance 44 in the active session 42 (and the corresponding application instances 54, 64 and 74 in the failover sessions 52, 62 and 72) identify a needed application resource (step 320). The application instances 44, 54, 64 and 74 from all of the sessions then request the application resource 82 from the back-end server 80 (step 322). The requests originating from the application instances 54, 64 and 74 in the failover sessions 52, 62 and 72 are filtered by filters 58, 68 and 78 on the network stack of the respective failover servers 50, 60 and 70. The filtered packets are discarded (step 324). The request from the active session 42 is forwarded to the application resource 82 on the back-end server 80 (step 326). The response to the request is routed to the distribution process 32 on the front-end server (step 328). In one implementation, the response is sent to the primary gateway server 40 which is hosting the active session 42 first and is forwarded to the distribution process 32. In another implementation, a separate distribution process different from the one used to accept input data from the client may be used to distribute application resource responses.

Continuing with **Figure 4**, following the receipt of the response from the application resource, the distribution process forwards the response data to the active and failover sessions 42, 52, 62 and 72 (step 330). The application instances 44, 54, 64 and 74 in the active session 42 and failover sessions 52, 62 and 72 then process the resource data (step 332). As with the data received from the client, a query is made to determine if the session states are synchronized (step 333). If the states are not synchronized the query is repeated until an affirmative response is received. Once the session states are synchronized, the application instance in the active session 42 communicates the results of its data processing to the client 10. The application instances 54, 64 and 74 on the failover servers also attempt to communicate their results to the client 10, but the packets are intercepted and discarded by the network filters 58, 68 and 78 on the respective failover servers 50, 60 and 70 (step 334).

In another aspect of the illustrative embodiment, the present invention may also be implemented in a "thin-client" or kiosk architecture where all of the processing is taking place on the server and only screen data is being pushed down to the client 10 from the remote session 42. **Figure 5** depicts an alternate embodiment in which

application screen buffers are kept synchronized for a remote session requested from a thin client or kiosk The client 400 may be a thin client with limited processing power such as a PDA or cell-phone or may be a publicly available kiosk terminal with a display and input device. The client establishes the remote session over the network 402 to a primary gateway server 410. The connection may be use RDP (Remote Display Protocol) from Microsoft Corporation or ICA from Citrix Systems, Inc. of Fort Lauderdale, Florida. The session may be established when the client 400 logs in via a secure web URL exposed by the primary gateway server. The primary gateway server accepts the connection request and establishes the remote active session 412. The illustrative embodiment also establishes failover sessions 422 and 432 on failover servers 420 and 430. The active session 412 and failover sessions 422 and 432 are similarly provisioned and include executing application instances 414, 424 and 434 respectively. Each of the application instances has an associated screen buffer 416, 426 and 436 respectively. A distribution process 418 on the primary gateway server copies and distributes input data received from the client 400 to each of the sessions 412, 422 and 432 in the manner set forth above. In an alternate implementation, the distribution process 418 may be located on a front end server. The application instances 414, 424 and 434 may request an application resource 442 from a back-end server 440 with the request from the primary gateway server being transmitted and the requests from the failover servers 420 and 430 being intercepted and discarded. As before, following the processing performed by the executing application instances 414, 424 and 434, a response to the client 400 is delayed until all of the screen buffers 416, 426 and 436 are in the same state. The contents (or changed contents when using optimization techniques) of the screen buffer 416 on the primary gateway server are then pushed down over the connection to the client 400. Attempts by the application instances 424 and 434 to push the contents of their screen buffers 426 and 436 to the client 400 are intercepted and the packets discarded.

The synchronization of session states performed by the present invention enables the transition of a remote session from a failed gateway server to a replacement gateway server without terminating a remote session and ensures a minimal loss of data. An agent 12 buffers a copy of the data sent from the client 10 (if a reliable protocol is being used such as TCP). The present invention does not return an acknowledgement of the

receipt of data from the client until all of the session states have been synchronized. The agent 10 therefore buffers a copy of any sent data until the acknowledgement is received from the primary gateway server and then discards the data. In the event of any subsequent failure of the primary gateway server 40, the data sent by the client to the primary gateway server is present in one of the failover sessions. Similarly, the data received from the application resource is also distributed to both the active and failover sessions so that they remain synchronized. If an acknowledgement is not received from the primary gateway server, the buffered data is resent to the newly appointed primary gateway server.

Additionally, when the remote connection is initially established, the IP address assigned to the client 10 is distributed not just to the primary gateway server 40 which in one implementation is performing NAT for the client, but also to the failover servers 50, 60 and 70. When a failure of the primary gateway server 40 is detected, one of the failover servers 50, 60 or 70 is selected as the primary gateway server and its attributes are changed to allow it to perform that role. The filter 58, 68 or 78 present on the network stack that prevents the failover servers 50, 60 and 70 from communicating with the client 10 and application resource 82 is disabled for the newly designated primary gateway server. The newly selected primary gateway server also intercepts any communications received from the client or directed to the client using the previously sent IP address and the NAT table 58, 68 or 78 on the newly selected primary gateway. The combination of a mirrored session state and client IP awareness by the failover servers prior to the primary gateway server failing thus allows a smooth transition with minimal data loss and without the need to reestablish a new remote session.

The detection of a failure of the primary gateway server 40 may happen in a number of different ways. The method used to detect the server failure may be dependent upon the manner in which the servers are deployed. For example, the primary and gateway servers may be arranged as nodes in a clustered computer system. Nodes in a clustered computer system frequently send each other "heartbeat" signals (which are also referred to as "responsive" or "activation" signals) over private communication channels. The heartbeat signals/tokens indicate whether the nodes are active and responsive to other nodes in the clustered computer system. The heartbeat

signals are sent periodically by each of the nodes so that if one or more nodes do not receive the heartbeat signal from another node within a specified period of time, a node failure can be suspected.

The final determination of a gateway server failure may be dictated by a failover policy controlling the gateway servers. The failover policy indicates under what circumstances the primary gateway server may be considered to have failed. For example, in the clustered computing system discussed above, in implementations which have multiple failover servers, the policy may require at least two failover servers to have not received the heartbeat token from the primary gateway server within the specified time period. Alternatively, the failover policy may require all of the failover nodes to have not received the heartbeat token from the primary gateway server. Those skilled in the art will recognize that a number of other techniques may be used alone or in combination to detect gateway server failure such as pinging the primary gateway server to determine its health. The pinging may be performed following the lack of the receipt of the heartbeat token in order to verify the failure of the primary gateway server.

Once a failure of the primary gateway server has been detected, a number of different methods of selecting a new primary gateway server from the failover servers may be employed. The server may be selected based on name or through a mathematical operation performed on a server identifier. Alternatively, the selection may be based on the server processing attributes (speed, memory, etc.) or the next primary gateway server may have been previously designated by a system administrator. Other possibilities will occur to those skilled in the art.

**Figure 6** depicts a sequence of steps followed by the present invention to handle primary gateway failures. The sequence begins with the primary and failover gateway servers being designated (step 500). The designation of the failover servers activates the filters on the network stack of those servers and disables the NAT tables. When a remote session is established, the IP address assigned to the client is forwarded to the failover servers. The primary gateway server is then queried to determine its health (step 502). If the primary gateway server is determined to be healthy (step 503), the query is repeated at set intervals. If the primary gateway server is not healthy (step 503),

the new primary gateway server is selected from among the failover servers and its attributes changed so that its filter is disabled, it handles communications to and from the client IP and the NAT table on the server is activated (step 504).

The present invention may be provided as one or more computer-readable programs embodied on or in one or more articles of manufacture. The article of manufacture may be a floppy disk, a hard disk, a compact disc, a digital versatile disc, a flash memory card, a PROM, a RAM, a ROM, or a magnetic tape. In general, the computer-readable programs may be implemented in any programming language. Some examples of languages that can be used include C, C++, C#, or JAVA. The software programs may be stored on or in one or more articles of manufacture as object code.

Since certain changes may be made without departing from the scope of the present invention, it is intended that all matter contained in the above description or shown in the accompanying drawings be interpreted as illustrative and not in a literal sense. Practitioners of the art will realize that the system configurations depicted and described herein are examples of multiple possible system configurations that fall within the scope of the current invention. Likewise, the sequence of steps utilized in the illustrative flowcharts are examples and not the exclusive sequence of steps possible within the scope of the present invention. Similarly, data structures other than the ones mentioned herein may be used to hold data without departing from the scope of the present invention.

We Claim:

1. A system for providing failover redundancy in a remote access solution, comprising:

at least one application resource on a back-end server;

a plurality of gateway servers, one of the plurality of gateway servers designated as a primary gateway server with the other servers in the plurality of gateway servers designated as failover gateway servers, each of the plurality of gateway servers hosting a session with at least one executing application instance for the same application, each of the sessions on the failover gateway servers being maintained in the same state as the session on the primary gateway server, the primary gateway server being the only one of the plurality of gateway servers allowed to communicate with the at least one application resource; and

a client device in communication over a VPN with the primary gateway server, the client device receiving output of the at least one application instance executing in the session on the primary gateway server over the VPN and sending input to the primary gateway server over the VPN; the received output displayed on a viewer by the client device.

2. The system of claim 1, further comprising;

a filter on each of the failover gateway servers wherein packets transmitted by the executing application instances in each session on the failover gateway servers addressed to the at least one application resource are intercepted and discarded.

3. The system of claim 1 wherein the input received by the session on the primary gateway server from the client device over the VPN is relayed to each of the executing application instances in each of the sessions on the failover gateway servers.

4. The system of claim 1 wherein the executing application instance in the session on the primary gateway server receives a reply to a request to the at least one application resource, and the reply is duplicated and forwarded to each session on each failover gateway server.

5. The system of claim 1 wherein the primary gateway server delays the transmission of data to the client device until the sessions on each of the failover gateway servers are synchronized with the primary gateway server.

6. The system of claim 1 wherein the client device and the primary gateway server perform Network Address Translation (NAT) for an IP address assigned to the client device.

7. The system of claim 6 wherein a selected one of the failover gateway servers is selected to replace a failed primary gateway server.

8. The system of claim 7 wherein the selected one of the failover gateway servers that was selected to replace the failed primary gateway server performs the NAT for the client device IP in place of the failed primary gateway server.

9. The system of claim 7, comprising further

an associated screen buffer for each executing application instance on the plurality of gateway servers, the screen buffers associated with the application instances executing on the failover gateway servers being maintained in the same state as screen buffer for the executing application instance on the primary gateway server.

10. The system of claim 9 wherein a copy of the contents in the screen buffer on the primary gateway server is forwarded to the client device for display.

11. A method for providing failover redundancy in a remote access solution, comprising:

providing at least one application resource on a back-end server;

designating one of a plurality of gateway servers as a primary gateway server, the other servers in the plurality of gateway servers designated as failover gateway servers, each of the plurality of gateway servers hosting a session with at least one executing application instance for the same application, the primary gateway server being the only one of the plurality of gateway servers allowed to communicate with the at least one application resource;

maintaining the sessions on the failover gateway servers in the same state as the session on the primary gateway server, and

receiving at a client device in communication over a VPN with the primary gateway server the output of the at least one application instance executing in the session on the primary gateway server, the received output displayed on a viewer by the client device.

12. The method of claim 11, further comprising:

sending input to the executing application instance in the session on the primary gateway server from the client device over the VPN, and

copying the received input and forwarding the copied input to each of the application instances executing in sessions on the failover gateway servers so that the sessions in the failover gateway servers are maintained in the same state as the session on the primary gateway server.

13. The method of claim 11, further comprising:

sending from the executing application instance in the session on the primary gateway server a request to the application resource on the back-end server;

receiving a reply to the request from the application resource at the primary gateway server; and

copying the reply at the primary gateway server and forwarding the copied reply to the executing instances on the failover gateway servers.

14. The method of claim 11, further comprising:

sending from the executing application instances on the failover gateway servers a request to the application resource on the back-end server;

intercepting the requests on the respective failover gateway servers; and

discarding the requests without forwarding the requests to the application resource.

15. The method of claim 11 wherein the primary gateway server uses the Remote Frame Buffer (RFB) protocol to transmit a change in a screen buffer associated with the at least one application instance on the primary gateway server to the client device.

16. The method of claim 11 wherein the primary gateway server uses the Remote Display Protocol (RDP) to transmit a change in the screen buffer associated with the at least one application instance on the primary gateway server to the client device.

17. The method of claim 11 wherein the primary gateway server uses an ICA connection to transmit a change in a screen buffer associated with the at least one application instance on the primary gateway server to the client device.

18. The method of claim 11, further comprising:

detecting a failure in the primary gateway server; and

selecting a designated failover server as the new primary gateway server, the new primary gateway server allowed to communicate with the at least one application resource following the selection.

19. The method of claim 18 further comprising:

detecting the failure in the primary gateway server through the use of a heartbeat token protocol.

20. The method of claim 18 further comprising:

detecting the failure in the primary gateway server through pinging of the primary gateway server at pre-determined intervals.

21. The method of claim 11, further comprising:

performing Network Address Translation (NAT) at the primary gateway server for an IP address assigned to the client device.

22. The method of claim 21, further comprising:

performing Network Address Translation (NAT) for the IP address assigned to the client device at a new primary gateway server.

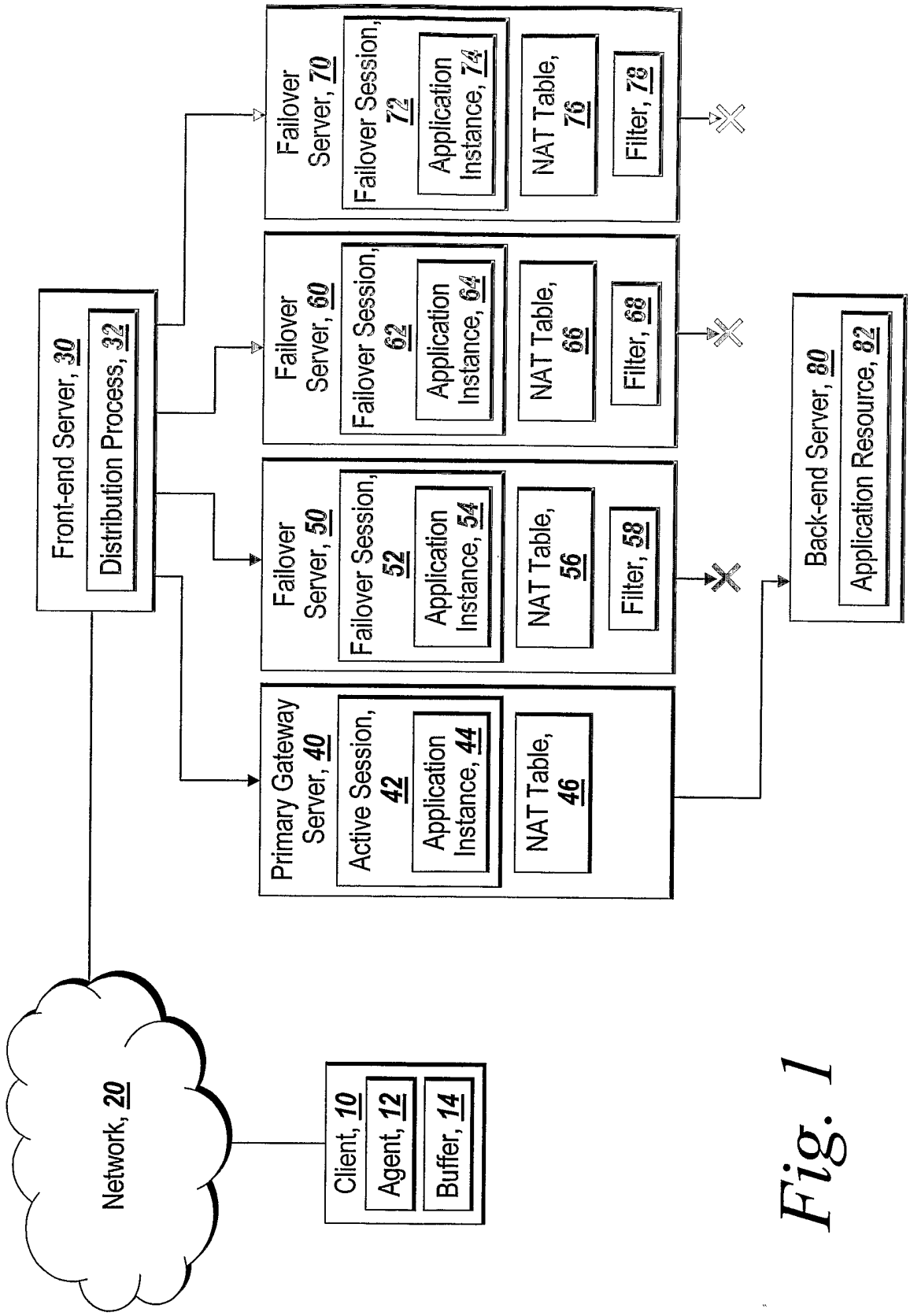

Network, 20
Client, 10
Agent, 12
Buffer, 14
Front-end Server, 30
Distribution Process, 32
Primary Gateway Server, 40
Active Session, 42
Application Instance, 44
NAT Table, 46
Failover Server, 50
Failover Session, 52
Application Instance, 54
NAT Table, 56
Filter, 58
Failover Server, 60
Failover Session, 62
Application Instance, 64
NAT Table, 66
Filter, 68
Failover Server, 70
Failover Session, 72
Application Instance, 74
NAT Table, 76
Filter, 78
Back-end Server, 80
Application Resource, 82

*Fig. 1*

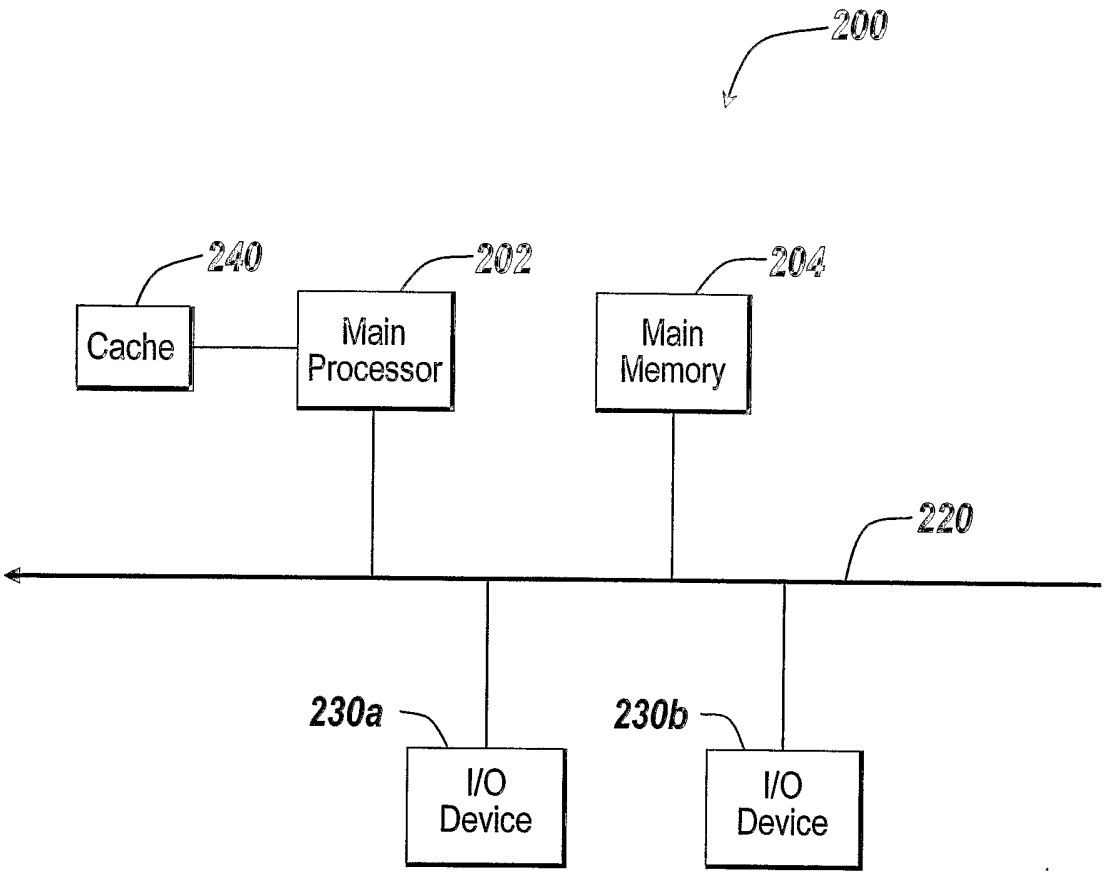

200
240
202
204
Cache
Main Processor
Main Memory
220
230a
230b
I/O Device
I/O Device

*Fig. 2A*

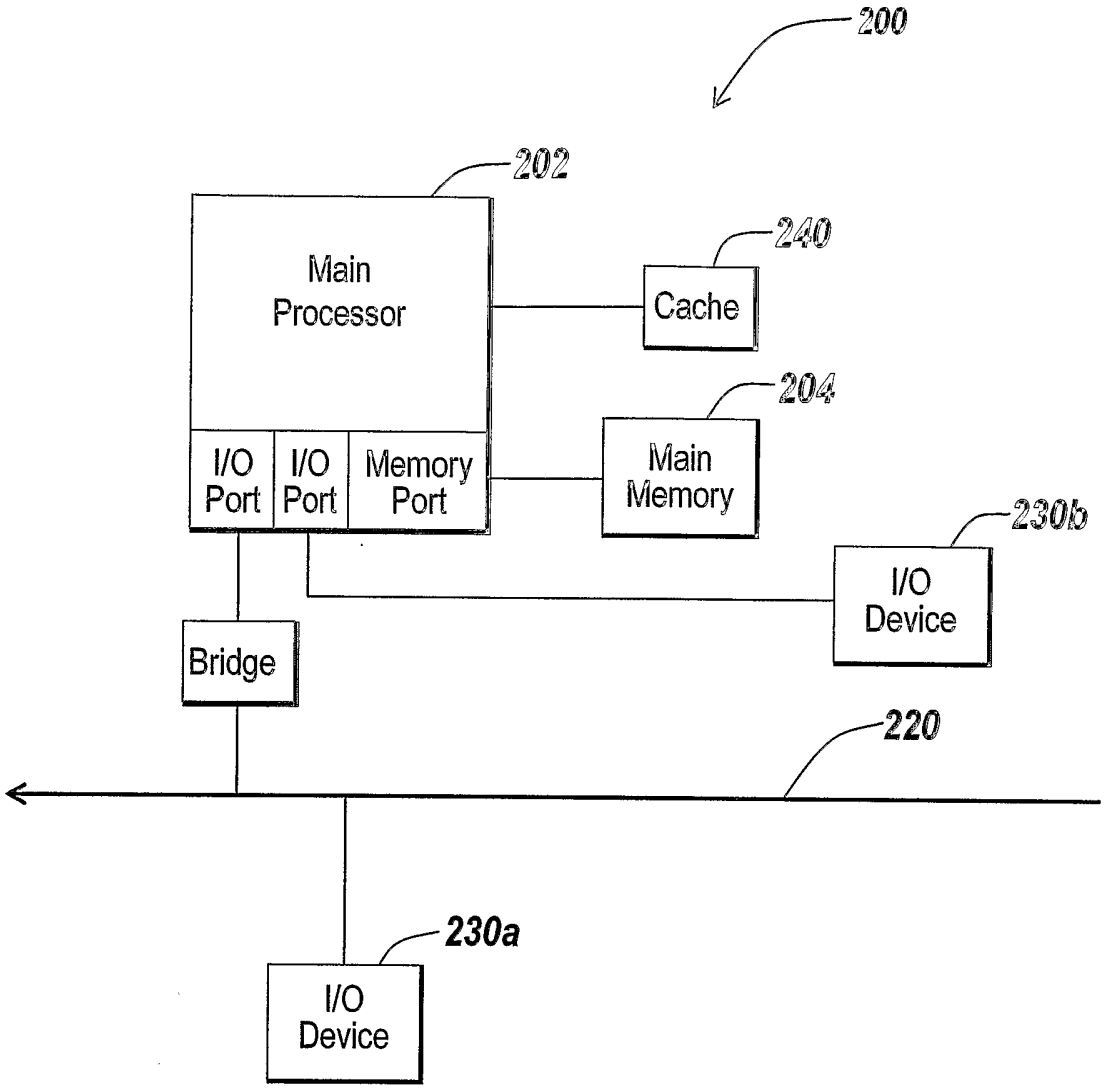

200
202
Main Processor
240
Cache
204
Main Memory
I/O Port
I/O Port
Memory Port
230b
I/O Device
Bridge
220
230a
I/O Device

*Fig. 2B*

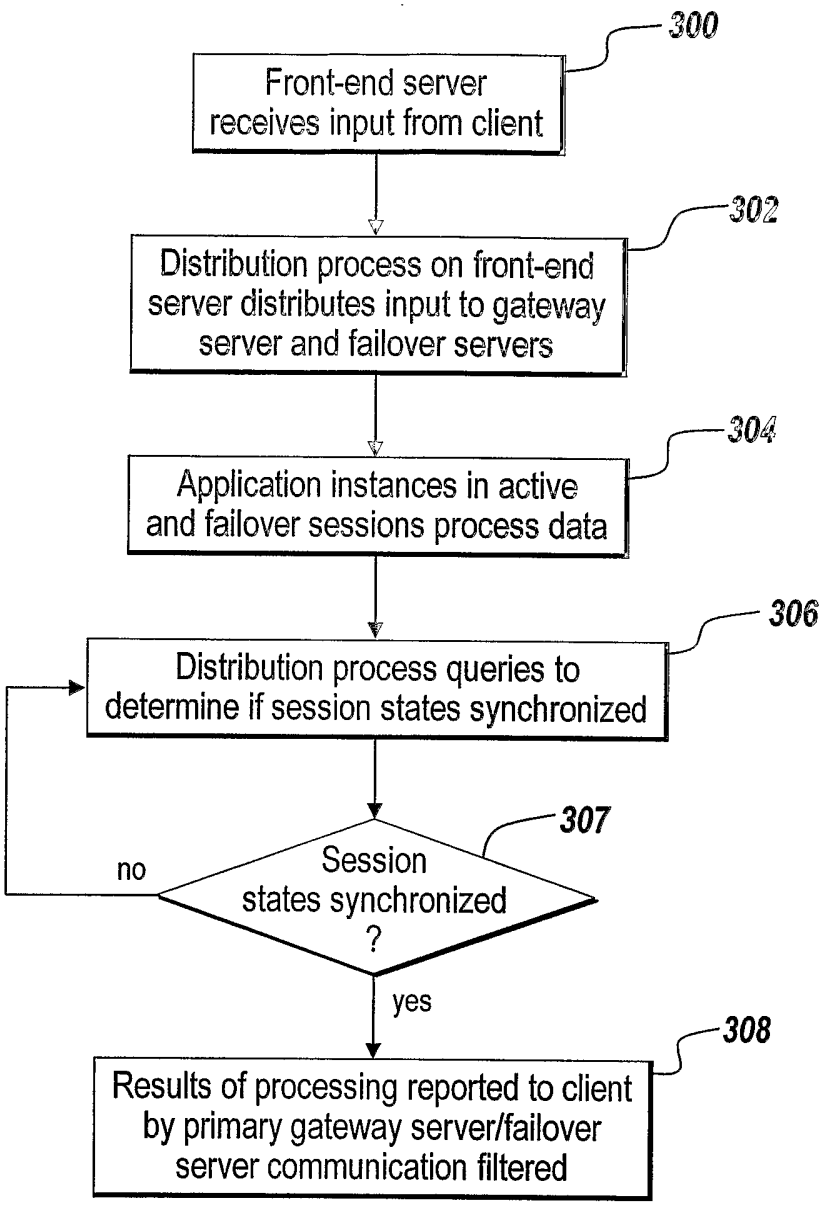

300
Front-end server receives input from client
302
Distribution process on front-end server distributes input to gateway server and failover servers
304
Application instances in active and failover sessions process data
306
Distribution process queries to determine if session states synchronized
307
Session states synchronized ?
no
yes
308
Results of processing reported to client by primary gateway server/failover server communication filtered

*Fig. 3*

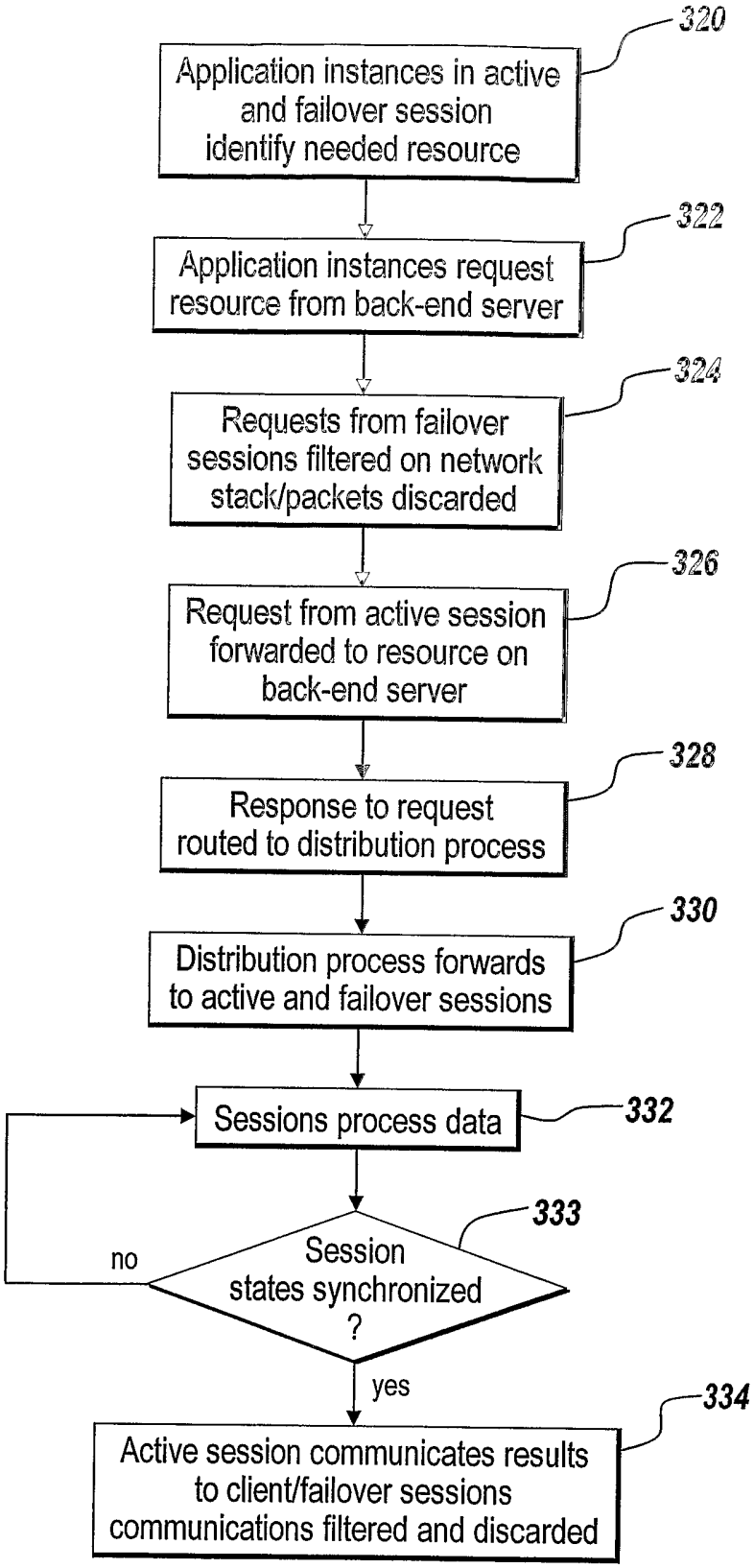

320
Application instances in active and failover session identify needed resource
322
Application instances request resource from back-end server
324
Requests from failover sessions filtered on network stack/packets discarded
326
Request from active session forwarded to resource on back-end server
328
Response to request routed to distribution process
330
Distribution process forwards to active and failover sessions
Sessions process data
332
333
Session states synchronized ?
no
yes
334
Active session communicates results to client/failover sessions communications filtered and discarded

*Fig. 4*

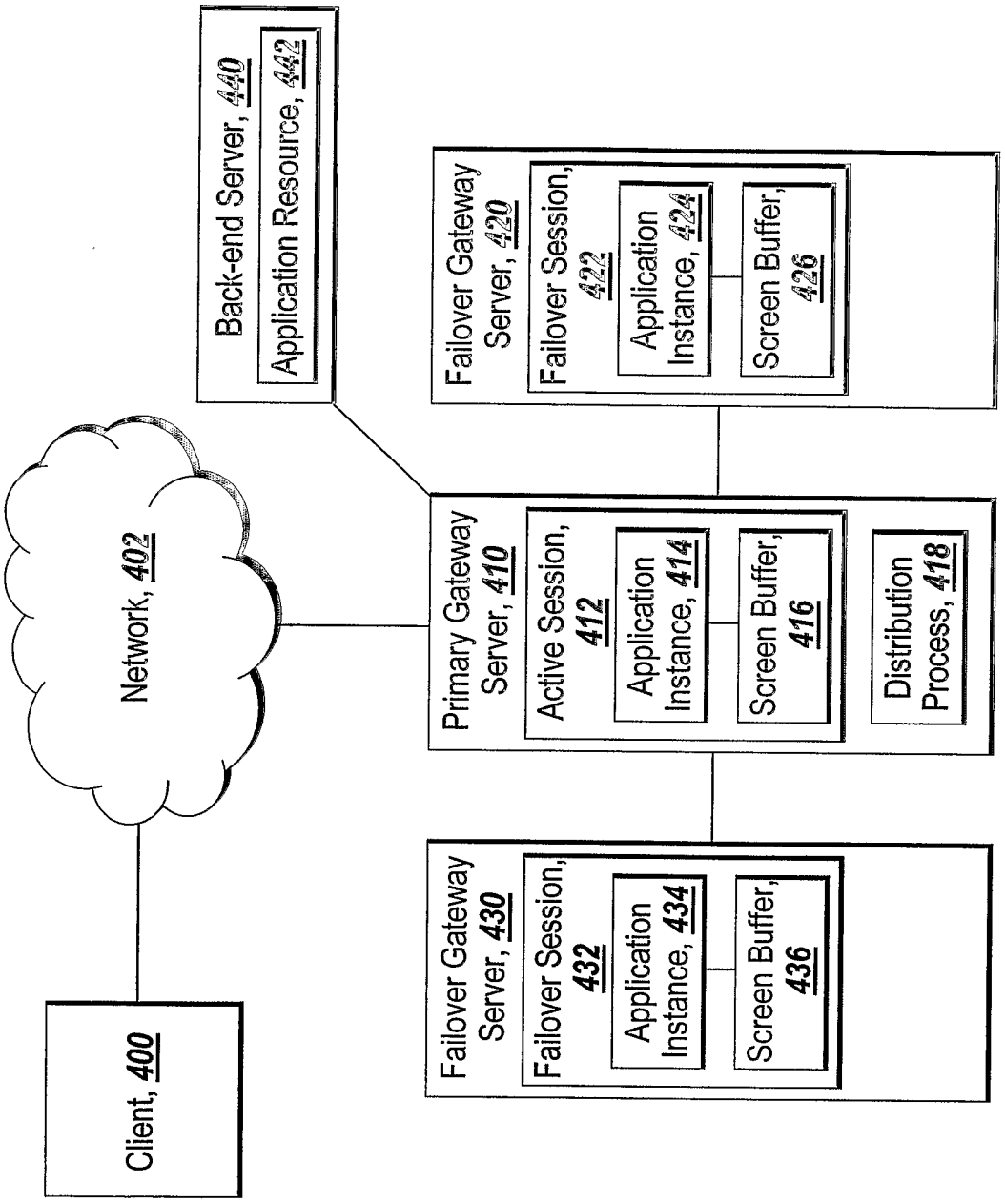

Client, 400
Network, 402
Back-end Server, 440
Application Resource, 442
Primary Gateway Server, 410
Active Session, 412
Application Instance, 414
Screen Buffer, 416
Distribution Process, 418
Failover Gateway Server, 420
Failover Session, 422
Application Instance, 424
Screen Buffer, 426
Failover Gateway Server, 430
Failover Session, 432
Application Instance, 434
Screen Buffer, 436

*Fig. 5*

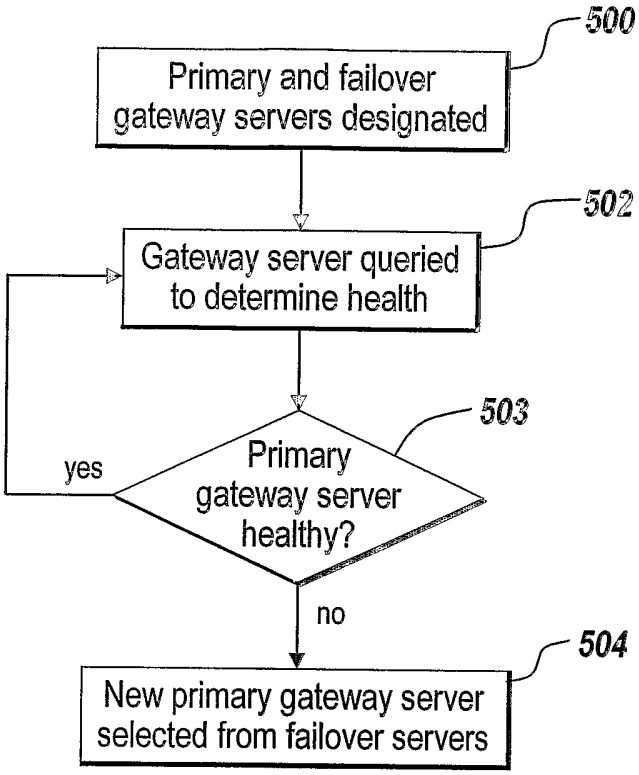

500
Primary and failover gateway servers designated
502
Gateway server queried to determine health
503
Primary gateway server healthy?
yes
no
504
New primary gateway server selected from failover servers

*Fig. 6*

# INTERNATIONAL SEARCH REPORT

Interna Application No
PCT/US2005/028663

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L29/08 H04L29/14 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ, INSPEC, COMPENDEX, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | US 2003/088698 A1 (SINGH INDERPREET ET AL)<br>8 May 2003 (2003-05-08)<br>abstract<br>paragraphs '0002!, '0004!, '0007! -<br>'0011!, '0015!, '0028!, '0053! -<br>'0059!, '0063!, '0085!<br>figures 1,8 | 1-22 |
| A | EP 1 432 209 A (HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P)<br>23 June 2004 (2004-06-23)<br>abstract<br>paragraphs '0001! - '0004!, '0014! -<br>'0025! | 1-22 |

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 9 November 2005 | 16/11/2005 |

| Name and mailing address of the ISA | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL – 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,<br>Fax: (+31-70) 340-3016 | Lopez Monclus, I. |

Form PCT/ISA/210 (second sheet) (January 2004)

# INTERNATIONAL SEARCH REPORT

Interna Application No
PCT/US2005/028663

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | US 2002/152423 A1 (MCCABE HARRY)<br>17 October 2002 (2002-10-17)<br>abstract<br>paragraphs '0001! - '0004!, '0009! -<br>'0012!, '0028! - '0033!<br>----- | 1-22 |

Form PCT/ISA/210 (continuation of second sheet) (January 2004)

# INTERNATIONAL SEARCH REPORT

Information on patent family members

Intern: Application No
PCT/US2005/028663

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| US 2003088698 A1 | 08-05-2003 | NONE | |
| EP 1432209 A | 23-06-2004 | JP 2004206695 A<br>US 2004122961 A1 | 22-07-2004<br>24-06-2004 |
| US 2002152423 A1 | 17-10-2002 | NONE | |