October 2023

METHOD AND DEVICE FOR AUTOMATED JOB FAILOVER

DEVESH GUPTA
VISA

FAIZ AHMED
VISA

METHOD AND DEVICE FOR AUTOMATED JOB FAILOVER

VISA

DEVESH GUPTA
FAIZ AHMED

TECHNICAL FIELD

[0001] The present disclosure generally relates to database management system. Particularly, but not exclusively, the present disclosure relates to a method and a system for automated failover from a primary database server to a secondary database server.

BACKGROUND

[0002] Generally, with increased computational requirements and complexity of data center systems, unplanned downtime due to workload failures has become a severe threat to reliability, availability, and scalability of data centers. For instance, the data center failure may cause various problems such as loss of vital data, non-availability of servers, mail systems, Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), etc., total cluster stagnation, and the like. Some of the factors which cause common outages include hardware malfunctioning, cyberattacks, cooling system failures, human error, backup power system failures, natural disasters, etc. In order to improve the reliability, availability, and scalability of data centers, the threats posed due to failure of data centers need to be addressed with an appropriate disaster recovery approach.

[0003] Currently, various approaches and techniques have been proposed to improve the reliability, availability, and scalability of data centers. For instance, using copies (backup) to quickly reestablish the failed datacenter after an outage is one of the various approaches available which improves the reliability, availability, and scalability of data centers. However, these conventional approaches involve switching over to a redundant datacenter until the failed datacenter is functional again, generally known as failover. For example, in an active-active datacenter configuration, most of the time, a report generated for end users should be on a single datacenter and sent from the same. If there is a failure or maintenance in that particular datacenter, in such case a manual switching process is required to run the job from other datacenter to ensure the completion of task.

[0004] Thus, existing database failure management process involves tedious manual procedure which requires enormous human resources, operational time, power, and database administer expertise to adjust parameters for a database instance and / or manage other aspects of a data repository.

[0005] The information disclosed in this background of the disclosure section is only for enhancement of understanding of the general background of the invention and should not be taken as an acknowledgement or any form of suggestion that this information forms the prior art already known to a person skilled in the art.

SUMMARY

[0006] In one non-limiting embodiment of the present disclosure, there is provided a method of managing multi data cluster failure. The method comprises scheduling one or more applications on each of a primary cluster and a secondary cluster, monitoring a status of the one or more applications scheduled in one cluster by the other cluster, updating the status of the one or more applications associated with each cluster on a Distributed File System (DFS) of respective cluster, generating output reports of the scheduled one or more applications and sending by the primary cluster to one or more users. The method comprises automatically converting secondary cluster to the primary cluster and vice-versa upon identifying the status of the primary cluster as under failure or maintenance during monitoring. Further, the method comprises updating the status of the one or more applications of the primary cluster to the DFS associated with the updated secondary cluster after a predefined threshold time. Thereafter, the method comprises generating output reports of the one or more applications and sending the output reports from the updated primary cluster, wherein updated secondary cluster is under maintenance.

[0007] In an embodiment of the disclosure, updating the status of the one or more applications to the DFS further comprises updating the status of the one or more applications of each cluster to the DFS of the respective cluster and to the opposite cluster he updated DFS.

[0008] In another non-limiting embodiment of the disclosure, there is provided a system for managing multi data cluster failure. The system comprises one or more processors and a memory communicatively coupled to the one or more processors. The one or more processors are configured to schedule one or more applications on a primary cluster and a secondary cluster, monitoring a status of the one or more applications scheduled in one cluster by the other cluster, identify the status of the one or more applications of one cluster by the other cluster, update the status of the one or more applications associated with each cluster on a Distributed File System (DFS) of respective cluster, generate outcome reports regarding the one or more scheduled applications, send the generated outcome reports via the primary cluster to one or more users. The one or more processors further configured to automatically convert the secondary cluster to the primary cluster and vice-versa upon identifying the status of the primary cluster as under failure or maintenance during monitoring. The one or more processors configured to update the status of the one or more applications of the primary cluster to the DFS associated with the secondary cluster after a predefined threshold time. Thereafter, the one or more processors configured to generate output reports of the one or more applications from the updated primary cluster, and updated secondary cluster is under maintenance.

[0009] The foregoing summary is illustrative only and is not intended to be in any way limiting. In addition to the illustrative aspects, embodiments, and features described above, further aspects, embodiments, and features will become apparent by reference to the drawings and the following detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] The accompanying drawings, which are incorporated in and constitute a part of this disclosure, illustrate exemplary embodiments and, together with the description, serve to explain the disclosed principles. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The same numbers are used throughout the figures to reference like features and components. Some embodiments of system and/or methods in accordance with embodiments of the present subject matter are now described, by way of example only, and regarding the accompanying figures, in which:

[0011] **Fig.1** illustrates an exemplary environment for managing automated failover from a primary database server to a secondary database server, in accordance with some embodiments of the present disclosure;

[0012] **Fig.2** shows a detailed block diagram of the system for managing multi data cluster failure under normal scenario, in accordance with some embodiments of the present disclosure;

[0013] **Fig.3** shows a detailed block diagram of the system for managing multi data cluster failure under maintenance and / or application failure scenario, in accordance with some embodiments of the present disclosure;

[0014] **Fig.4** shows a detailed block diagram of the system for managing multi data cluster failure under post maintenance and / or application failure scenario, in accordance with some embodiments of the present disclosure;

[0015] **Fig.5** illustrates a flowchart showing a method for managing multi data cluster failure under normal scenario, in accordance with some embodiments of present disclosure; and

[0016] **Fig.6** illustrates a flowchart showing a method for managing multi data cluster failure under maintenance and/or application failure scenario, in accordance with some embodiments of present disclosure; and

[0017] **Fig.7** illustrates a block diagram of an exemplary computer system for managing multi data cluster failure in accordance with some embodiments of the present disclosure.

[0018] It should be appreciated by those skilled in the art that any block diagrams herein represent conceptual views of illustrative systems embodying the principles of the present subject matter. Similarly, it will be appreciated that any flow charts, flow diagrams, state transition diagrams, pseudo code, and the like represent various processes which may be substantially represented in computer readable medium and executed by a computer or processor, whether such computer or processor is explicitly shown.

<u>DESCRIPTION OF THE DISCLOSURE</u>

[0019] In the present document, the word “exemplary” is used herein to mean “serving as an example, instance, or illustration.” Any embodiment or implementation of the present subject matter described herein as “exemplary” is not necessarily to be construed as preferred or advantageous over other embodiments.

[0020] While the disclosure is susceptible to various modifications and alternative forms, specific embodiment thereof has been shown by way of example in the drawings and will be described in detail below. It should be understood, however, that it is not intended to limit the disclosure to the particular forms disclosed, but on the contrary, the disclosure is to cover all modifications, equivalents, and alternatives falling within the scope of the disclosure.

[0021] The terms “comprises”, “comprising”, or any other variations thereof, are intended to cover a non-exclusive inclusion, such that a setup, device, or method that comprises a list of components or steps does not include only those components or steps but may include other components or steps not expressly listed or inherent to such setup or device or method. In other words, one or more elements in a system or apparatus proceeded by “comprises… a” does not, without more constraints, preclude the existence of other elements or additional elements in the system or apparatus.

[0022] The terms "an embodiment", "embodiment", "embodiments", "the embodiment", "the embodiments", "one or more embodiments", "some embodiments", and "one embodiment" mean "one or more (but not all) embodiments of the invention(s)" unless expressly specified otherwise.

[0023] The terms "including", "comprising", “having” and variations thereof mean "including but not limited to", unless expressly specified otherwise.

[0024] The present disclosure provides a method and a system for managing multi data cluster failure. In the present disclosure, one or more applications are scheduled on a primary cluster and a secondary cluster, status of the one or more applications scheduled in one cluster are monitored by the other cluster. Herin, the status of the one or more applications associated with each cluster are updated on a Distributed File System (DFS) / file system of respective cluster. Further, outcome reports regarding the one or more scheduled applications are generated, and the generated outcome reports via the primary cluster are sent to one or more users. Further, the secondary cluster is converted automatically into the primary cluster and vice-versa upon identifying the status of the primary cluster as under failure or maintenance during monitoring. Furthermore, the status of the one or more applications of the primary cluster are updated to the DFS associated with the updated primary cluster after a predefined threshold time. Thereafter, output reports of the one or more applications scheduled in the primary cluster are generated and transmitted to users by the updated primary cluster (originally secondary cluster).

[0025] **Fig.1** illustrates an exemplary environment for managing automated failover from a primary database server to a secondary database server, in accordance with some embodiments of the present disclosure. The environment discloses a system for automated job failover 101 comprising one or more Data Management and Storage (DMS) clusters including a primary cluster 103 and a secondary cluster 105. The primary cluster 103 comprises a primary Distributed File System (DFS) cluster 109 with plurality of DFS nodes 117 and primary computing resources 107 with plurality of virtual machines (VMs) 115. The secondary cluster 105 comprises a secondary DFS cluster 113 with plurality of DFS nodes 121 and secondary computing resources 111 with plurality of virtual machines (VMs) 119. The system for automated job failover 101 provisions for automatically moving one or more applications from one data cluster to another data cluster during a data cluster failure in DMS to a computational infrastructure 100, which may be used by an enterprise such as a corporation, university, or government agency. Many different types of computational infrastructures are possible. Some examples include serving web pages, implementing e-commerce services and marketplaces, and providing computational resources for an enterprise's internal use. The computational infrastructure can include production environments, in addition to development or other environments. Fig.1 illustrates an exemplary computational infrastructure 100 according to one of the embodiments of the present disclosure.

[0026] In an embodiment, the computational infrastructure 100 includes both virtual machines (VMs) 115, 119 and physical machines (PMs). VMs can be based on different protocols. VMware, Microsoft Hyper-V, Microsoft Azure, GCP (Google Cloud Platform), Nutanix AHV, Linux KVM (Kernel-based Virtual Machine), and Xen are some examples. The physical machines can also use different operating systems running various applications. Microsoft Windows running Microsoft SQL or Oracle databases, and Linux running web servers are some examples. The VMs 115, 119 and the PMs are also collectively referred to as the machines.

[0027] Both of the DFS cluster including the primary cluster 103 and the secondary cluster 105 manage and store data for the computational infrastructure 100. This can include the states of machines, configuration settings of machines, network configuration of machines, and data stored on machines. Example of DMS services includes backup, recovery, replication, archival, and analytics services. A primary DFS cluster 109 enables creation and recovery of backup data for the applications of primary computational infrastructure. Derivative workloads (e.g., testing, development, and analytic workloads) may also use the primary DFS cluster 109 as a primary storage platform to read and/or modify past versions of data.

[0028] In one of the embodiments, to provide redundancy, two DMS clusters 103 and 105 are used. If the primary cluster 103 fails, the secondary cluster 105 can be used to provide DFS services to the computational infrastructure 100 with minimal interruption.

[0029] Each DFS cluster 109, 113 include multiple peer DFS nodes represented by 117, 121 that operate autonomously to collectively provide the DFS services, including managing and storing data. In an embodiment, the DFS nodes 117 and 121 include a software stack, processor, and data storage. DFS nodes 117, 121 can be implemented as physical machines and/or as virtual machines. The DFS nodes 117, 121 are interconnected with each other, for example, via cable, fiber, backplane, and/or network switch. An end user does not interact separately with each DFS node but interacts with the DFS nodes 117, 121 collectively as one entity, namely, the primary cluster 103 and the secondary cluster 105.

[0030] The DFS nodes 117, 121 are peers and each DFS node 117 and 121 includes the same functionality. The DFS clusters 109, 113 automatically configures the DFS nodes 117, 121 as new nodes are added, or existing nodes are dropped or fail.

[0031] Considering each component shown in Fig. 1 the virtual machine (VM) 115, 119 is a software simulation of a computing system. The VMs 115, 119 each provide a virtualized infrastructure that allows execution of operating systems as well as software applications such as a database application or a web server. A virtualization module resides on a physical host (i.e., a physical computing system) (not shown), and creates and manages the VMs. The virtualization module facilitates backups of VMs 115, 119 along with other virtual machine related tasks, such as cloning virtual machines, creating new virtual machines, monitoring the state of virtual machines, and moving virtual machines between physical hosts for load balancing purposes. In addition, the virtualization module provides an interface for other computing devices to interface with the virtualized infrastructure.

[0032] A physical machine is a physical computing system that allows execution of operating systems as well as software applications such as a database application or a web server. In the following example, an agent is installed on the physical machines to facilitate DFS services for the physical machines.

[0033] The components shown in Fig.1 may also include storage devices, which for example can be a hard disk drive (HDD), a magnetic tape drive, a solid-state drive (SSD), or a disk array (e.g., a storage area network (SAN) storage device, or a networked-attached storage (NAS) device). A storage device can be separated from or integrated with a physical machine.

[0034] The components in Fig.1 are interconnected with each other via networks, although many different types of networks could be used. In some cases, the relevant network uses standard communications technologies and/or protocols and can include the Internet, local area networks, and other types of private or public networks. The components can also be connected using custom and/or dedicated data communications technologies.

[0035] As shown, the primary DFS cluster 109 of the primary cluster 103 is coupled to the secondary DFS cluster 113 of the secondary cluster 105 and vice-versa. Therefore, the system for automated job failover 101 performs one or more actions to automatically enable the secondary cluster 105 to generate the output reports of the one or more applications of primary cluster 103 upon identifying the status of primary cluster 103 as failure or maintenance status. Further, system for automated job failover 101 updates the status of the one or more applications to the DFS after a predefined threshold time and generates the output reports from the updated primary cluster of the one or more applications.

[0036] After the primary cluster 103 is restored post failure and / or maintenance, the failback process can be initialized to failback from the secondary cluster 105 to the primary cluster 103. The failback process can be initialized according to a user instruction.

[0037] **Fig.2** illustrates an exemplary embodiment for managing multi data cluster failure under normal scenario, in accordance with the present disclosure. In one of the embodiments, a system for automated application failover 200 comprises a primary cluster 201, and a secondary cluster 203. The primary cluster 201 comprises a file system 205, an Automated Job Failover (AJF) primary module 207, and an output location 209. The secondary cluster 203 comprises a file system 211, an AJF secondary module 213, and an output location 215. The system for automated application failover 200 under normal scenario, is configured to schedule one or more applications to the primary cluster 201 and the secondary cluster 203, monitor a status of the one or more applications scheduled in one cluster by the other cluster. That is, the primary cluster 201 may monitor the status of the one or more applications of the secondary cluster 203 and simultaneously the secondary cluster 203 may monitor the status of the one or more applications of the primary cluster 201. Further, the system for automated application failover 200 is configured to update the status on the one or more application on respective file system, i.e., file system205 for primary and in file system 211 for secondary and generate output reports of the scheduled one or more applications and send the same to end user by the primary cluster 201.

[0038] **Fig.3** illustrates an exemplary embodiment for managing multi data cluster failure under failure or maintenance scenario, in accordance with the present disclosure.

[0039] In one of the embodiments, a system for automated application failover 300 comprises an updated secondary cluster (originally primary cluster) 301, and an updated primary cluster (originally secondary cluster) 303. The updated primary cluster 303 comprises a file system 311, an Automated Job Failover (AJF) primary module 313, and an output location 315. The updated secondary cluster 301 comprises a file system 305, an AJF stopped module 307, and an output location 309. The system for automated application failover 300 under failure or maintenance scenario, is configured to automatically convert the secondary cluster 303 to the primary cluster 301 and vice-versa upon identifying the status of primary cluster 301 as failure or maintenance during monitoring as discussed above under Figure.2. Herein, since the one or more applications of the primary cluster 301 is identified to be failed or under maintenance, the update to the file system 305 of the primary cluster 301 is stopped. Further, the system for automated application failover 300 under failure or maintenance scenario is configured to update the status of the one or more applications of the updated primary cluster 303 to the file system 311 after a predefined threshold. Thereafter, output reports are generated from the updated primary cluster 303 (originally secondary cluster) for the one or more applications of updated secondary cluster 301, wherein updated secondary cluster 301 (originally primary cluster) is under maintenance.

[0040] **Fig.4** illustrates an exemplary embodiment for managing multi data cluster failure post failure or maintenance scenario, in accordance with the present disclosure. In one of the embodiments, a system for automated application failover 400 post failure or maintenance scenario, comprises a primary cluster 401, and a secondary cluster 403. Post failure or maintenance scenario, an updated secondary cluster 401 (originally primary cluster) comprises a file system 405, an Automated Job Failover (AJF) secondary module 407, and an output location 409. While an updated primary cluster 403 (originally secondary cluster) comprises a file system 411, an AJF primary module 413, and an output location 415. The system for automated application failover 400 post failure or maintenance scenario is configured to restore the updated secondary cluster 401 (originally primary cluster) and initiate a failback from the updated primary cluster 403 (originally secondary cluster) to the updated secondary cluster 401 (originally primary cluster) as illustrated in Fig.4.

[0041] In an embodiment, each of the one or more modules may be a hardware unit which may be outside the memory and coupled with the system for automated job failover 101. As used herein, the term modules such as Automated Job Failover (AJF) modules including primary and secondary modules refers to an Application Specific Integrated Circuit (ASIC), an electronic circuit, a Field-Programmable Gate Arrays (FPGA), Programmable System-on-Chip (PSoC), a combinational logic circuit, and/or other suitable components that provide described functionality. The one or more modules when configured with the described functionality defined in the present disclosure will result in a novel hardware.

[0042] **Fig.5** shows an exemplary flow chart illustrating method steps for automated job failover, in accordance with some embodiments of the present disclosure. As illustrated in Figure 5, the method 500 may comprise one or more steps. The method 500 may be described in the general context of computer executable instructions. Generally, computer executable instructions can include routines, programs, objects, components, data structures, procedures, modules, and functions, which perform particular functions or implement particular abstract data types.

[0043] The order in which the method 500 is described is not intended to be construed as a limitation, and any number of the described method blocks can be combined in any order to implement the method. Additionally, individual blocks may be deleted from the methods without departing from the scope of the subject matter described herein. Furthermore, the method can be implemented in any suitable hardware, software, firmware, or combination thereof.

[0044] At step 501, the method schedules one or more applications on a primary cluster and a secondary cluster.

[0045] At step 503, the method monitors the status of the one or more applications scheduled in one data cluster by the other data cluster. The method further identifies the status of the one or more applications of one cluster by the other cluster.

[0046] At step 505, the method includes updating the status of the one or more applications associated with each cluster and the opposite cluster on the File system/Distributed File System (DFS) of respective cluster and the opposite cluster to generate report regarding the scheduled one or more applications.

[0047] At step 507, the method automatically converts the secondary cluster to the primary cluster and vice-versa upon identifying the status of primary cluster as failure or under maintenance during monitoring. Further, the method comprises updating the status of the one or more applications of the primary cluster to the DFS associated with the updated primary cluster after a predefined threshold time.

[0048] At step 509, the method generates the output reports of the one or more applications and sends to the user from the updated primary cluster (originally secondary cluster), wherein updated secondary cluster (originally primary cluster) is under maintenance.

[0049] **Fig.6** shows an exemplary flow chart illustrating method steps for automated job failover under various status including under normal scenario, under failure / maintenance scenario, post failure / maintenance scenario of the primary and the secondary clusters, in accordance with some embodiments of the present disclosure. As illustrated in Figure 6, method 600 may comprise one or more steps. The method 600 may be described in the general context of computer executable instructions. Generally, computer executable instructions can include routines, programs, objects, components, data structures, procedures, modules, and functions, which perform particular functions or implement particular abstract data types.

[0050] The order in which the method 600 is described is not intended to be construed as a limitation, and any number of the described method blocks can be combined in any order to implement the method. Additionally, individual blocks may be deleted from the methods without departing from the scope of the subject matter described herein. Furthermore, the method can be implemented in any suitable hardware, software, firmware, or combination thereof.

[0051] At step 601, the method monitors status of one or more applications scheduled in one cluster by the other cluster in a distributed environment with various configurations to support high availability (HA) clustering. The most commonly used HA clustering configurations are active-active and active passive. In active-active clustering configuration, the one or more servers actively implement a different database instance with each serving as a backup for each other. For example, in the active-active architecture, if the primary cluster is scheduled with one or more applications, then the primary cluster updates the status of the one or more applications in its own cluster / file system and eventually generates output reports regarding the one or more applications. Further, the primary cluster monitors the status of the one or more applications associated with the secondary clusters simultaneously to be updated.

[0052] At step 603, the status of the clusters is identified as under one of, a normal scenario, under failure / maintenance scenario or post failure / maintenance by the system for automated job failover during monitoring.

[0053] At step 605, the method 600 updates the status of one or more applications to the corresponding DFS cluster, upon identifying the status of the clusters is under normal scenario.

[0054] At step 607, the method monitors the status of the one or more applications of the opposite cluster to update.

[0055] Alternatively, at step 609, the status of the clusters is identified as under failure / maintenance or post failure / maintenance by the system for automated failover during monitoring.

[0056] At step 611, the method 600 converts the primary cluster to the secondary cluster and vice versa.

[0057] Alternatively, at step 613, the status of the clusters is identified as under post failure / maintenance scenario by the system for automated job failover during monitoring.

[0058] At step 615, the method 600 monitors the status of the one or more applications scheduled in one cluster by the opposite cluster. Under the post failure / maintenance status of the clusters, the secondary cluster acts as a new primary cluster, herein referred as updated primary cluster and the primary cluster becomes new secondary cluster, herein referred as updated secondary cluster, to ensure high availability of clusters in active-active configuration.

[0059] At step 617, the method 600 generates the output reports regarding the one or more scheduled applications to report to the end users.

[0060] At step 629, the method may provide the generated output reports to the user.

COMPUTER SYSTEM

[0061] **Fig.7** illustrates a block diagram of an exemplary computer system 700 for implementing embodiments consistent with the present disclosure. In an embodiment, the computer system 700 may be used to implement the system for automated job failover 101. Thus, the computer system 700 may be used for automated job failover 101. The computer system 700 may communicate with the one or more DFS clusters 103 and 105 over a communication network 709. The computer system 700 may comprise a Central Processing Unit 702 (also referred as “CPU” or “processor”). The processor 702 may comprise at least one data processor. The processor 702 may include specialized processing units such as integrated system (bus) controllers, memory management control units, floating point units, graphics processing units, digital signal processing units, etc.

[0062] The processor 702 may be disposed in communication with one or more input/output (I/O) devices (not shown) via I/O interface 701. The I/O interface 701 may employ communication protocols/methods such as, without limitation, audio, analog, digital, monoaural, RCA, stereo, IEEE (Institute of Electrical and Electronics Engineers) -1394, serial bus, universal serial bus (USB), infrared, PS/2, BNC, coaxial, component, composite, digital visual interface (DVI), high-definition multimedia interface (HDMI), Radio Frequency (RF) antennas, S-Video, VGA, IEEE 802.n /b/g/n/x, Bluetooth, cellular (e.g., code-division multiple access (CDMA), high-speed packet access (HSPA+), global system for mobile communications (GSM), long-term evolution (LTE), WiMax, or the like), etc.

[0063] Using the I/O interface 701, the computer system 700 may communicate with one or more I/O devices. For example, the input device 710 may be an antenna, keyboard, mouse, joystick, (infrared) remote control, camera, card reader, fax machine, dongle, biometric reader, microphone, touch screen, touchpad, trackball, stylus, scanner, storage device, transceiver, video device/source, sensors, etc. The output device 711 may be a printer, fax machine, video display (e.g., cathode ray tube (CRT), liquid crystal display (LCD), light-emitting diode (LED), plasma, Plasma display panel (PDP), Organic light-emitting diode display (OLED) or the like), audio speaker, etc.

[0064] The processor 702 may be disposed in communication with the communication network 709 via a network interface 703. The network interface 703 may communicate with the communication network 709. The network interface 703 may employ connection protocols including, without limitation, direct connect, Ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, etc. The communication network 709 may include, without limitation, a direct interconnection, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, etc. The network interface 703 may employ connection protocols include, but not limited to, direct connect, Ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, Bluetooth mesh, Zigbee, etc.

[0065] The communication network 709 includes, but is not limited to, a direct interconnection, an e-commerce network, a peer to peer (P2P) network, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, Wi-Fi, and such. The first network and the second network may either be a dedicated network or a shared network, which represents an association of the different types of networks that use a variety of protocols, for example, Hypertext Transfer Protocol (HTTP), Transmission

Control Protocol/Internet Protocol (TCP/IP), Wireless Application Protocol (WAP), etc., to communicate with each other. Further, the first network and the second network may include a variety of network devices, including routers, bridges, servers, computing devices, storage devices, etc.

[0066] In some embodiments, the processor 702 may be disposed in communication with a memory 705 (e.g., RAM, ROM, etc. not shown in Figure 7) via a storage interface 704. The storage interface 704 may connect to memory 705 including, without limitation, memory drives, removable disc drives, etc., employing connection protocols such as serial advanced technology attachment (SATA), Integrated Drive Electronics (IDE), IEEE-1394, Universal Serial Bus (USB), fiber channel, Small Computer Systems Interface (SCSI), etc. The memory drives may further include a drum, magnetic disc drive, magneto-optical drive, optical drive, Redundant Array of Independent Discs (RAID), solid-state memory devices, solid-state drives, etc.

[0067] The memory 705 may store a collection of program or database components, including, without limitation, user interface 706, an operating system 707, web browser 708 etc. In some embodiments, computer system 700 may store user/application data, such as, the data, variables, records, etc., as described in this disclosure. Such databases may be implemented as fault-tolerant, relational, scalable, secure databases such as Oracle ® or Sybase®.

[0068] The operating system 707 may facilitate resource management and operation of the computer system 700. Examples of operating systems include, without limitation, APPLE MACINTOSH^{R} OS X, UNIX^{R}, UNIX-like system distributions (E.G., BERKELEY SOFTWARE DISTRIBUTION^{TM} (BSD), FREEBSD^{TM}, NETBSD^{TM}, OPENBSD^{TM}, etc.), LINUX DISTRIBUTIONS^{TM} (E.G., RED HAT^{TM}, UBUNTU^{TM}, KUBUNTU^{TM}, etc.), IBM^{TM} OS/2, MICROSOFT^{TM} WINDOWS^{TM} (XP^{TM}, VISTA^{TM}/7/8, 10 etc.), APPLE^{R} IOS^{TM}, GOOGLE^{R} ANDROID^{TM}, BLACKBERRY^{R} OS, or the like.

[0069] In some embodiments, the computer system 700 may implement the web browser 708 stored program component. The web browser 708 may be a hypertext viewing application, for example MICROSOFT^{R} INTERNET EXPLORER^{TM}, GOOGLE^{R} CHROME^{TM0}, MOZILLA^{R} FIREFOX^{TM}, APPLE^{R} SAFARI^{TM}, etc. Secure web browsing may be provided using Secure Hypertext Transport Protocol (HTTPS), Secure Sockets Layer (SSL), Transport Layer Security (TLS), etc. Web browsers 708 may utilize facilities such as AJAX^{TM}, DHTML^{TM}, ADOBE^{R} FLASH^{TM}, JAVASCRIPT^{TM}, JAVA^{TM}, Application Programming Interfaces (APIs), etc. In some embodiments, the computer system 700 may implement a mail server (not shown in Figure) stored program component. The mail server may be an Internet mail server such as Microsoft Exchange, or the like. The mail server may utilize facilities such as ASP^{TM}, ACTIVEX^{TM}, ANSI^{TM} C++/C#, MICROSOFT^{R},.NET^{TM}, CGI SCRIPTS^{TM}, JAVA^{TM}, JAVASCRIPT^{TM}, PERL^{TM}, PHP^{TM}, PYTHON^{TM}, WEBOBJECTS^{TM}, etc. The mail server may utilize communication protocols such as Internet Message Access Protocol (IMAP), Messaging Application Programming Interface (MAPI), MICROSOFT^{R} exchange, Post Office Protocol (POP), Simple Mail Transfer Protocol (SMTP), or the like. In some embodiments, the computer system 700 may implement a mail client stored program component. The mail client (not shown in Figure) may be a mail viewing application, such as APPLE^{R} MAIL^{TM}, MICROSOFT^{R} ENTOURAGE^{TM}, MICROSOFT^{R} OUTLOOK^{TM}, MOZILLA^{R} THUNDERBIRD^{TM}, etc.

[0070] Furthermore, one or more computer-readable storage media may be utilized in implementing embodiments consistent with the present disclosure. A computer-readable storage medium refers to any type of physical memory on which information or data readable by a processor may be stored. Thus, a computer-readable storage medium may store instructions for execution by one or more processors, including instructions for causing the processor(s) to perform steps or stages consistent with the embodiments described herein. The term “computer-readable medium” should be understood to include tangible items and exclude carrier waves and transient signals, i.e., be non-transitory. Examples include Random Access Memory (RAM), Read-Only Memory (ROM), volatile memory, non-volatile memory, hard drives, Compact Disc Read-Only Memory (CD ROMs), Digital Video Disc (DVDs), flash drives, disks, and any other known physical storage media.

[0071] An embodiment of the present disclosure reduces the manual effort and intervention of different teams in case of failure or maintenance of data clusters to start different applications from different datacenters.

[0072] The enumerated listing of items does not imply that any or all of the items are mutually exclusive, unless expressly specified otherwise. The terms "a", "an" and "the" mean "one or more", unless expressly specified otherwise.

[0073] A description of an embodiment with several components in communication with each other does not imply that all such components are required. On the contrary a variety of optional components are described to illustrate the wide variety of possible embodiments of the invention.

[0074] When a single device or article is described herein, it will be readily apparent that more than one device/article (whether or not they cooperate) may be used in place of a single device/article. Similarly, where more than one device or article is described herein (whether or not they cooperate), it will be readily apparent that a single device/article may be used in place of the more than one device or article, or a different number of devices/articles may be used instead of the shown number of devices or programs. The functionality and/or the features of a device may be alternatively embodied by one or more other devices which are not explicitly described as having such functionality/features. Thus, other embodiments of the invention need not include the device itself.

[0075] The illustrated operations of Figure 4-5 shows certain events occurring in a certain order. In alternative embodiments, certain operations may be performed in a different order, modified, or removed. Moreover, steps may be added to the above-described logic and still conform to the described embodiments. Further, operations described herein may occur sequentially or certain operations may be processed in parallel. Yet further, operations may be performed by a single processing unit or by distributed processing units.

[0076] Finally, the language used in the specification has been principally selected for readability and instructional purposes, and it may not have been selected to delineate or circumscribe the inventive subject matter. It is therefore intended that the scope of the invention be limited not by this detailed description, but rather by any claims that issue on an application based here on. Accordingly, the disclosure of the embodiments of the invention is intended to be illustrative, but not limiting, of the scope of the invention.

[0077] While various aspects and embodiments have been disclosed herein, other aspects and embodiments will be apparent to those skilled in the art. The various aspects and embodiments disclosed herein are for purposes of illustration and are not intended to be limiting.

METHOD AND DEVICE FOR AUTOMATED JOB FAILOVER

ABSTRACT

In an embodiment of the present disclosure, there is provided a method of managing multi data cluster failure. The method comprises scheduling one or more applications on a primary cluster and a secondary cluster, monitoring a status of the one or more applications scheduled in one cluster by the other data cluster, identifying the status of the applications of one cluster by the other cluster, updating the status of the one or more applications associated with each cluster on the distributed file system (DFS) of respective cluster, and generating output reports of the scheduled one or more applications. The method automatically converts the secondary data cluster to the primary cluster and vice-versa upon identifying the status of primary cluster as under failure or under maintenance during monitoring.

1/7

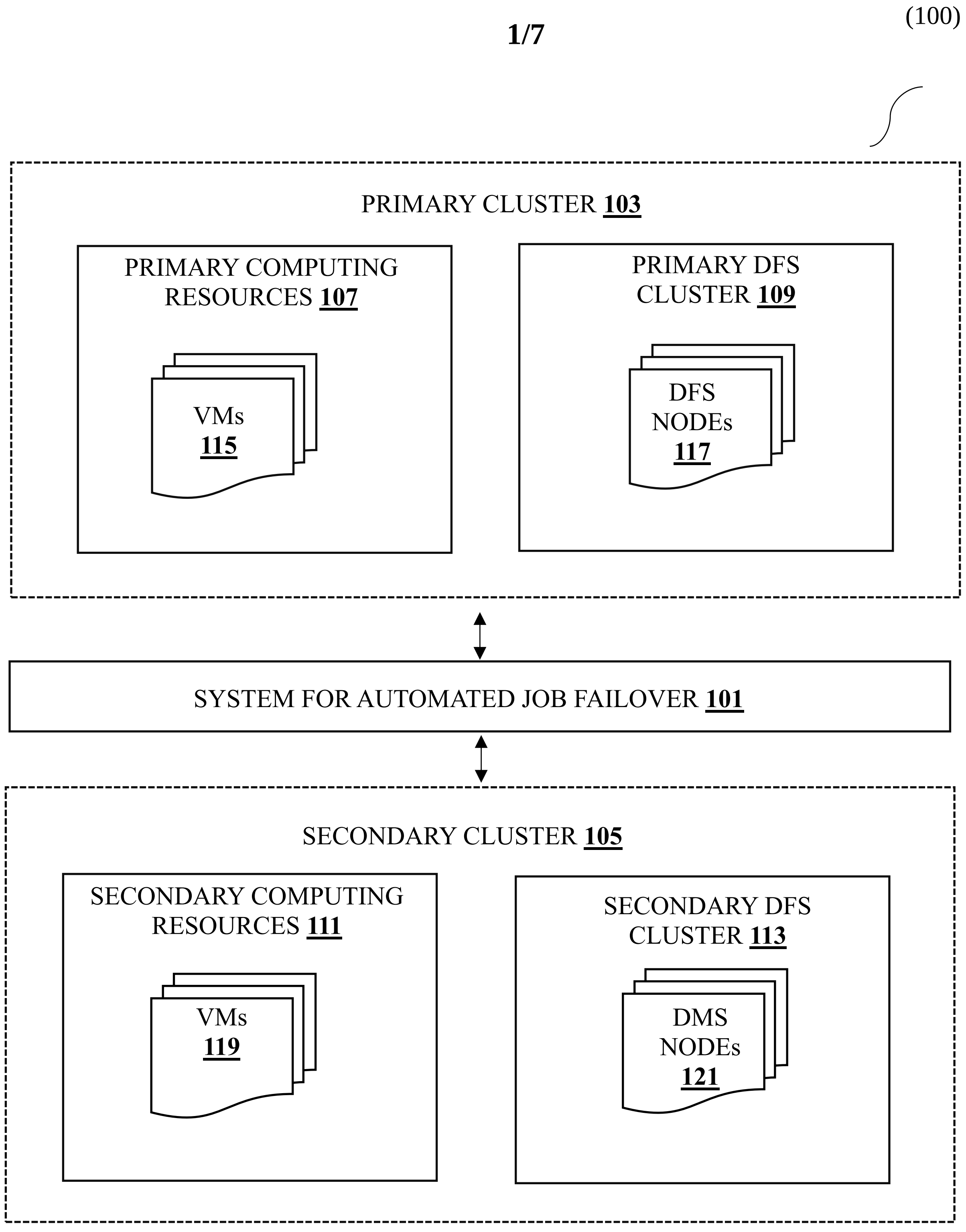
(100)
PRIMARY CLUSTER 103
PRIMARY COMPUTING RESOURCES 107
VMs 115
PRIMARY DFS CLUSTER 109
DFS NODEs 117
SYSTEM FOR AUTOMATED JOB FAILOVER 101
SECONDARY CLUSTER 105
SECONDARY COMPUTING RESOURCES 111
VMs 119
SECONDARY DFS CLUSTER 113
DMS NODEs 121

Fig. 1

2/7

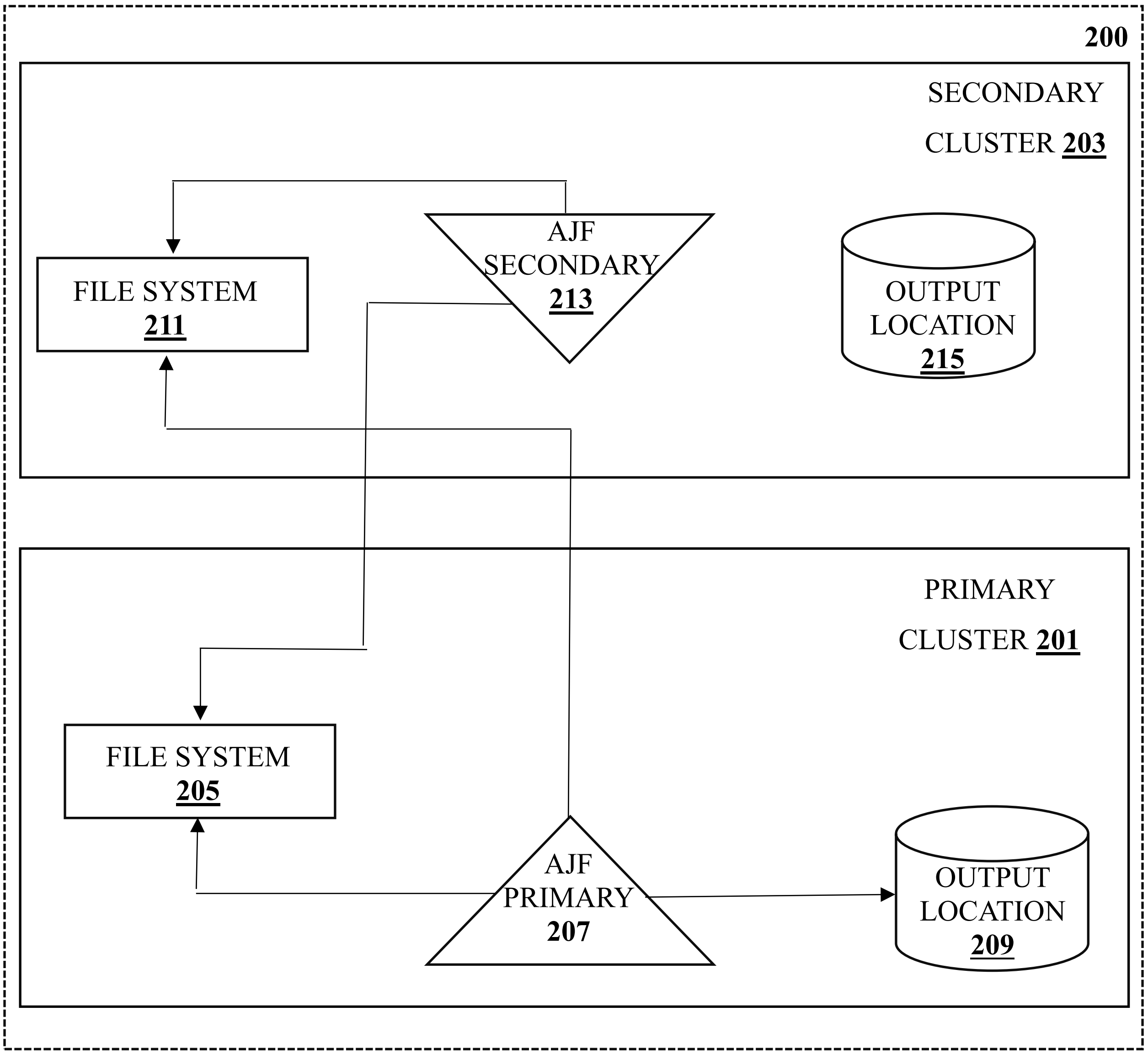
200
SECONDARY CLUSTER 203
FILE SYSTEM 211
AJF SECONDARY 213
OUTPUT LOCATION 215
PRIMARY CLUSTER 201
FILE SYSTEM 205
AJF PRIMARY 207
OUTPUT LOCATION 209

Fig. 2

3/7

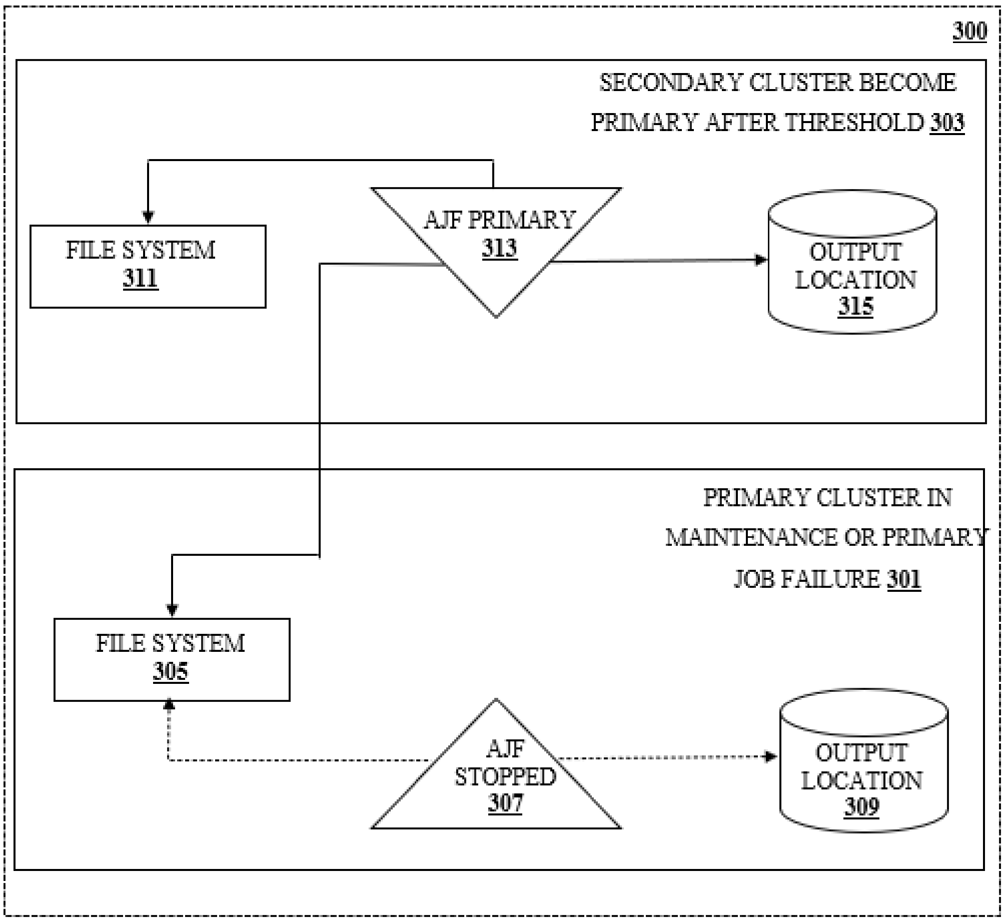
300
SECONDARY CLUSTER BECOME PRIMARY AFTER THRESHOLD 303
FILE SYSTEM 311
AJF PRIMARY 313
OUTPUT LOCATION 315
PRIMARY CLUSTER IN MAINTENANCE OR PRIMARY JOB FAILURE 301
FILE SYSTEM 305
AJF STOPPED 307
OUTPUT LOCATION 309

Fig. 3

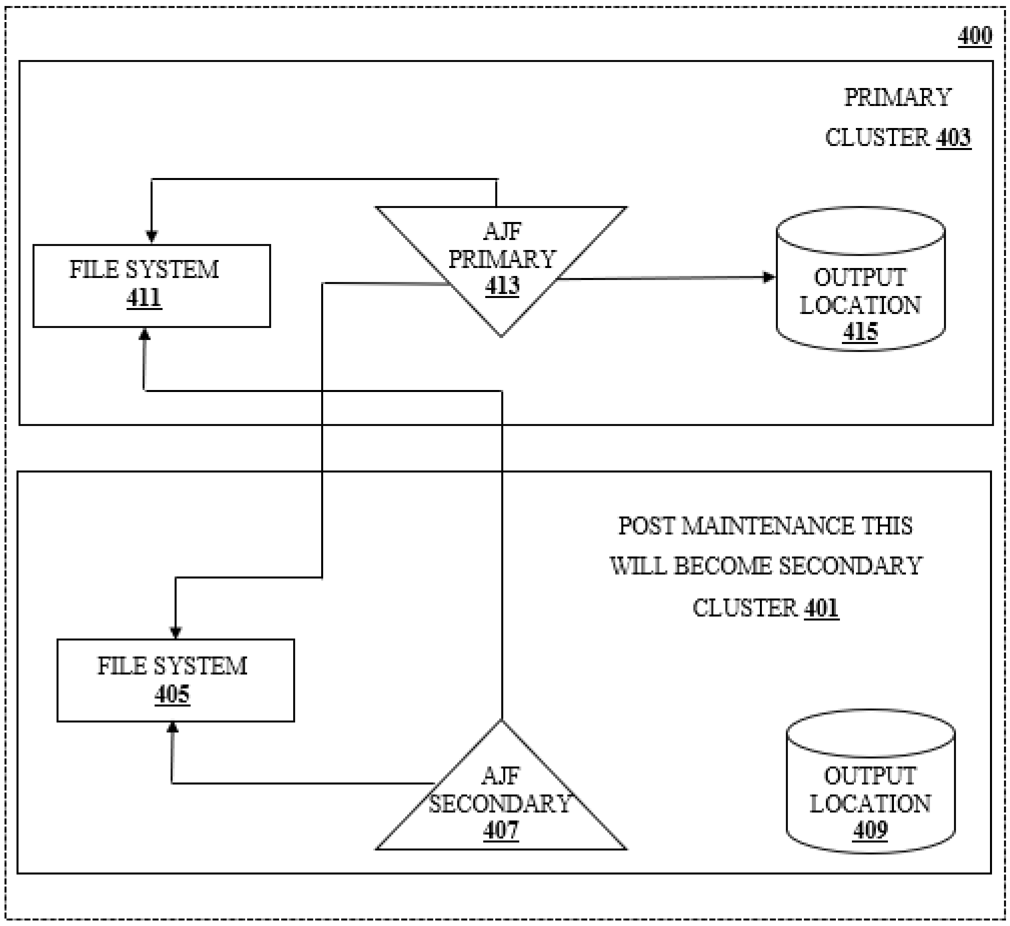
400
PRIMARY CLUSTER 403
FILE SYSTEM 411
AJF PRIMARY 413
OUTPUT LOCATION 415
POST MAINTENANCE THIS WILL BECOME SECONDARY CLUSTER 401
FILE SYSTEM 405
AJF SECONDARY 407
OUTPUT LOCATION 409

Fig. 4

5/7

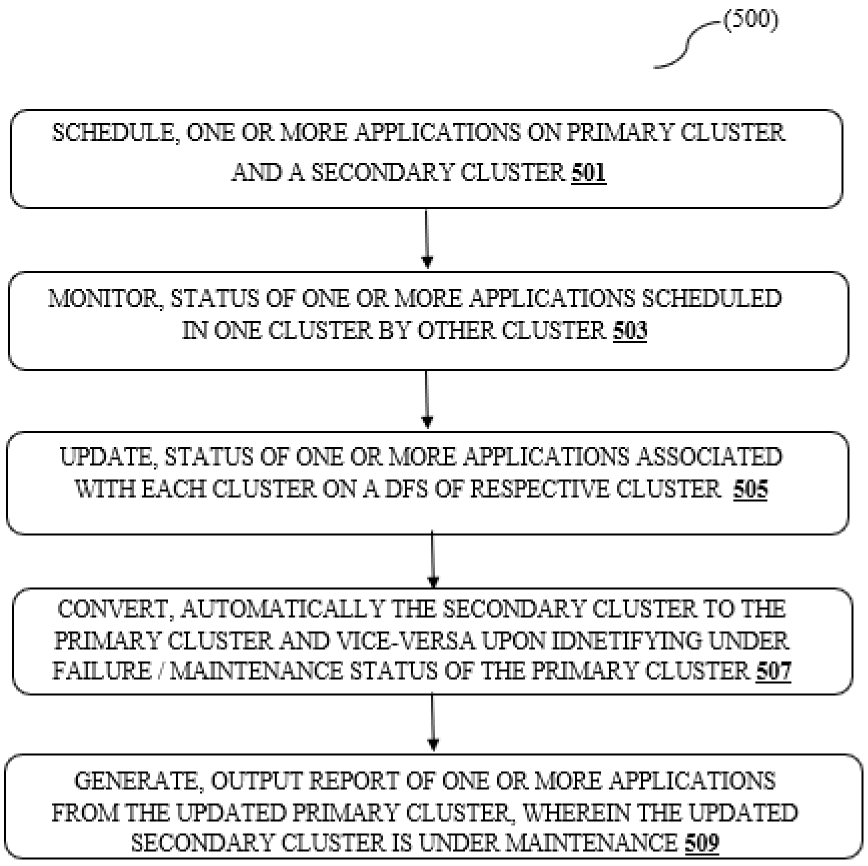
(500)
SCHEDULE, ONE OR MORE APPLICATIONS ON PRIMARY CLUSTER AND A SECONDARY CLUSTER 501
MONITOR, STATUS OF ONE OR MORE APPLICATIONS SCHEDULED IN ONE CLUSTER BY OTHER CLUSTER 503
UPDATE, STATUS OF ONE OR MORE APPLICATIONS ASSOCIATED WITH EACH CLUSTER ON A DFS OF RESPECTIVE CLUSTER 505
CONVERT, AUTOMATICALLY THE SECONDARY CLUSTER TO THE PRIMARY CLUSTER AND VICE-VERSA UPON IDNETIFYING UNDER FAILURE / MAINTENANCE STATUS OF THE PRIMARY CLUSTER 507
GENERATE, OUTPUT REPORT OF ONE OR MORE APPLICATIONS FROM THE UPDATED PRIMARY CLUSTER, WHEREIN THE UPDATED SECONDARY CLUSTER IS UNDER MAINTENANCE 509

Fig. 5

6/7

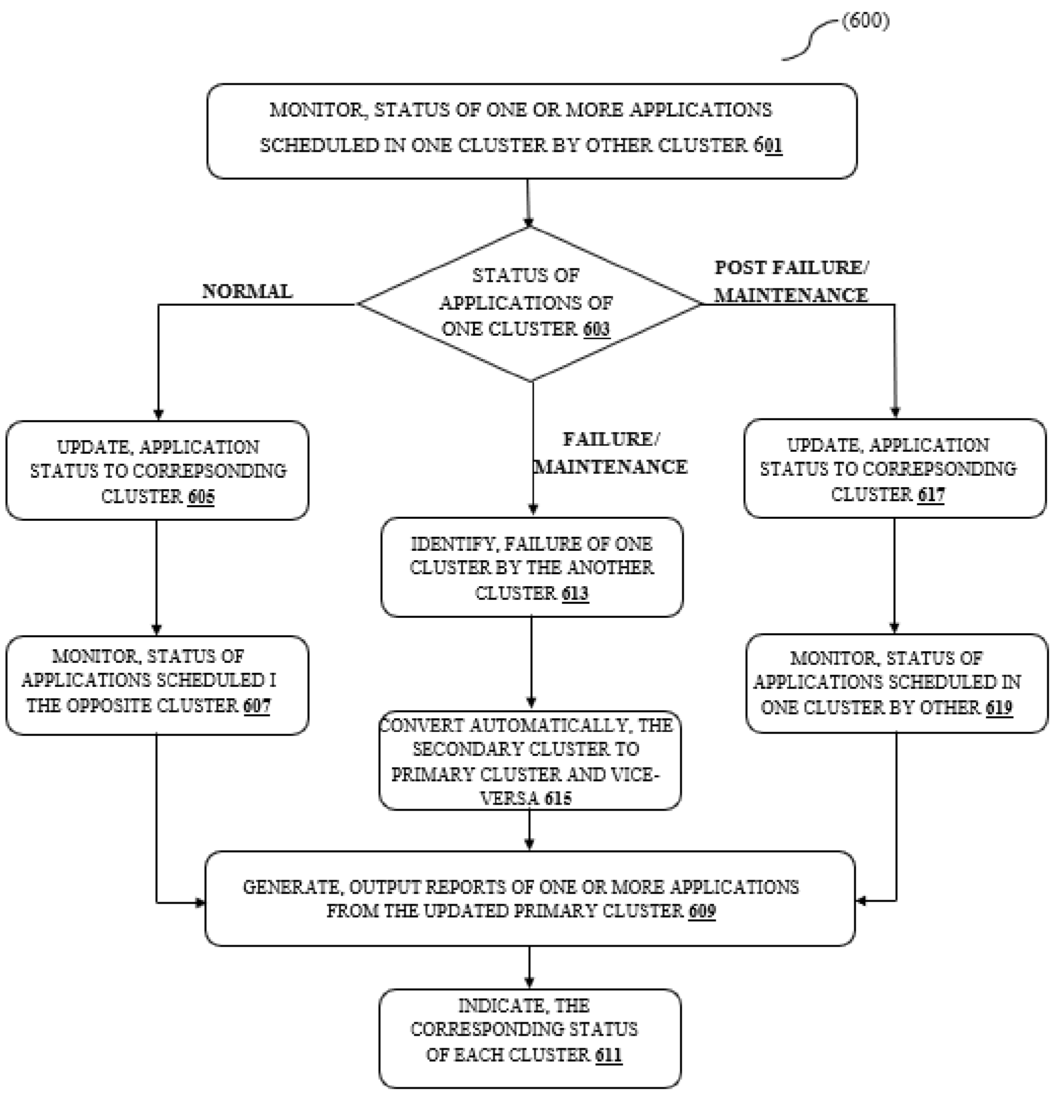
(600)
MONITOR, STATUS OF ONE OR MORE APPLICATIONS SCHEDULED IN ONE CLUSTER BY OTHER CLUSTER 601
STATUS OF APPLICATIONS OF ONE CLUSTER 603
NORMAL
POST FAILURE/ MAINTENANCE
FAILURE/ MAINTENANCE
UPDATE, APPLICATION STATUS TO CORREPSONDING CLUSTER 605
IDENTIFY, FAILURE OF ONE CLUSTER BY THE ANOTHER CLUSTER 613
UPDATE, APPLICATION STATUS TO CORREPSONDING CLUSTER 617
MONITOR, STATUS OF APPLICATIONS SCHEDULED I THE OPPOSITE CLUSTER 607
CONVERT AUTOMATICALLY, THE SECONDARY CLUSTER TO PRIMARY CLUSTER AND VICE-VERSA 615
MONITOR, STATUS OF APPLICATIONS SCHEDULED IN ONE CLUSTER BY OTHER 619
GENERATE, OUTPUT REPORTS OF ONE OR MORE APPLICATIONS FROM THE UPDATED PRIMARY CLUSTER 609
INDICATE, THE CORRESPONDING STATUS OF EACH CLUSTER 611

Fig. 6

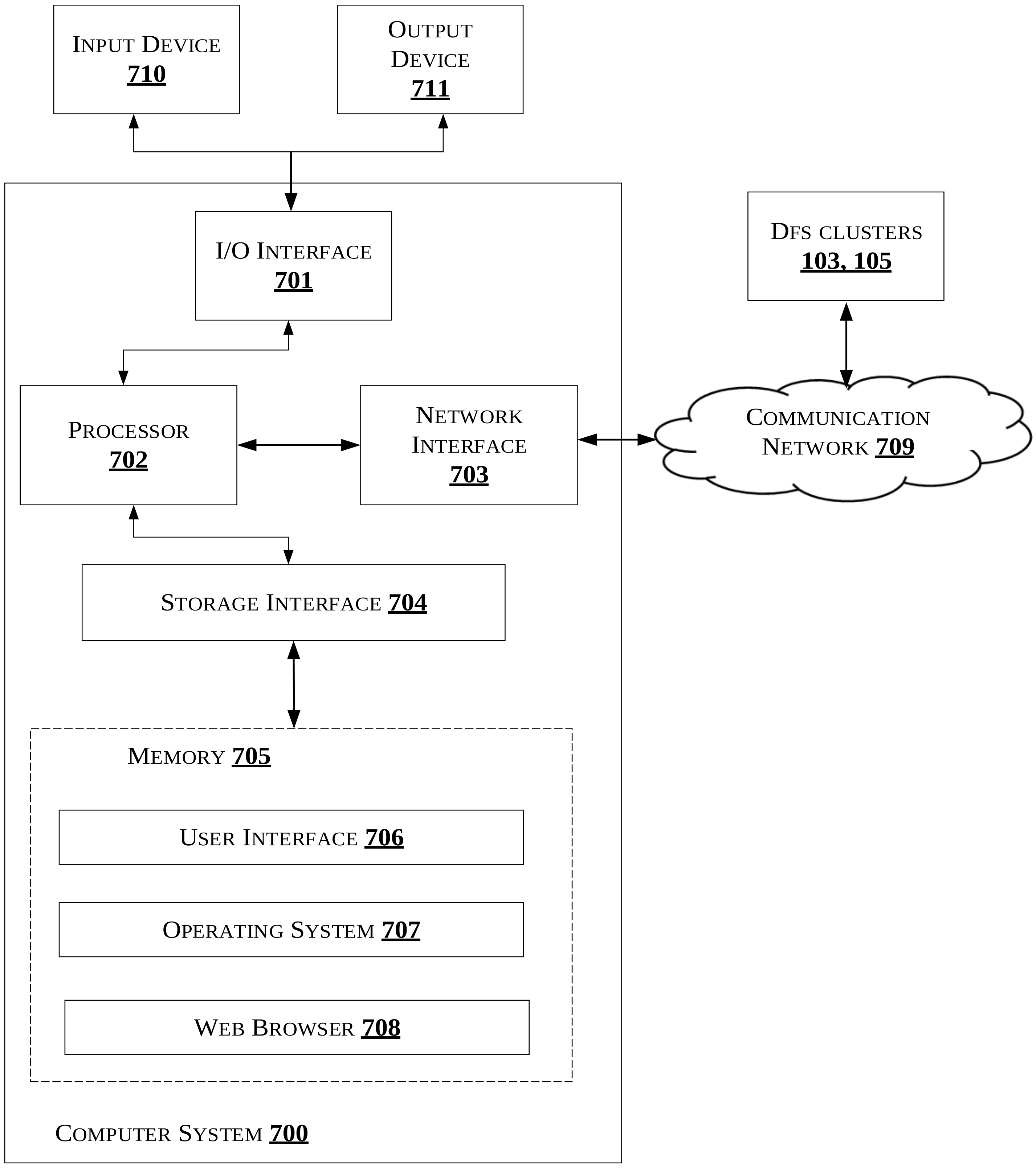
INPUT DEVICE 710
OUTPUT DEVICE 711
I/O INTERFACE 701
DFS CLUSTERS 103, 105
PROCESSOR 702
NETWORK INTERFACE 703
COMMUNICATION NETWORK 709
STORAGE INTERFACE 704
MEMORY 705
USER INTERFACE 706
OPERATING SYSTEM 707
WEB BROWSER 708
COMPUTER SYSTEM 700

Fig. 7