

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
5 December 2002 (05.12.2002)

PCT

(10) International Publication Number
WO 02/097621 A2

(51) International Patent Classification⁷: **G06F 9/46**

(21) International Application Number: PCT/EP02/04839

(22) International Filing Date: 3 May 2002 (03.05.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2,349,086 30 May 2001 (30.05.2001) CA

(71) Applicant (for all designated States except US): **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, NY 10504 (US).

(71) Applicant (for LU only): **IBM DEUTSCHLAND GMBH** [DE/DE]; Pascalstrasse 100, 70569 Stuttgart (DE).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **DEBOER, Timothy, G.** [CA/CA]; 11 Willimart Court, Toronto, Ontario

M2M 1G5 (CA). **FRANCIS, Timothy, M.** [CA/CA]; 260 Davis Drive, Apt. 403, Newmarket, Ontario L3Y 7C3 (CA). **WOSNICK, Sheldon, B.** [CA/CA]; 1 Lillooet Crescent, Richmond Hill, Ontario L4C 5A6 (CA). **YUEN, Siu, Chung** [CA/CA]; 55 Brimwood Blvd., #10, Scarborough, Ontario M1V 1E4 (CA).

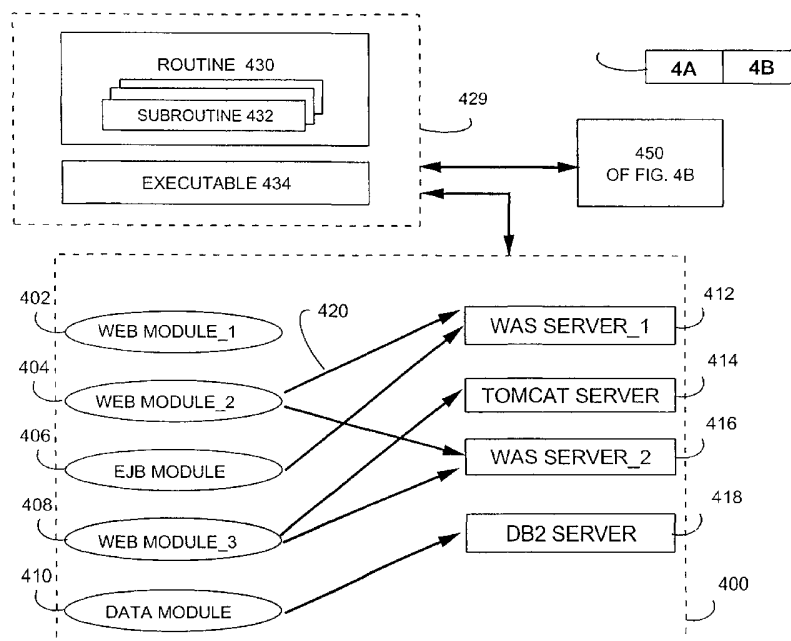
(74) Agent: **TEUFEL, Fritz**; IBM Deutschland GmbH, Intellectual Property, Pascalstr. 100, 70548 Stuttgart (DE).

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

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent

[Continued on next page]

(54) Title: SELECTION AND CONFIGURATION OF SERVERS



(57) Abstract: The invention provides a mechanism for convenient management for configuring and selecting servers for running modules. The invention provides a method for managing a plurality of servers including receiving a request to configure a suitable server being suitable for running a module, locating an identifier of the suitable server from a table comprising identifiers of suitable servers, configuring the suitable server to run the module.



(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

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.

Published:

— *without international search report and to be republished upon receipt of that report*

D E S C R I P T I O N

Selection and Configuration of Servers

Field of the Invention

This invention relates to computer programmed code, and more specifically to a method and a system for configuring and selecting servers to run software modules in a computer software environment.

Background of the Invention

Software development environments for developing software, such as software for Java or HTML, are well known. An example of a software development environment is J2EE™ (Java 2 Platform Enterprise Edition) which is available from Sun Microsystems. J2EE includes a set of services, application programming interfaces (APIs) and protocols for developing multi-tiered, web-based software.

A server is a network-connected computer having server software for managing resources. For example, a file server includes a computer, server and a data storage device for storing files so that users can store files on the file server over a network. A print server includes a computer and server software for managing a network of printers. A network server includes a computer and server software for managing network traffic. A database server is used for processing database queries. Servers are dedicated to perform their predetermined server tasks. For some operating systems, a single computer can execute or run several servers or programs simultaneously. A computer can be used to execute or run several servers of differing types and perhaps simultaneously. A server in this case could refer to the software that performs tasks rather

- 2 -

than the combination of the computer and the server. For the sake of simplicity, server software will hereinafter be called "server".

Deployment is a process where servers, in a software development environment, are configured to run software modules hereinafter called "modules". The task of configuring the servers may be performed by the members of the software development team. There are several types of modules, such as EJB (Enterprise Java Beans), WEB (Internet WEB information), and application clients. A module can include computer programmed files having computer programmed code or software, computer executable code, or computer readable data. The computer files can be organized in a directory having sub-directories.

Referring to Fig. 3, block 300 depicts a prior art software development environment that resides in the memory of a computer. It will be appreciated that environment 300 can reside in the memories of various network-connected computers, and environment 300 is not merely limited to residing on a single computer. Blocks 302, 304, 306, and 308 depict various modules. Modules 302 and 308 represent a WEB type module. Module 304 represents an EJB type module. Module 306 represents a data type module. Blocks 310 and 314 represent various types of servers of environment 300. Servers 310 and 314 each have a configuration file 312 and 316 respectively. Server 310 represents a Websphere™ Application Server available from IBM Corporation. Server 312 represents a DB2™ (database) Server also available from IBM Corporation. Arrows 318, 320, and 322 indicate that a server has been deployed or configured to run a particular module or modules. Arrows 318 and 320 indicate that server 310 has been configured or deployed to execute or run modules 302 and 304 respectively. Arrow 310 indicates that server 314 has been

- 3 -

deployed or configured to run module 306. A server of development environment 300 has yet to be configured to execute module 310. It can also be said that module 310 has yet to be deployed to run on a server. Configuration files 312 and 316 are used to indicate which modules can be run by servers 310 and 314 respectively. If a software developer sends a request to a server that is configured to execute a module, then the configured server will execute the module. However, if a server has not yet been configured to execute the module, then the module is not executed by the server. To have an appropriate server run module 308, a software developer must first change the configuration file of the appropriate server. Therefore, every time a new module is to be run by a server, the configuration file of a suitable server must be modified. During the software development cycle, many modules are created and require testing. It is inconvenient to modify configuration files of various appropriate servers every time a new module is created and needs to be tested, especially when numerous software developers are involved in developing various modules.

The prior art provides an inadequate mechanism for configuring and selecting servers in a software development environment. Software developers are required to spend valuable time performing the task of configuring appropriate servers to execute or run newly developed modules before the modules can be executed or run by the appropriate servers, which can lead to confusion when numerous developers are involved. The prior art mechanism becomes problematic and inconvenient when the development environment includes a multitude of types of servers, and the software developers are working to meet a deadline for developing and delivering fully tested modules.

Summary of the Invention

- 4 -

An object of the invention is to provide a mechanism for configuring and selecting suitable servers being suitable for running various modules.

The invention provides a method for configuration and selection of suitable servers being suitable for running various modules. The method can be tangibly embodied in an executable program which can configure and select various servers of a software environment. Suitable servers can be identified from a lookup table. Various types of lookup tables can be implemented and used in a suitable combination with each other, such as a backup table, a preferred table, and a priority table.

The backup table includes identifiers for identifying types of suitable servers that correspond to types of modules. If a suitable server cannot be readily located from the software environment, then a type of a suitable server can be located or identified from the backup table and a suitable server can subsequently be created to run the module. The backup table can contain predetermined identification of various types of servers and corresponding types of modules that the servers can execute, or can contain no identifiable servers in which the user must insert the required identification in the default table.

The preferred table includes identifiers for identifying names of suitable servers that correspond to names of modules. The server selection logic can be set up so that to locate or identify a suitable server, the preferred table is examined before examining the backup table. If the preferred table does not provide an identifier of a suitable server, then the backup table is used. The preferred table can contain predetermined identification of various identifiers or names of servers and corresponding identifiers or names of modules

- 5 -

that the servers can execute. Alternatively, the preferred table can contain no identifiers of servers and identifiers of corresponding modules in which the user must insert the required identifiers in the default table.

The priority table can be used as a list which includes identifiers of previously created suitable servers in which the software developers can rank or prioritize the order of the suitable servers. If all three types of tables are used, the server selection logic can be set up so that to locate or identify a suitable server, the priority table is examined before examining the preferred table or the backup table. If the priority table does not contain the identifier of a suitable server, then the preferred table is examined before examining the backup table. If the preferred table does not contain the identifier of a suitable server, then the backup table is examined to locate the identifier of a suitable server. The priority table can contain predetermined identifiers or names of various servers, or can contain no identified servers in which the user must insert the required identification in the priority table.

According to a first aspect of the present invention, there is provided a method for managing a plurality of servers including receiving a request to configure a suitable server being suitable for running a module, locating an identifier of the suitable server from a table including identifiers of suitable servers, and configuring the suitable server to run the module.

According to a second aspect of the present invention, there is provided a computer program product for use in a computer system operatively coupled to a computer readable memory, the computer program product including a computer-readable data storage medium tangibly embodying computer readable program

- 6 -

code for directing the computer to for manage a plurality of servers, the code including code for instructing the computer system to receive a request to configure a suitable server being suitable for running a module, code for instructing the computer system to locate an identifier of the suitable server from a table including identifiers of suitable servers, and code for instructing the computer system to configure the suitable server to run the module.

According to a third aspect of the present invention, there is provided a computer system operatively coupled to a computer readable memory, the computer system for managing a plurality of servers, the computer system including means for receiving a request to configure a suitable server being suitable for running a module, means for locating an identifier of the suitable server from a table including identifiers of suitable servers, and means for configuring the suitable server to run the module.

According to a fourth aspect of the invention, there is provided a set of data for configuring and selecting a server to run a module, the set of data including a table for containing an identifier of a suitable server being suitable for running the module.

A better understanding of these and other aspects of the invention can be obtained with reference to the following drawings and description of the preferred embodiments of the invention.

Drawings of the Invention

The following figures are embodiments of the invention, in which:

- 7 -

Fig. 1 depicts a computer system;

Fig. 2 depicts a structure of the computer of Fig. 1;

Fig. 3 depicts a prior art software development environment;

Fig. 4 depicts a deployer software routine and deployment data for setting up servers to deploy modules in a software development environment; and

Figs. 5 depicts a flowchart for setting up servers to deploy modules.

Detailed Description of the Preferred Embodiments of the Invention

Referring to Fig. 1, there is depicted conventional computer network suitable for the preferred embodiments of the present invention. A conventional computer system 100 operationally coupled to a networked computer 118 via suitable network connections 112, 116 and network 114. Network 114 is a conventional network such as a local area network, wide area network, intranets, Internet, and the like, or a convenient combination thereof. The network 114 provides a convenient mechanism for transporting data to and from the computer system 100. It will be appreciated that another embodiment of the invention can include a computer system 100 not being connected to the network 114 via network connection 112, provided the data is entered directly to the memory of computer system 100 via a keyboard/mouse 106 or via a removable computer readable media, such as a floppy disk 110.

For convenience, aspects of the preferred embodiments of the present invention can be distributed amongst various networked computers interacting with a computer system 100 via network

- 8 -

114 or a combination of networks. However, for convenient illustration of the preferred embodiments of the invention, the flowchart of Fig. 5 will be generally implemented in computer system 100.

Computer system 100 includes a computer 104 which communicates with various output devices such as a display terminal 102 or a printer 108, with the network 114, and with various input devices, such as keyboard/mouse 106, or a disk 110. Other devices can include various computer peripheral devices, such as a scanner, CD-ROM drives, and the like.

Referring to Fig. 2, there is depicted a structure of computer 104 suitable for the preferred embodiments of the present invention. Computer 104 includes a bus 206 that operationally interconnects various sub systems or components of the computer 104, such as a central processing unit (CPU) 202, a memory 204, a network interface (I/F) 208, and an input/output interface 210.

CPU 202 is a commercially available central processing unit suitable for operations described herein. Other variations of CPU 202 can include a plurality of CPUs. Suitable support circuits or components can be included for adapting the CPU 202 for optimum performance with the subsystems of computer 104.

Input/output (I/O) interface 210 enables communication between various subsystems of computer 104 and various input/output devices, such as keyboard/mouse 106. Input/output interface includes a video card for operational interfacing with display unit 102, and preferably a disk drive unit for reading suitable removable computer-readable media, such as a floppy disk 110, or CD. Removable media 110 provides programming instructions for subsequent execution by CPU 202 to configure

- 9 -

and enable system 104 to achieve the functionality depicted in the flow chart of Fig. 5, or can provide removable data storage if desired.

Network interface 208, in combination with a communications suite 214, enables suitable communication between computer 104 and other computers operationally connected via network 114. Examples of a conventional network interface can include an Ethernet card, a token ring card, a modem, or the like. Optionally, network interface 208 may also enable retrieval of transmitted programming instructions or data to configure and enable computer 104 to achieve the preferred embodiments of the invention. Optionally, aspects of the preferred embodiments of the invention can be enabled in various computer systems operationally networked to form a distributed computing environment.

Memory 204 includes volatile and persistent memory for storage of programming instructions 216 for instructing the CPU 202, data structures 218 such as data repositories, operating system 212, and communications suite 214. Preferably, memory 204 includes a combination of random access memory (RAM), read-only memory (ROM), and a hard disk storage device. It will be appreciated that programming instructions 216 can be delivered to memory 204 from an input/output device, such as a floppy disk 110 of Fig. 1 inserted in a floppy disk drive via input/output interface 210, or downloaded to memory 204 from network 114 via network interface 208.

Operating system 212 suitably co-operates with CPU 202 to enable various operational interfaces with various subsystems of computer 104, and for providing various operational functions, such as multitasking chores and the like. Communications suite 214 provides, through interaction with operating system 212 and network interface 208, suitable

- 10 -

communications protocols to enable appropriate communications with networked computing devices via network 114, such as TCP/IP, ethernet, token ring, and the like.

Referring to Fig. 4, there is depicted a software development environment for a preferred embodiment of the present invention. Fig. 4 includes figures 4A and 4B. Block 400 depicts a software development environment. Block 402 represents a module having a name 'WEB MODULE_1'. It will be appreciated that this invention can be used for types of modules yet to be developed, and that for the sake of simplifying the explanation of the operation of the invention, the name of a module will be used to identify the module. The invention is not limited to using names to identify modules, and a suitable identifier can be used for identifying a module. Block 404 represents a module having the name 'WEB MODULE_2'. Block 406 represents a module having the name 'EJB MODULE'. Block 408 represents a module having the name 'WEB MODULE_3'. Block 410 represents a module having the name 'DATA MODULE' which includes computer data.

Block 412 represents a server having the name 'WAS SERVER_1'. It will be appreciated that for the sake of simplifying the explanation of the operation of the invention the name of a server will be used to identify the server. The invention is not limited to using names to identify servers, and a suitable identifier can be used for identifying a server. Block 414 represents a server having the name 'TOMCAT SERVER'. Block 416 represents a server having the name 'WAS SERVER_2'. Block 418 represents a server having the name 'DB2 SERVER'.

Arrows 420 represents the servers that are currently configured to run modules. Server 412 is currently configured to run modules 404 and 406. Server 414 is currently configured to run module 408. Server 416 is currently

- 11 -

configured to run modules 404 and 408. Server 418 is currently configured to run module 410. A server has yet to be selected or configured for running module 402. Therefore, when a software developer requests to run module 404, either server 412 or 416 can run the module 404. However, when the software developer requests to run module 402, none of the servers will respond to the request, unless a table of block 450 is used to select and configure a suitable server being suitable for running the module, which will be described in the flowchart depicted in Fig. 5.

Block 429 represents computer instructions expressed in various forms. Block 430 depicts a software routine, hereinafter called a "routine", which includes computer programmed instructions for carrying out the flowchart depicted in Fig. 5. The instructions of block 430 are written in a suitable computer programming language such as Java. The instructions of block 430 are subsequently interpreted, compiled, or converted to generate instructions for execution by central processor 202, the instructions hereinafter called "executable code" 434. Code 434 is subsequently placed in memory 204 of computer 104, depicted in Fig. 2. Executable code 434 will be used for directing the CPU 202 to carry out the steps depicted in the flowchart of Fig. 5. Block 432 represents a set of subroutines 432 that carry out the steps depicted in the flowchart of Fig. 5.

Block 450 depicts a group of tables which will be used by the instructions of block 430 to select and configure suitable servers. For the sake of providing a simplified explanation for the operation of the invention. Separate tables are depicted by blocks 452, 454, and 456, each table being identifiable by name. The name of a table having data will be used to identify the table. The invention is not limited to using names to identify tables, and a suitable identifier can

- 12 -

be used for identifying a table. A table can be a look-up table, a file, a list, a map, or a suitable database.

Block 452 represents a priority table having names or identifiers of servers of software environment 400. Table 452 is not a list of all identified servers of environment 400. Table 452 includes a listing of identified servers that are identified by the user as being priority servers. Routine 430 will examine table 452 to select a suitable server, as will be explained in Fig. 5. The data contained in table 452 can be read, edited or modified, and subsequently updated or stored by a user via the keyboard or mouse 106 of Fig. 1, which can be accomplished via a suitable subroutine 432 of routine 430, or which can be accomplished via routine 430 following the steps depicted in Fig. 5. The table 452 includes names of suitable servers that are placed in a sequence or order in a preferred hierarchy, in which a topmost positioned, user-identified server will be initially selected to deploy a module. However, if the topmost positioned user-identified server is not suitable (i.e., the server is not currently configured to run the module), then the next (i.e., lower) identified server in table 452 is checked for suitability. A user determines a preferred hierarchical order of the selection of suitable servers that are listed or identified in table 452. When a user requests to run a module, table 452 is initially examined to locate or identify a suitable server that can run the module. If table 452 cannot provide a suitable server, then table 454 or table 456 can be examined to locate a suitable server. Persons having ordinary skill in the art of computer programming can set up table 452 so that an identifier of a server is associated with a rank, and that other suitable forms of ranking are possible for use in table 452.

- 13 -

Block 454 represents a preferred table having pairs of data in which each pair of data includes a name or identifier of a module, and includes a name of a preferred suitable server that corresponds to the name of the module. Table 454 is used for identifying the modules which can be run in environment 400 and for identifying the preferred suitable server. When a user requests to run a module and the module is cannot be identified from table 454, then table 456 can be examined for configuring a suitable server to run the module. For example, when it is required to run WEB MODULE_2, WAS SERVER_1 will be used to run WEB MODULE_2. It will be appreciated that table 454 can be substituted by inserting portions of table 454 in the modules. For sake of convenient explanation of the operation of the invention, table 454 will be used, and it will be understood that the scope of the invention can cover embodiments of the invention in which portions of table 454 can be embedded in various modules. For example, a module can have a deployment descriptor which would provide a convenient mechanism for identifying a preferred server for the module.

Block 456 represents a backup table including pairs of data in which each pair of data includes an identifier of a type of module and an identifier of a type of server that corresponds to the identifier of the type of module. The purpose of table 456 is to provide a list of types of servers that are known to be capable of supporting types of modules. Table 456 contains data provided by the manufacturer of code 434, such as a database in which data cannot be written to table 456. It will be appreciated that table 456 can be adapted so that code 434 can write data to table 456. If table 452 is empty or a suitable server is not identified in table 454, then code 434 refers to table 456 to obtain an identifier for a suitable server capable of running a module. When a module is required to run and the identifier of the type of module cannot be located from table 456, then the user can be given an

- 14 -

opportunity to select or choose a suitable server for running the module.

Referring to Fig. 5, there is depicted a flowchart illustrating a process for selecting and configuring servers to run modules in a software environment 400 depicted in Fig. 4. Fig. 5 includes Figs. 5A to 5H inclusive. The flow chart of Fig. 5 is a preferred embodiment of the invention. The flowchart is used for selecting and/or configuring a suitable server to run a module. Routine 430 of Fig. 4 will include computer programmed code or instructions for encoding the flowchart of Fig. 5. The programmed code will be used for directing CPU 202 of computer 104 of Fig. 2 to achieve operations illustrated in the flowchart in Fig. 5.

Referring to Fig. 5A, from S500 the process of the preferred embodiment begins. The operations depicted in Fig. 5 will be performed by the software routine 430 of Fig. 4, unless otherwise indicated in the description.

S502 receives a request to run a module. The request can be provided by a user via the keyboard/mouse 106 or from the networked computer 118 via network 114 of Fig. 1. If the user is not yet ready to immediately run or execute the module, then the requests received can be a request to configure a suitable server with an option to run the module at a subsequent time. S502 can be adapted so that the routine can wait to receive the request to run a module or configure a suitable server to run a module, or both.

S504 determines whether there are any suitable servers in the software environment 400 of Fig. 4 that are currently configured to run the module. If there are no suitable servers currently configured to run the module, then processing continues to S506. If there are any suitable

- 15 -

servers configured to run the module, processing continues to S538 of Fig. 5E. By proceeding to S506, the decision has been made that there is no suitable server defined or configured to run the module. Consequently, either a new suitable server must be created or an existing suitable server, which has not yet been configured to run the module, must be configured. By proceeding to S538, the decision has been made that there are multiple suitable servers (or at least one server) that can run the module.

S506 locates a name of a suitable server being suitable for running the module. The name of the suitable server can be located from priority table 452 of Fig. 4. To simplify the operation of the preferred embodiment of the invention, the priority table 452 can be removed from the tables 450, and S504, S506, S508, S510, S512, S514, and S516 can be removed from the flowchart of Fig. 5, and processing can flow from S502 to S518 and then subsequently proceed to S520 and through to S532. However, by including S502 to S516, the invention provides an additional mode of operation.

For example, assume the user needs to run module 404 WEB MODULE_2. In S506, table 452 will be examined in which WAS SERVER_1 is located. Since WAS SERVER_1 is a suitable server that has been previously configured to run WEB MODULE_2 then WAS SERVER_1 runs WEB MODULE_2, and processing continues to S510 of Fig. 5B. It is assumed that if the user needs to run module WEB MODULE_3, then table 452 is examined. In latter scenarios, the entries of table 452 will be cycled in a prioritized manner in which it will be found out that no suitable server can be identified or located from table 452 for which then processing continues to S518 of Fig. 5C.

S508, when performed, determines whether the located name (i.e., the identifier) of the suitable server was located from

- 16 -

table 452. If the name was located, then processing continues to S510 in Fig. 5B. If the name was not located, then processing continues to S518 in Fig. 5C.

Programming flow continues to S510 of Fig. 5B when a suitable server was not located or identified from table 452. S510 inserts the located name of the suitable server into the preferred table 454 of Fig. 4. S510 is performed to that next time that the same module is required to run, then table 454 will be in a position to provide identification of the suitable server for the module. S510 can include adapting a configuration file of the suitable server to run the module so that when the suitable server is requested to run the module, the suitable server can check its configuration and run the module. After modifying table 454, S512 configures the server having the name of the located name of the suitable server. S514 sends a command to the suitable server to subsequently run or execute the module. S516 stops the routine 430. Processing could also continue to S502 in which the routine 430 can remain in a wait state until it receives another request to run another module or configure another server.

Programming flow continues to S518 of Fig. 5C when table 452 does not provide identification of a suitable server from S508 of Fig. 5B. Performance of S518 locates a type of a suitable server being suitable for running the module. The type of the suitable server can be located from a backup table 456 of Fig. 4 because the suitable server was not located from priority table 452. S520 determines whether the located type of the suitable server was located from table 456. If the type of the suitable server was located from table 454, then processing continues to S522. If the type of the suitable server was not located from table 454, then processing continues to S534 of Fig. 5D. S522 creates a server of the located type of the suitable server that was identified or

- 17 -

located from table 456. The created suitable server becomes another server of the software environment 400 of Fig. 4. S524 inserts a name of the created server into the priority table 452 of Fig. 4, so that the next time the same module is required to run, the name of the suitable server can be identified from table 452. S526 inserts the name of the created suitable server into the preferred table 454 of Fig. 4, so that the next time the same module is required to run, the name of a suitable server can be identified from table 454. S526 provides a way to update the table 454 just in case table 452 happened to have been changed by a user at some future time. S528 configures the created suitable server to run the module. S530 executes the created suitable server, in which the executed created suitable server runs or executes the module. S532 stops the routine 430. Processing could also continue to S502 in which case the routine 430 can remain in a wait state until it receives another request to run another module or configure another server.

Programming flow continues to S534 of Fig. 5D when S520 of Fig. 5C does not locate a suitable server from table 456. S534 indicates a failure condition because the type of the suitable server was not located from table 456. S536 stops the routine 430. Processing could also continue to S502 in which case the routine 430 can remain in a wait state until it receives another request to run or deploy another module. S534 can also be adapted to allow a user to choose a suitable server since table 456 could not provide identification of the suitable server.

Programming flow continues to S538 of Fig. 5E if S506 of Fig. 5A determines that there is one suitable server. S538 determines whether there is one suitable server currently configured to run the module. If there is one suitable server that is currently configured to run the module, then

- 18 -

processing continues to S540. If there is more than one suitable server currently configured to run the module, then processing continues to S546. S538 and higher are used to add additional features to the operation of software routine 430. S540 inserts a name of the currently configured suitable server to into the preferred table 454 so that next time the same module is required to run, the name of the suitable server can be identified from table 454. S542 executes the currently configured suitable server to run or execute the module. S544 stops the routine 430. Processing could also continue to S502 in which case the routine 430 can remain in a wait state until it receives another request to run or deploy another module.

Programming flow continues to S546 of Fig. 5F when table 454 does not identify a suitable server in which table 452 will be subsequently examined in S546. S546 locates a name of a suitable server being suitable for running the module from table 454. S548 determines whether the name of the suitable server was located from table 454. If the name of the suitable server was located from table 454, then processing continues to S550. If the name was not located from table 454, then processing continues to S554. S550 executes a server having the located name of the suitable server, in which the executing server runs the module. S552 stops the routine 430. Processing could also continue to S502 in which case the routine 430 can remain in a wait state until it receives another request to run or deploy another module.

Programming flow continues to S554 of Fig. 5G when table 454 does not identify a suitable server in which table 452 will be examined in S554. S554 locates a name of a suitable server being suitable for running the module. The name of the suitable server can be located from the priority table 452 of Fig. 4. S556 determines whether the name of the suitable

- 19 -

server was located from table 452. If the name was located, then processing continues to S558. If the name was not located, then processing continues to S564. S558 inserts the located name of the suitable server into the preferred table 454 of Fig. 4, so that next time the same module is required to run, the suitable server can be located from table 454. S560 executes a server having the located name of the suitable server, in which the server runs the module. S562 stops the routine 430. Processing could also continue to S502 in which case the routine 430 can remain in a wait state until it receives another request to run or deploy another module.

Programming flow continues to S564 of Fig. 5H when table 452 does not identify a suitable server in which processing continues to S564 as a last resort to identify a suitable server. S564 choosing a name of any server being suitable for running the module. S566 inserting the chosen name of any suitable server into the preferred table 454 of Fig. 4. S568 executing a suitable server being identified by the chosen name of any suitable server, in which the suitable server having the chosen name runs or executes the module. S570 stopping the routine 430. Processing could also continue to step S502 in which case the routine 430 can remain in a wait state until it receives another request to run another module or configure another server.

For simplifying the explanation of the operation of the invention, an example of the use a method embodying the invention will involve a software development environment. It will be appreciated that the method provided by the preferred embodiment of the invention can be applied to other software systems such as software agents.

It will be appreciated that variations of some elements are possible to adapt the invention for specific conditions or

- 20 -

functions. The concepts of the present invention can be further extended to a variety of other applications that are clearly within the scope of this invention. Having thus described the present invention with respect to a preferred embodiment as implemented, it will be apparent to those skilled in the art that many modifications and enhancements are possible to the present invention without departing from the basic concepts as described in the preferred embodiment of the present invention. Therefore, what is intended to be protected by way of letters patent should be limited only by the scope of the following claims.

- 21 -

CLAIMS

1. A method for managing a plurality of servers comprising:
receiving a request to configure a suitable server being
suitable for running a module;
locating an identifier of the suitable server from a
table comprising identifiers of suitable servers; and
configuring the suitable server to run the module.
2. The method of claim 1 wherein the table further comprises
identifiers of modules corresponding to the identifiers
of suitable servers.
3. The method of claims 1 or 2 further comprising:
creating the suitable server; and
having the suitable server run the module.
4. The method of claims 2 or 3 wherein:
the identifier of the suitable server identifies a type
of a server; and
the identifier of the module identifies a type of a
module.
5. The method of claim 4 further comprising:
using a priority table comprising prioritized identifiers
of suitable servers for:
identifying a prioritized suitable server before
identifying the identifier of a suitable server from the
table;
configuring a server being identified from the priority
table to run the module;
having the suitable server being identified from the
priority table to run the module;
using a preferred table comprising identifiers of servers
corresponding to identifiers of modules for:

- 22 -

identifying a suitable server from the preferred table before identifying the suitable server from the table; configuring a server being identified from the preferred table to run the module; having the suitable server being identified from the preferred table to run the module.

6. The method of claims 2 or 3 wherein:
the identifier of the suitable server identifies a name of a server; and
the identifier of the module identifies a name of a module.
7. The method of claim 6 further comprising:
using a priority table comprising prioritized identifiers of suitable servers for:
identifying a suitable server from the priority table before identifying the suitable server from the table; configuring a server being identified from the priority table to run the module;
having the suitable server being identified from the priority table to run the module;
using a backup table comprising types of servers corresponding to types of modules for:
identifying a type of a suitable server corresponding to a type of module from the backup table when the table does not provide the identifier of the suitable server; configuring a server being identified from the backup table to run the module;
having the suitable server being identified from the backup table to run the module.
8. The method of claim 1 wherein the table comprises prioritized identifiers of suitable servers, and the method further comprises:

- 23 -

creating the suitable server; and
having the suitable server run the module.

9. The method of claim 8 further comprising:
using a preferred table comprising identifiers of servers
corresponding to identifiers of modules for:
identifying a preferred suitable server from the
preferred table when the table does not provide the
identifier of the suitable server;
configuring a server being identified from the preferred
table to run the module;
having the server being identified from the preferred
table to run the module;
using a backup table comprising types of servers
corresponding to types of modules for:
identifying a type of a suitable server from the backup
table when the table does not provide the identifier of
the suitable server;
configuring a server being identified from the backup
table to run the module;
having the suitable server being identified from the
backup table to run the module.
10. The method of claims 5, 7 or 9 wherein the method is
performed in a software development environment.
11. A computer program product for use in a computer system
operatively coupled to a computer readable memory, the
computer program product including a computer-readable
data storage medium tangibly embodying computer readable
program code for directing said computer to for manage a
plurality of servers, said code comprising:
code for instructing said computer system to receive a
request to configure a suitable server being suitable for
running a module;

- 24 -

code for instructing said computer system to locate an identifier of the suitable server from a table comprising identifiers of suitable servers;
and code for instructing said computer system to configure the suitable server to run the module.

12. The computer program product of claim 11 wherein the table further comprises identifiers of modules corresponding to the identifiers of suitable servers.
13. The computer program product of claims 11 or 12 further comprising:
code for instructing said computer system to create the suitable server; and
code for instructing said computer system to have the suitable server run the module.
14. The computer program product of claims 12 or 13 wherein:
the identifier of the suitable server identifies a type of a server; and
the identifier of the module identifies a type of a module.
15. The computer program product of claim 14 further comprising:
code for instructing said computer system to use a priority table comprising prioritized identifiers of suitable servers for:
identifying a prioritized suitable server before identifying the identifier of a suitable server from the table;
configuring a server being identified from the priority table to run the module;
having the suitable server being identified from the priority table to run the module;

- 25 -

code for instructing said computer system to use a preferred table comprising identifiers of servers corresponding to identifiers of modules for:
identifying a suitable server from the preferred table before identifying the suitable server from the table;
configuring a server being identified from the preferred table to run the module;
having the suitable server being identified from the preferred table to run the module.

16. The computer program product of claims 12 or 13 wherein:
the identifier of the suitable server identifies a name of a server; and
the identifier of the module identifies a name of a module.

17. The computer program product of claim 16 further comprising:
code for instructing said computer system to use a priority table comprising prioritized identifiers of suitable servers for:
identifying a suitable server from the priority table before identifying the suitable server from the table;
configuring a server being identified from the priority table to run the module;
having the suitable server being identified from the priority table to run the module;
code for instructing said computer system to use a backup table comprising types of servers corresponding to types of modules for:
identifying a type of a suitable server corresponding to a type of module from the backup table when the table does not provide the identifier of the suitable server;
configuring a server being identified from the backup table to run the module;

- 26 -

having the suitable server being identified from the backup table to run the module.

18. The computer program product of claim 11 wherein the table comprises prioritized identifiers of suitable servers, and the method further comprises:
code for instructing said computer system to create the suitable server; and
code for instructing said computer system to have the suitable server run the module.
19. The computer program product of claim 18 further comprising:
code for instructing said computer system to use a preferred table comprising identifiers of servers corresponding to identifiers of modules for:
identifying a preferred suitable server from the preferred table when the table does not provide the identifier of the suitable server;
configuring a server being identified from the preferred table to run the module;
having the server being identified from the preferred table to run the module;
code for instructing said computer system to use a backup table comprising types of servers corresponding to types of modules for:
identifying a type of a suitable server from the backup table when the table does not provide the identifier of the suitable server;
configuring a server being identified from the backup table to run the module;
having the suitable server being identified from the backup table to run the module.

- 27 -

20. The computer program product of claims 15, 17 or 19 wherein the computer program product is used in a software development environment.
21. A computer system operatively coupled to a computer readable memory, the computer system for managing a plurality of servers, the computer system comprising:
means for receiving a request to configure a suitable server being suitable for running a module;
means for locating an identifier of the suitable server from a table comprising identifiers of suitable servers;
and
means for configuring the suitable server to run the module.
22. The computer system of claim 21 wherein the table further comprises identifiers of modules corresponding to the identifiers of suitable servers.
23. The computer system of claims 21 or 22 further comprising:
means for creating the suitable server; and
means for having the suitable server run the module.
24. The computer system of claims 22 or 23 wherein:
the identifier of the suitable server identifies a type of a server; and
the identifier of the module identifies a type of a module.
25. The computer system of claim 24 further comprising:
means for using a priority table comprising prioritized identifiers of suitable servers for:

- 28 -

identifying a prioritized suitable server before
identifying the identifier or a suitable server from the
table;
configuring a server being identified from the priority
table to run the module;
having the suitable server being identified from the
priority table to run the module;
means for using a preferred table comprising identifiers
of servers corresponding to identifiers of modules for:
identifying a suitable server from the preferred table
before identifying the suitable server from the table;
configuring a server being identified from the preferred
table to run the module;
having the suitable server being identified from the
preferred table to run the module.

26. The computer system of claims 22 or 23 wherein:
the identifier of the suitable server identifies a name
of a server; and
the identifier of the module identifies a name of a
module.
27. The computer system of claim 26 further comprising:
means for using a priority table comprising prioritized
identifiers of suitable servers for:
identifying a suitable server from the priority table
before identifying the suitable server from the table;
configuring a server being identified from the priority
table to run the module;
having the suitable server being identified from the
priority table to run the module;
means for using a backup table comprising types of
servers corresponding to types of modules for:
identifying a type of a suitable server corresponding to
a type of module from the backup table when the table

- 29 -

does not provide the identifier of the suitable server;
configuring a server being identified from the backup
table to run the module;
having the suitable server being identified from the
backup table to run the module.

28. The computer system of claim 21 wherein the table comprises prioritized identifiers of suitable servers, and the method further comprises:
means for creating the suitable server; and
means for having the suitable server run the module.
29. The computer system of claim 28 further comprising:
means for using a preferred table comprising identifiers of servers corresponding to identifiers of modules for:
identifying a preferred suitable server from the preferred table when the table does not provide the identifier of the suitable server;
configuring a server being identified from the preferred table to run the module;
having the server being identified from the preferred table to run the module;
means for using a backup table comprising types of servers corresponding to types of modules for:
identifying a type of a suitable server from the backup table when the table does not provide the identifier of the suitable server;
configuring a server being identified from the backup table to run the module;
having the suitable server being identified from the backup table to run the module.
30. The computer system of claims 25, 27 or 29 wherein the computer system is used in a software development environment.

- 30 -

31. A set of data for configuring and selecting a server to run a module, the set of data comprising a table for containing an identifier of a suitable server being suitable for running the module.
32. The set of data of claim 31 wherein the identifier identifies a type of module and the data set further comprises an identifier for identifying a type of server corresponding to the identifier of the type of the module.
33. The set of data of claim 31 wherein the identifier identifies a name of a server and the data set further comprises an identifier for identifying a corresponding name of the module.
34. The set of data of claim 31 wherein the items of the data set are prioritized.

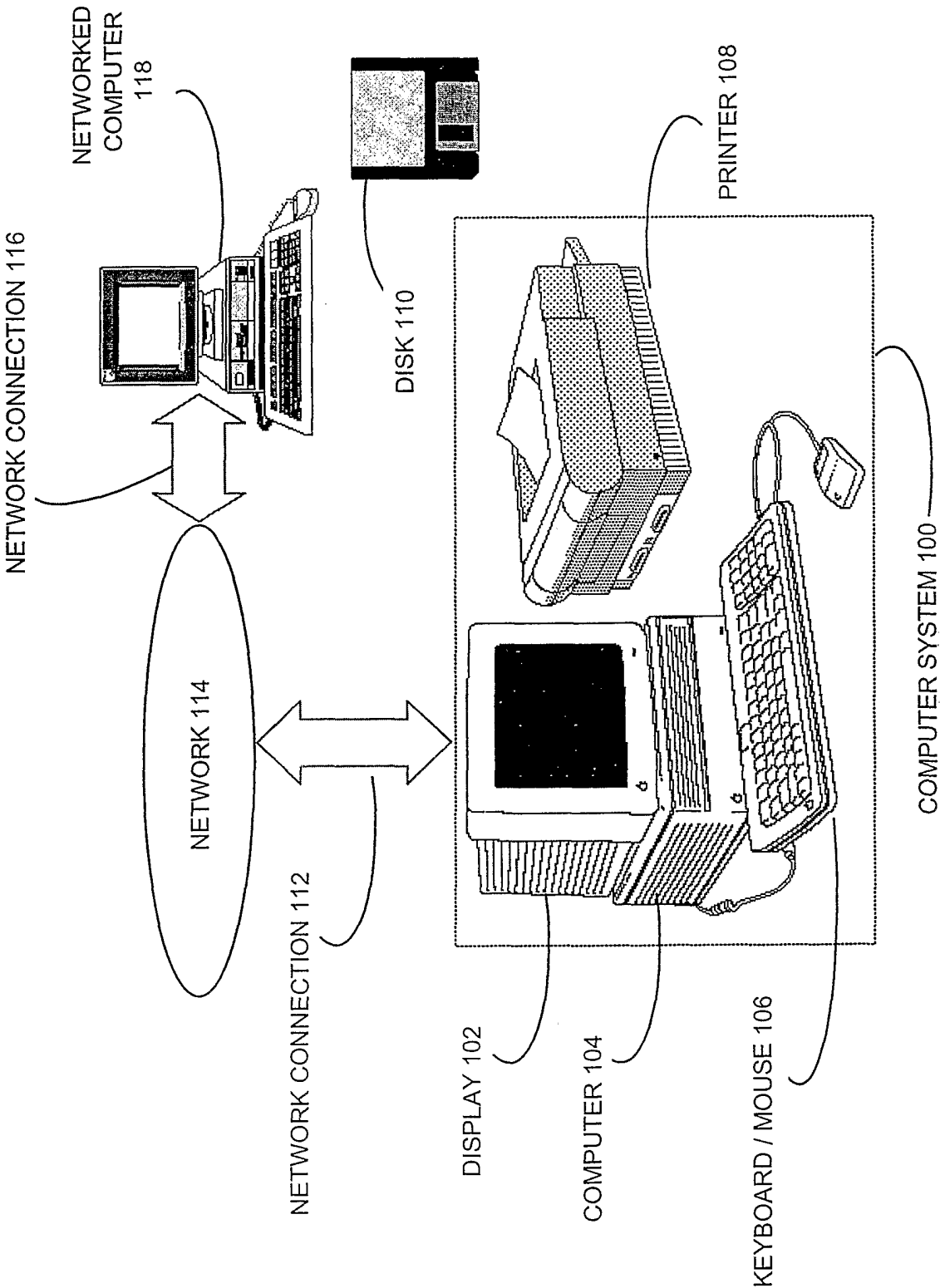


FIG. 1

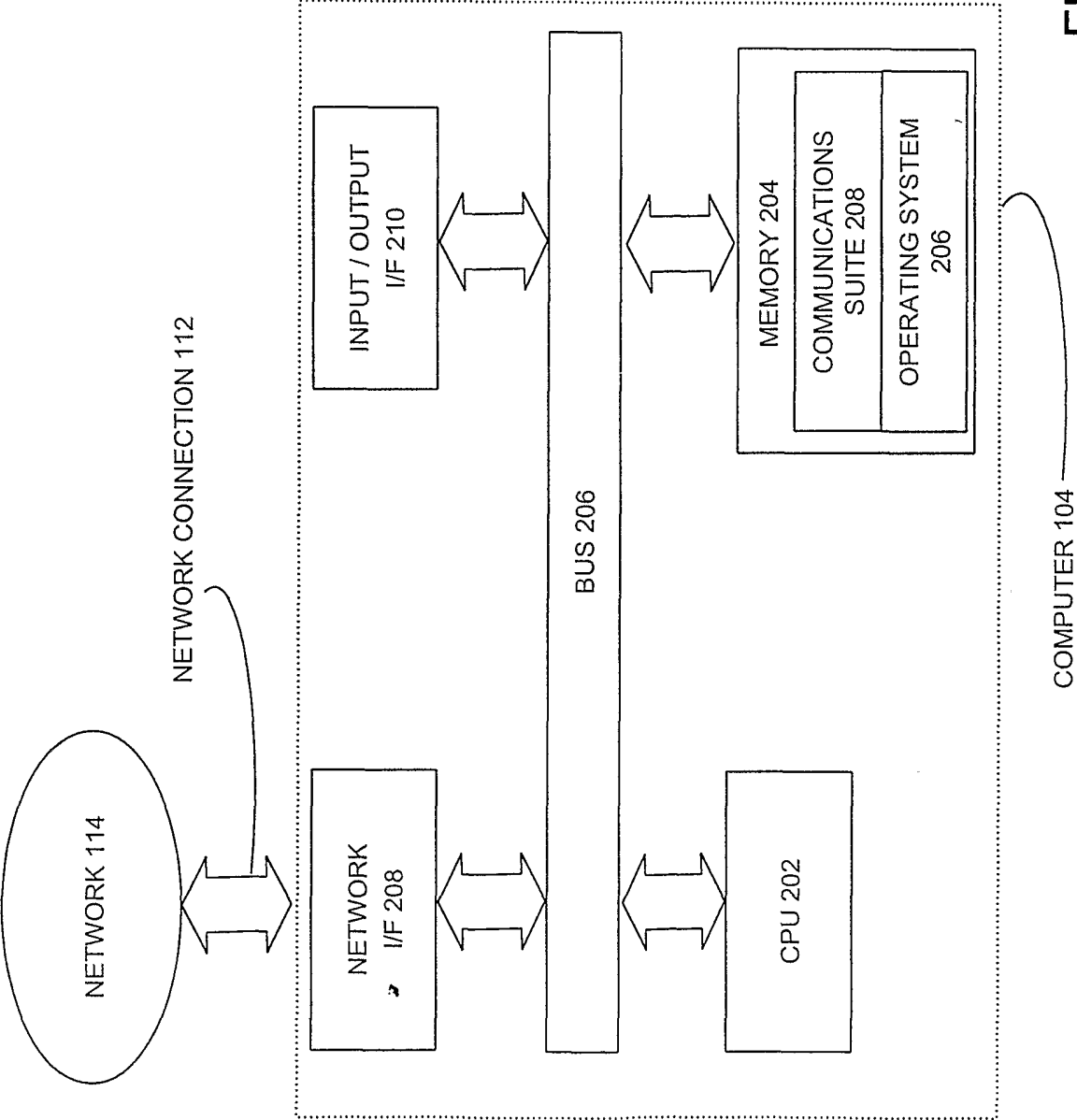
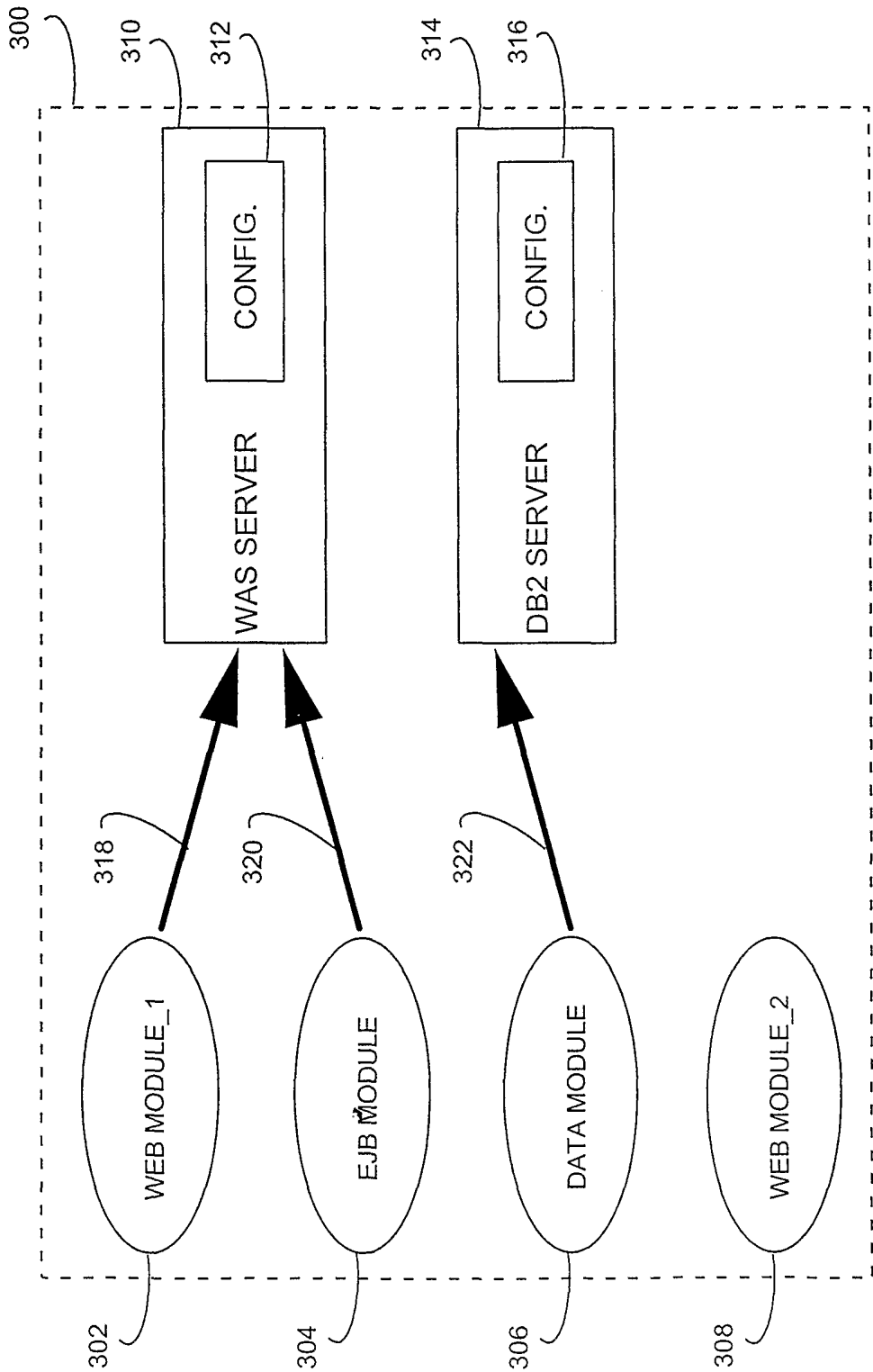


FIG. 2



PRIOR ART

FIG. 3

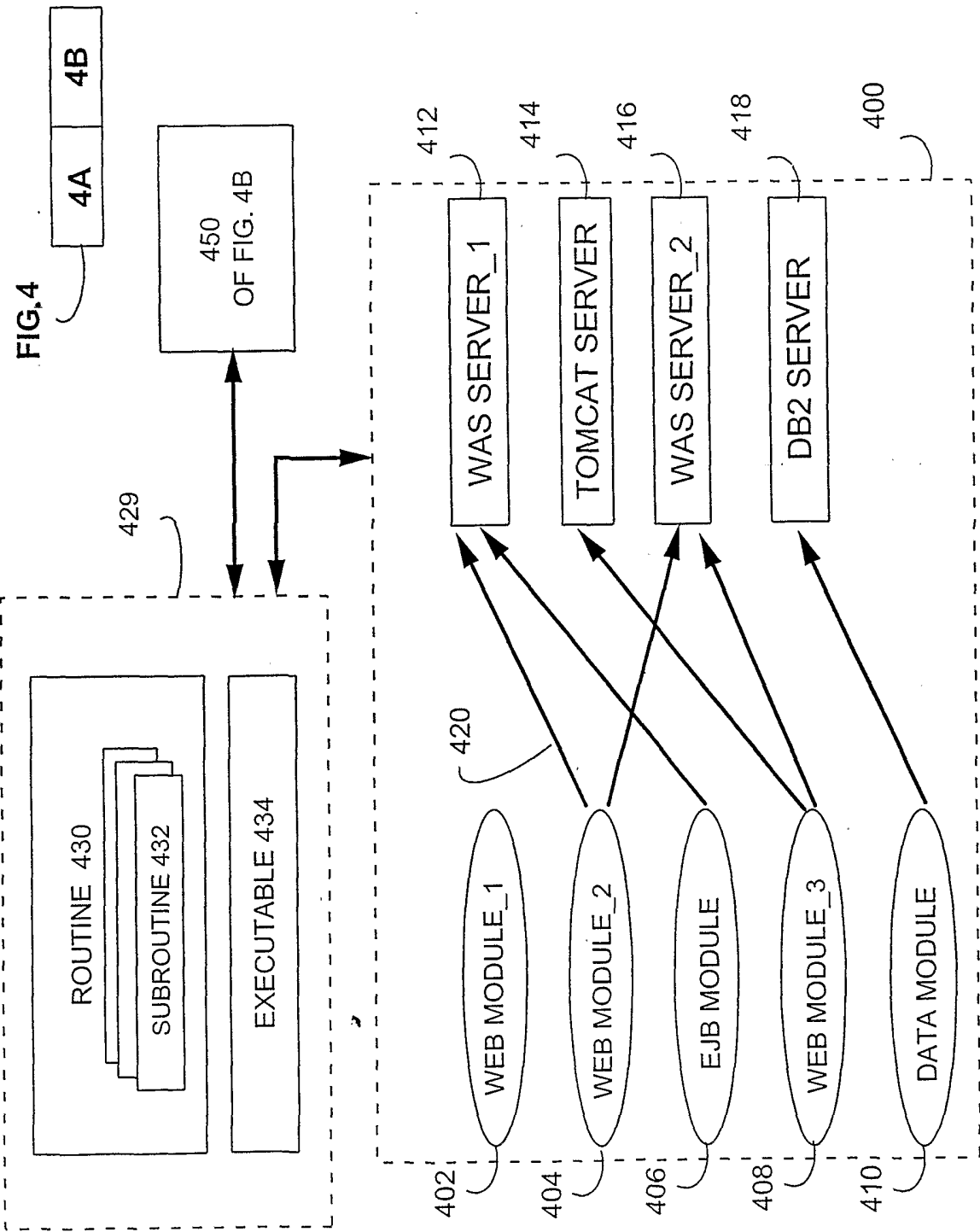


FIG. 4A

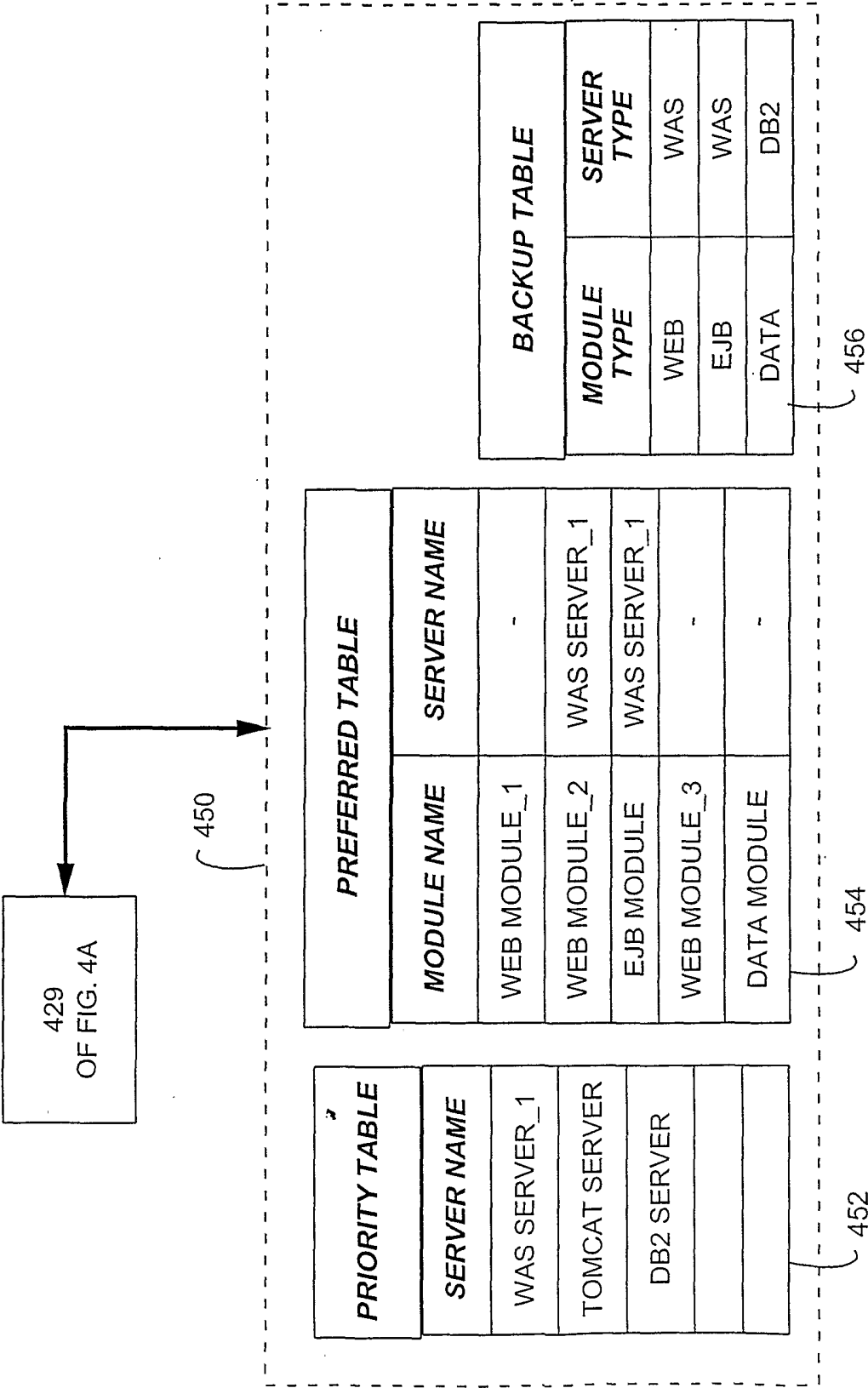
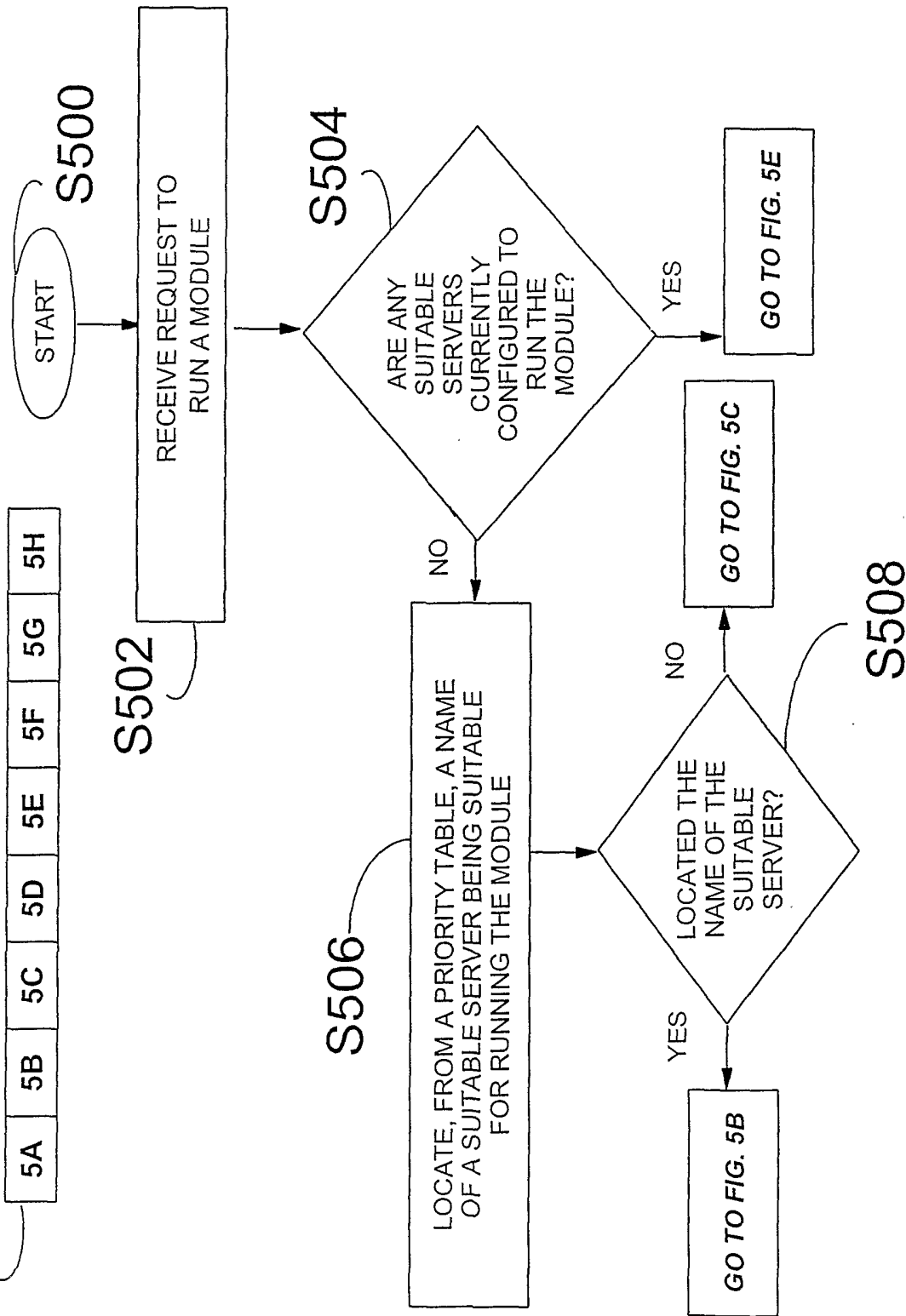


FIG. 4B

FIG.5



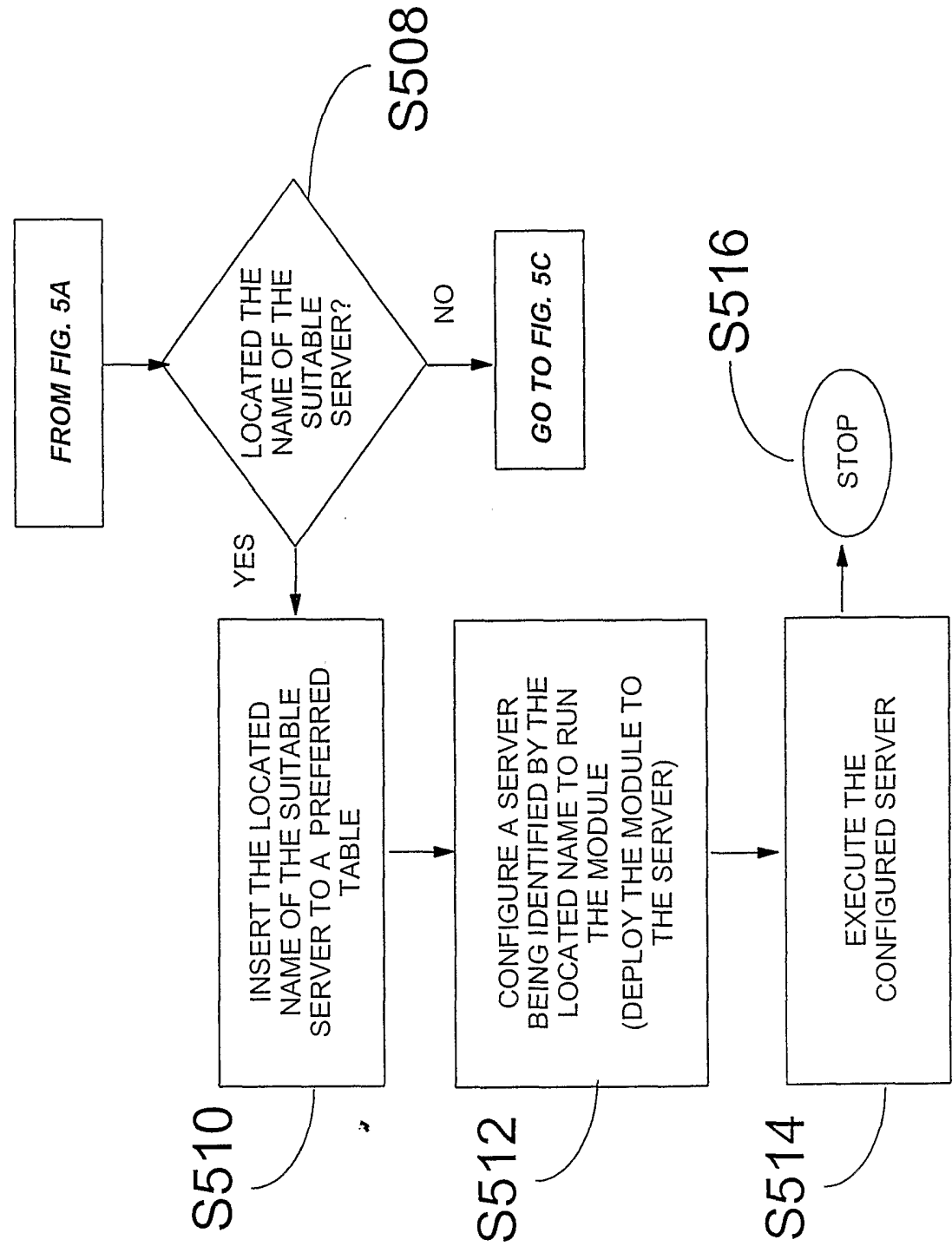


FIG. 5B

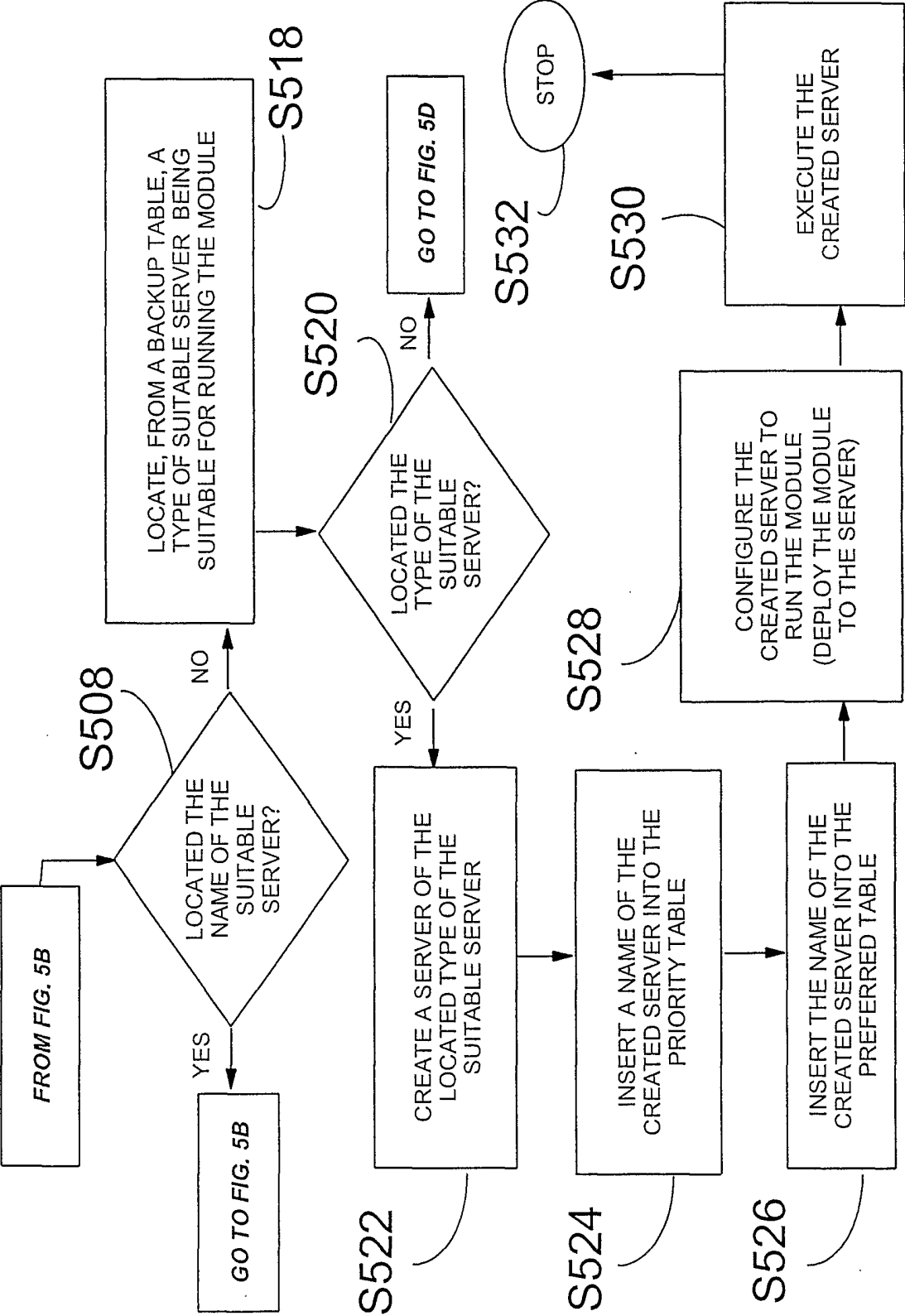


FIG. 5C

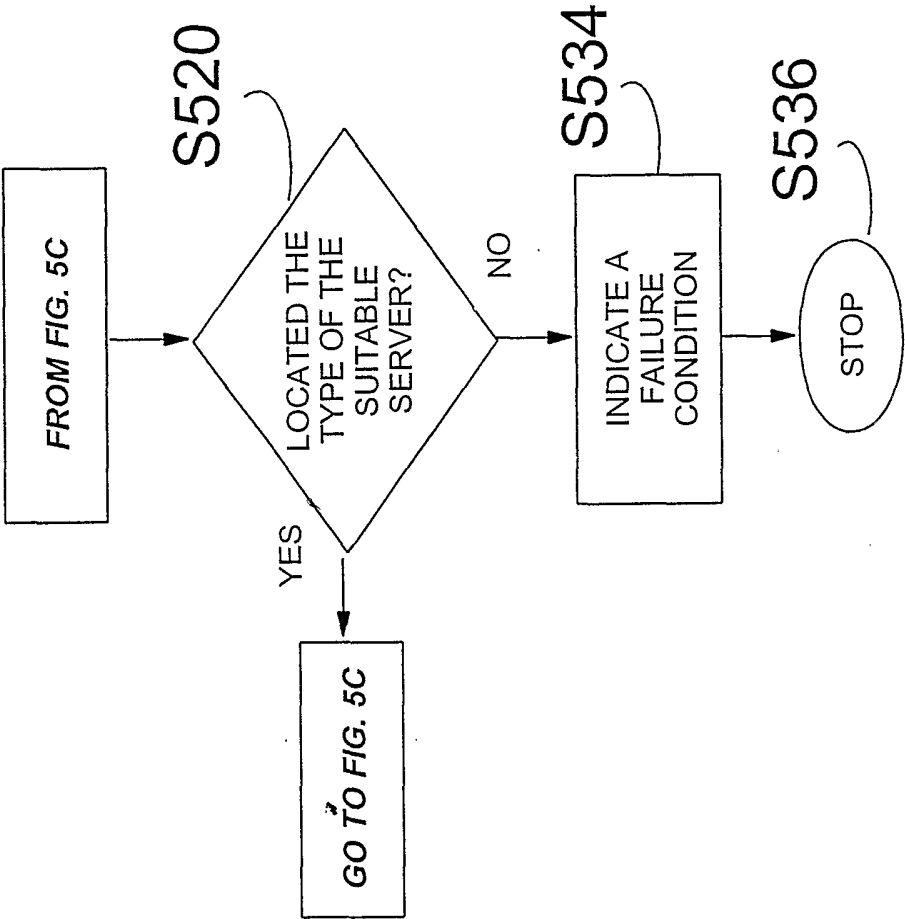


FIG. 5D

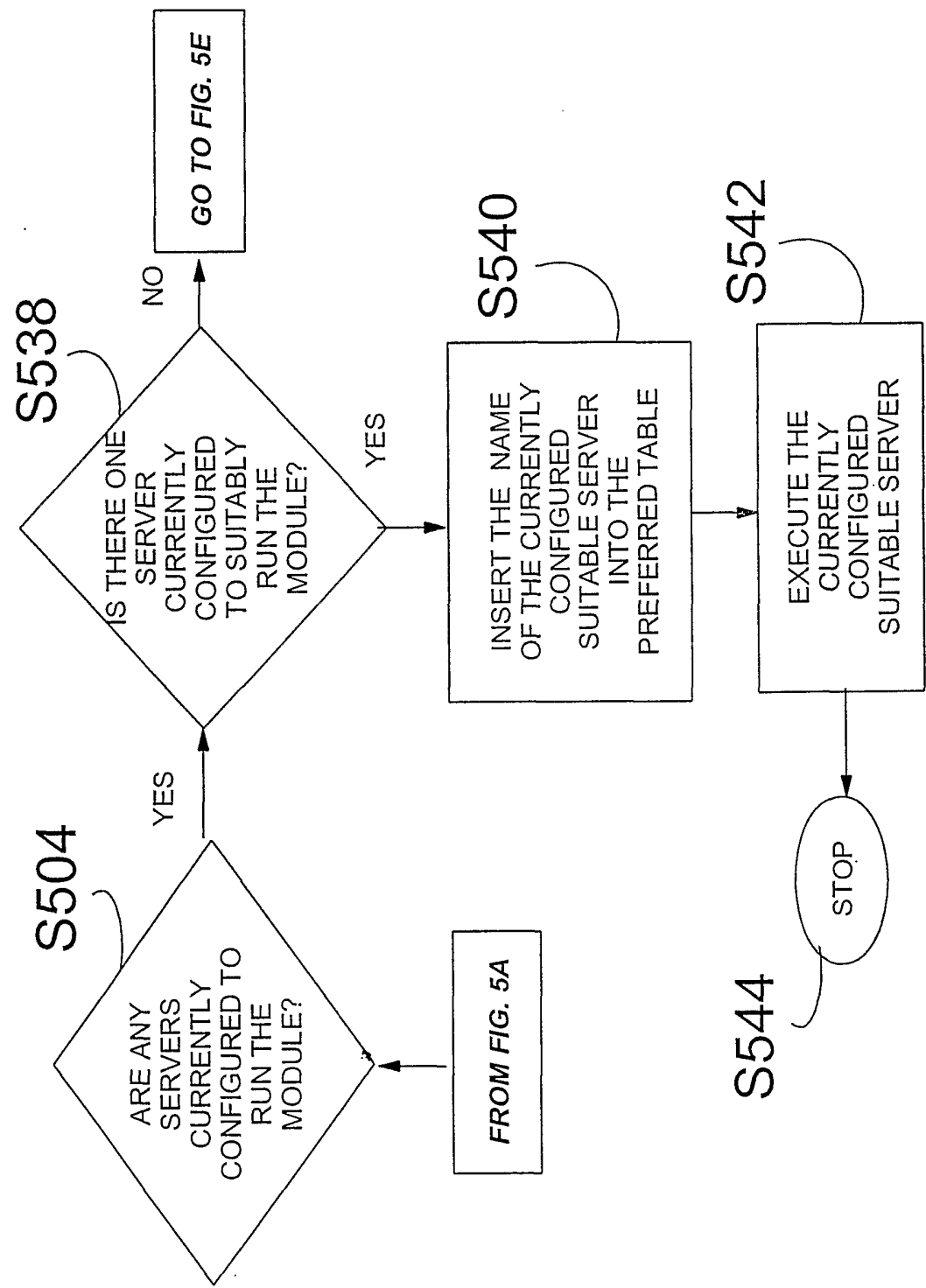


FIG. 5E

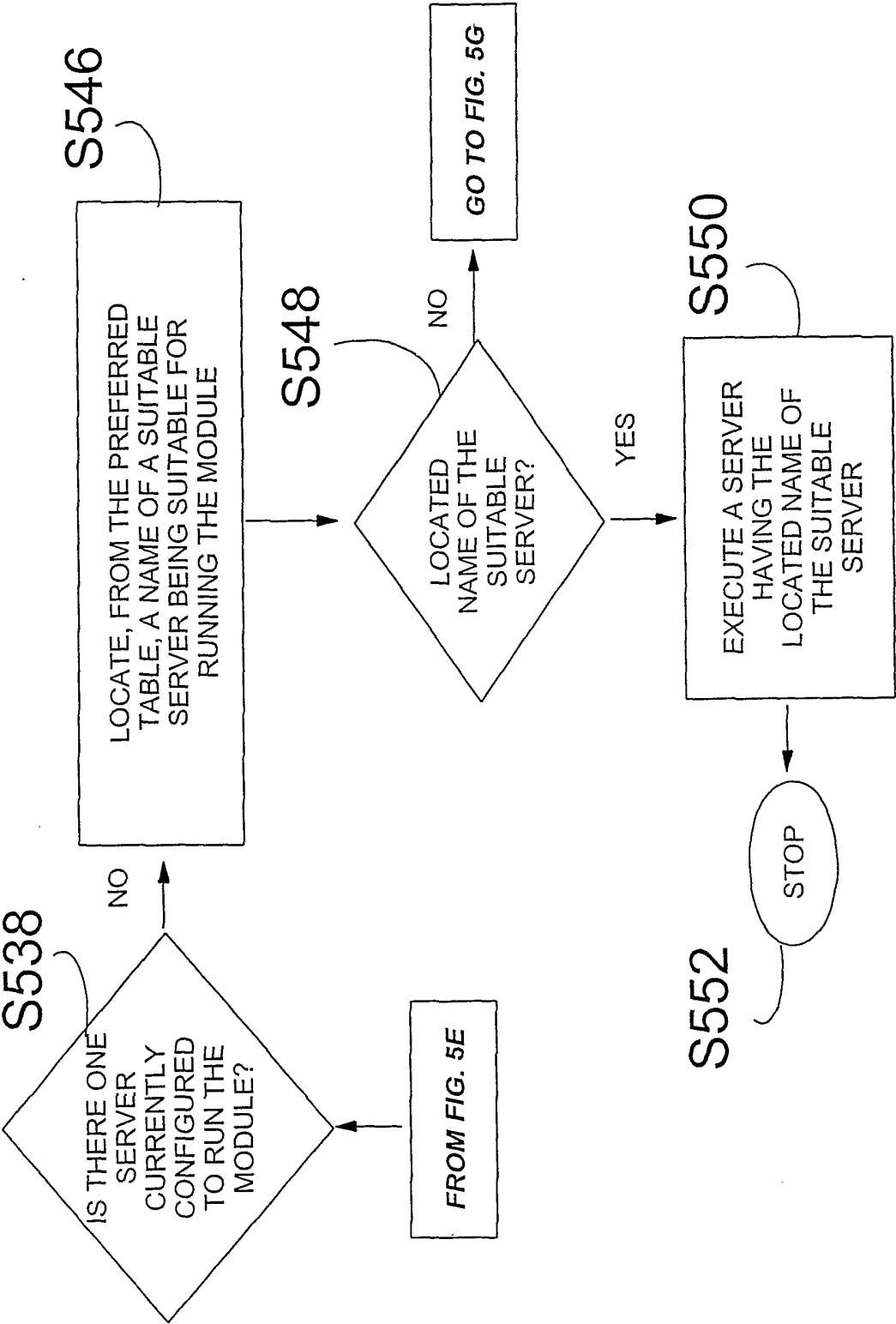


FIG. 5F

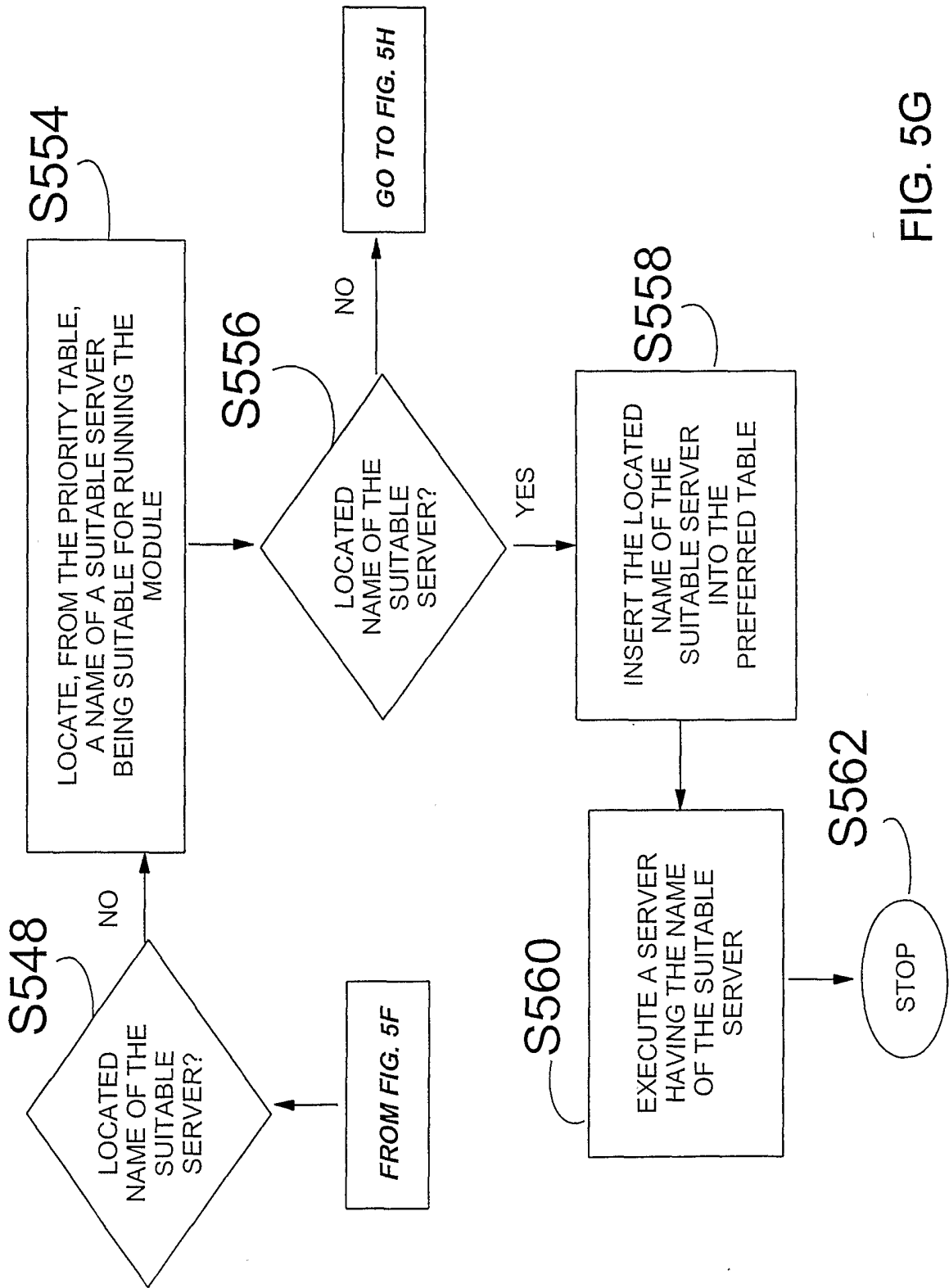


FIG. 5G

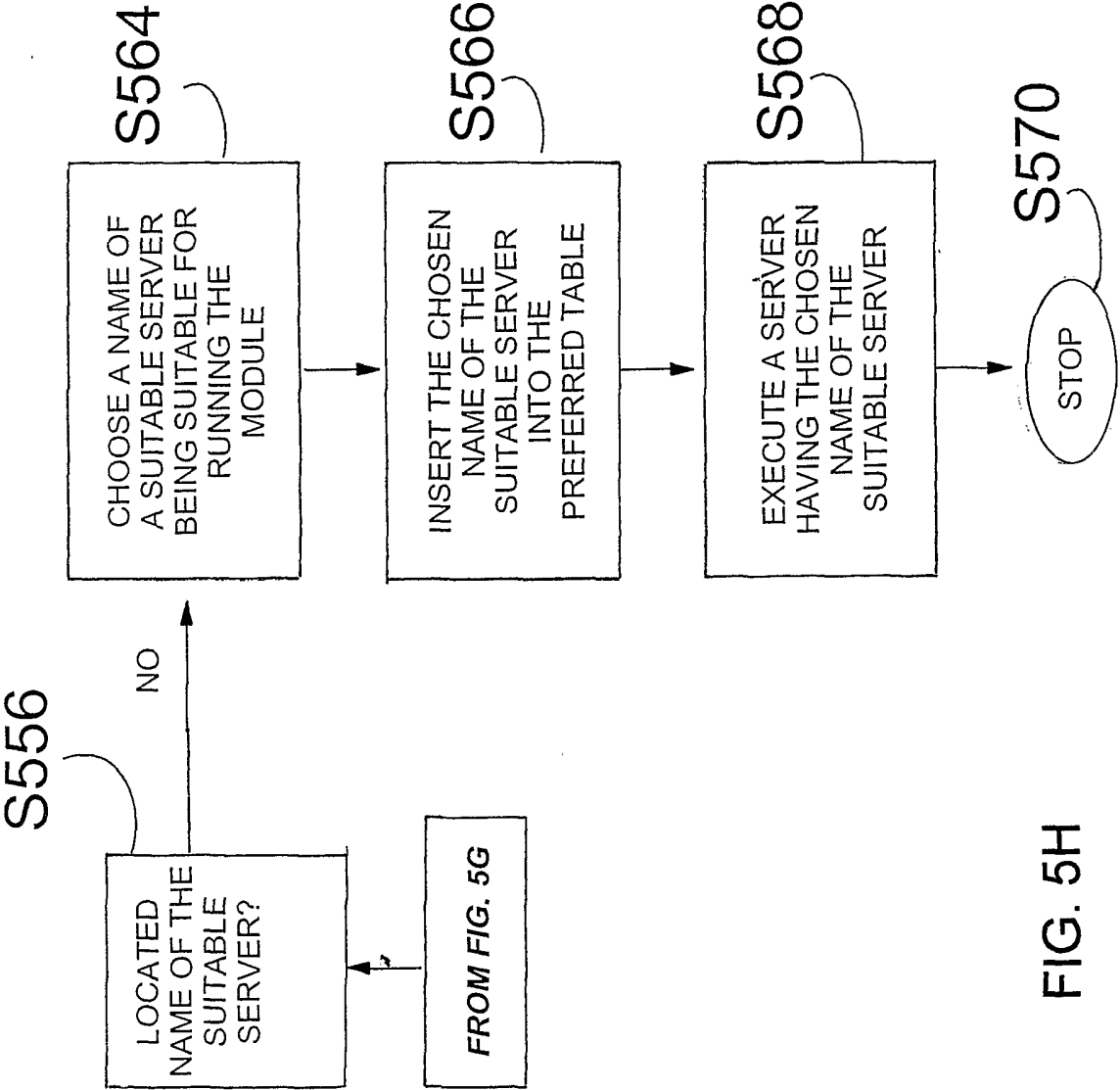


FIG. 5H