



(11) (21) (C) **2,216,901**
(22) 1997/09/26
(43) 1999/03/26
(45) 2000/12/26

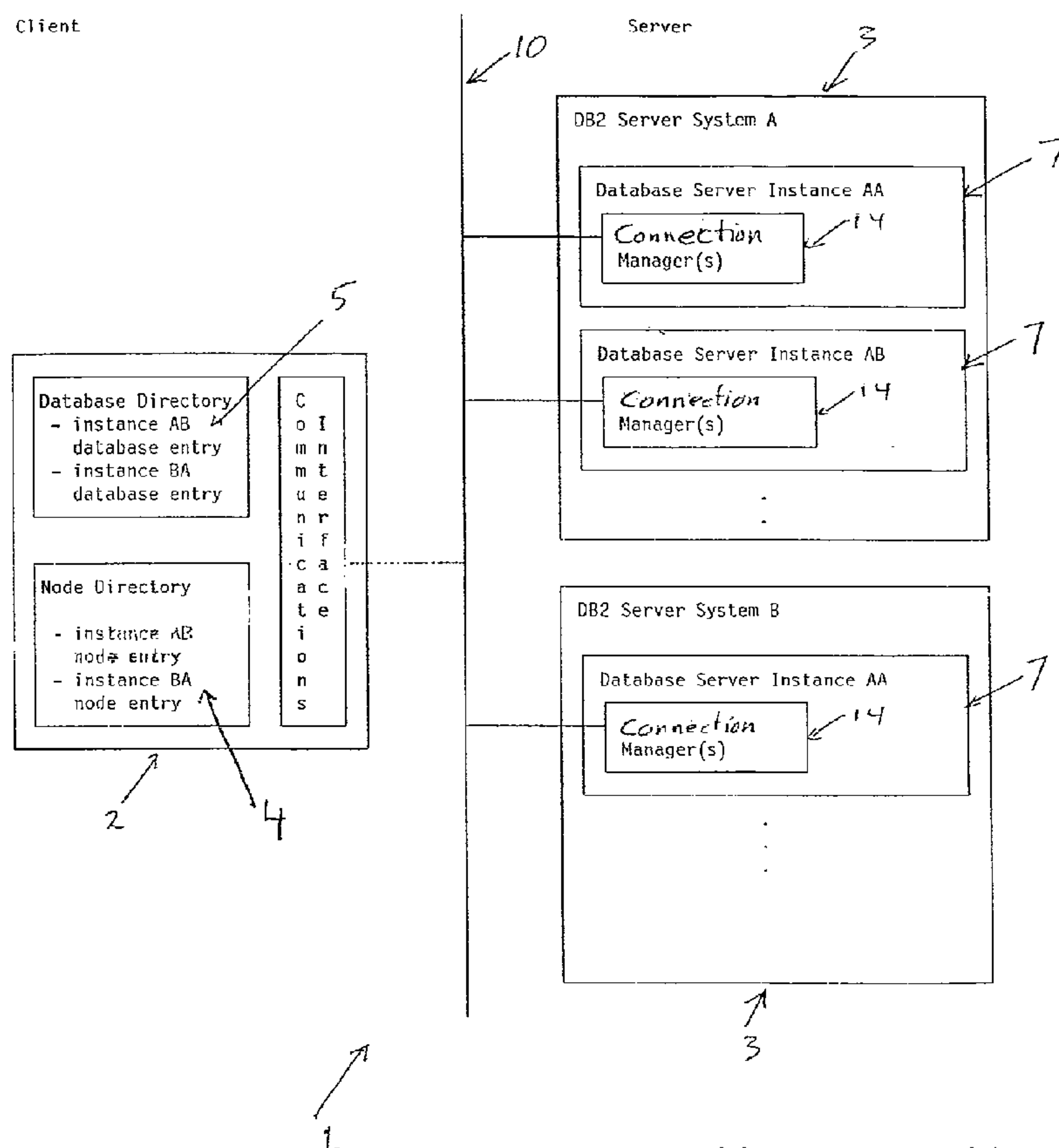
(72) HSU, JULIANA MIN-NI, CA

(73) IBM CANADA LTD. - IBM CANADA LIMITÉE, CA

(51) Int.Cl.⁶ G06F 17/30

(54) **METHODE ET DISPOSITIF DE DECOUVERTE DE BASES DE
DONNEES DANS UN RESEAU CLIENT-SERVEUR**

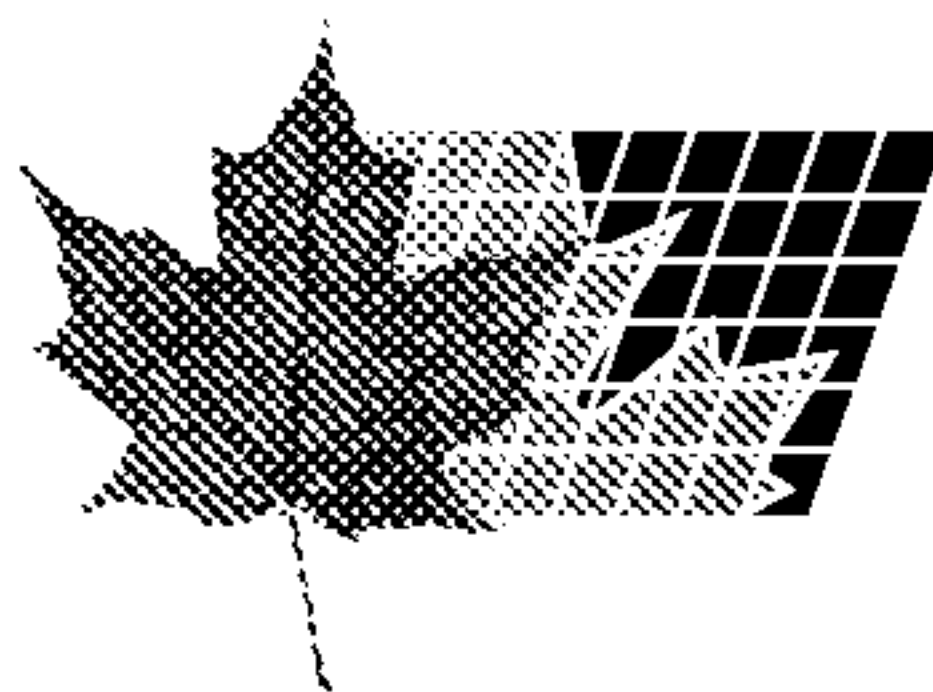
(54) **METHOD AND APPARATUS FOR DISCOVERY OF
DATABASES IN A CLIENT SERVER NETWORK**



(57) L'invention a trait à des réseaux de traitement d'information client-serveur comprenant un ou plusieurs systèmes serveurs de bases de données. Sont présentés une méthode et des moyens permettant aux clients des bases de données d'identifier et de consulter les serveurs de bases de données qui sont sur le réseau ou interconnectés à celui-ci de façon qu'ils soient accessibles à partir dudit réseau. À cette fin, une méthode et un dispositif permettent de réaliser un réseau de traitement d'information client-serveur doté d'un ou de plusieurs systèmes serveurs de bases de données commandant au moins une base de données, au moins un

(57) This invention relates to client server information handling networks having one or more database server systems in which a method and means are provided for database clients to identify and access database servers which are on the network or interconnected therewith in such a manner that they are accessible from said network by providing a method and apparatus for implementing a client server information handling network having one or more database server systems controlling at least one database, at least one database client and a communications link between the servers and the client; the network including: discovery module for the





(11) (21) (C) **2,216,901**
(22) 1997/09/26
(43) 1999/03/26
(45) 2000/12/26

client de base de données et une liaison de communication entre les serveurs et le client. Le réseau comprend un module de découverte permettant au client d'identifier les serveurs de bases de données, qui sont accessibles sur le réseau. Le module de découverte comprend : un module de découverte/recherche associé au client de base de données afin de générer une interrogation de découverte de bases de données comprenant une demande de découverte/recherche de base de données visant à identifier les systèmes serveurs de bases de données; un module gestionnaire de découverte de bases de données associé aux serveurs de bases de données et adapté pour répondre à la demande de découverte/recherche en renvoyant de l'information d'adressage afin de permettre audit client d'avoir accès aux serveurs de bases de données.

database client to identify database servers accessible with the network. The discovery module includes: a search discovery module associated with the database client for generating a database discovery inquiry including a database search discovery request to identify the database server systems; a database discovery manager module associated with the database servers, the database discovery manager module being adapted to respond to the search discovery request by returning addressing information to permit access to the database servers by the database client.

CA9-97-042

Abstract

This invention relates to client server information handling networks having one or more database server systems in which a method and means are provided for database clients to identify and access database servers which are on the network or interconnected therewith in such a manner that they are accessible from said network by providing a method and apparatus for implementing a client server information handling network having one or more database server systems controlling at least one database, at least one database client and a communications link between the servers and the client; the network including: discovery module for the database client to identify database servers accessible with the network. The discovery module includes: a search discovery module associated with the database client for generating a database discovery inquiry including a database search discovery request to identify the database server systems; a database discovery manager module associated with the database servers, the database discovery manager module being adapted to respond to the search discovery request by returning addressing information to permit access to the database servers by the database client.

**METHOD AND APPARATUS FOR DISCOVERY
OF
DATABASES IN A CLIENT SERVER NETWORK**

5

Field of the Invention

This invention relates to client server information handling networks having one or more database server systems in which a method and means are provided for database clients to identify and access database servers which are on the network or
10 interconnected therewith in such a manner that they are accessible from the network.

Background of the Invention

In client server networks having database servers, when a client desires to access one or more of the database servers the client requires communications information for
15 each server. This communications information includes the protocols supported by the server and the protocol specific address information, which may include one or more items of data. For instance in networks using the TCP/IP protocol, the address includes a port number or service name and host name or IP address. Using the NETBIOS protocol, an Nname (workstation name) and local adapter is required. Using the
20 IPX/SPX protocol, an internetwork address is required comprising an <8 byte net ID>.<12 byte node ID>.<4 byte socket #>. Using the APPC protocol, a significant amount of complex information is required for accessing databases connected to a network.

25 Considering the DB2 database management products introduced by IBM several years ago the establishment of a client server connection to access a database requires a user of a database client to have knowledge of a number of items of information. Before a connection can be established, the user must catalog a node and database directory. To catalog a node directory the user must have an intersection of the
30 protocols supported by both the database client and database server, and enough

CA9-97-042

2

protocol-specific communications information to communicate from the database client to database server. For example to connect to a DB2 server by using the TCP/IP protocol, the user must specify a host name or IP address, and a service name or port number in the node directory entry. To catalog a database directory the user must specify the database name. One of the problems inherent in this procedure is that it requires the user to know which databases are available and to have sufficient skill and communications information to catalog the entries required to access the database. It would be preferable for the database client to be able to simply identify the databases to which the user can have access and to simplify the mode by which the user can connect to those databases.

Summary of the Invention

The invention herein provides these and other advantages over prior systems. One implementation of the invention provides a client server information handling network having at least one database server system controlling at least one database, at least one database client and a communications link between the servers and the client; the network including:

- discovery module for the database client to identify database servers accessible with the network. The discovery module includes:

- a search discovery module associated with the database client for generating a database discovery inquiry including a database search discovery request to identify the database server systems;

- a database discovery manager module associated with the database servers, the database discovery manager module being adapted to respond to the search discovery request by returning addressing information to permit access to the database servers by the database client.

The network may include a module for the client to respond to receive the addressing information provided by the database discovery manager module and to display the information to a user of the client.

CA9-97-042

3

The network of claim may include a known discovery module for the client for generating a database known discovery request to a selected database server system previously identified to determine detailed database server system information.

5

The database connection manager module associated with the selected database server system may be adapted to respond to the known discovery request by packaging detailed database server system information relating to the selected database server system to permit access to databases on the system.

10

A module may be provided for the client to respond to receive the detailed database server system information and to display the information to a user of the client.

15

A graphical display interface (GUI) is preferably used to display server system information to a user, the GUI being adapted to respond to user input to initiate the search discovery request and the known discovery request.

20

The GUI should be responsive to the selection by a user, to initiate the search discovery request, and display server system names returned, and be further responsive to the selection of a database server system displayed on the GUI to expand the display of the selected database server system by initiating the generation of a known search request and depicting the detailed server system information returned in treelike fashion identifying the databases controlled by the selected database server system.

25

The GUI is responsive to user selection of an identified database by auto-configuring for a database access request.

30

The database server system may control a plurality of database server instances, and wherein each of the database server instances controls at least one database.

CA9-97-042

4

The database server system may include:

discovery manager and connection manager module adapted to listen for discovery requests on the network, and comprised of:

5 request receiver module to receive the discovery requests,
request parser module to parse the requests for processing,
request service module for responding to the parsed requests, and in response to:

10 (a) the receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and

(b) the receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with the specified database server system and databases controlled by the database server instances; and

15 communications response module adapted to transmit the packaged database server system addressing information and the packaged detailed database server system information to any database client requesting the information.

20 The modules mentioned above preferably are embodied in software routines. Modules identified singularly can be replaced by multiple modules and conversely multiple modules could be replaced by singular modules.

25 Another aspect of the invention is embodied in the method of operation of the client server network described for identifying (and then accessing databases as desired) including:

generating a database discovery inquiry for the client including a database search discovery request to identify the database server systems;

30 using database discovery manager module associated with the database server system to respond to the search discovery request by returning addressing information to permit access to the database servers by the database client.

CA9-97-042

5

The database client responds to receive the addressing information provided by the database discovery manager module to display the information to a user of the database client.

5

The method preferably includes generating a database known discovery request for the client to a selected database server system previously identified to determine detailed database server system information.

10 The selected database server responds to the known discovery request by packaging detailed database server system information relating to the selected database server system to permit access to databases on the system.

15 The client responds to receive the detailed database server system information and displays the information to a user of the client.

A graphical display interface module (GUI) responds to user input to initiate the search discovery request and the known discovery request.

20 The GUI responds to selection by a user to initiate the search discovery request, it displays server system names returned, and further responds to the selection of a database server system displayed on the GUI by expanding the display of the selected database server system through initiation of a known discovery request. Detailed addressing information returned is displayed in treelike fashion, identifying the
25 databases controlled by the selected database server system.

The GUI further responds to selection of an identified database by auto-configuring for a database access request.

CA9-97-042

6

The database server system in an embodiment of the invention listens for discovery requests on the network, receives the discovery requests, parses the requests for processing, responds to the parsed requests, and in response to:

(a) the receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and

(b) the receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with the specified database server system and databases controlled by the database server instances;

and is adapted to transmit the packaged database server system addressing information and the packaged detailed database server system information to any database client requesting the information.

As was indicated, the invention may be implemented by a computer program product that may be conveniently stored on a storage medium such as tape or disk. The computer program product is designed for establishing a client server information handling network having at least one database server system controlling at least one database, at least one database client and a communications link between the servers and the client. The computer program product includes:

a software routine for establishing a discovery module for the database client to identify database servers accessible with the network, the discovery module including:

search discovery module associated with the database client for generating a database discovery inquiry including a database search discovery request to identify the database server systems;

a software routine for establishing a database discovery manager module associated with the database servers, the database discovery manager module being adapted to respond to the search discovery request by returning addressing information to permit access to the database servers by the database client.

CA9-97-042

7

The computer program product may also include a software routine to establish a routine for the database client to respond to receive the addressing information provided by the database discovery manager module and to display the information to a user of the database client.

5

The computer program product may preferably also include a software routine for establishing a known discovery module for the database client for generating a database known discovery request to a selected database server system previously identified to determine detailed database server system information.

10

The computer program product of claim can include a software routine for establishing a database connection manager module associated with the selected database server adapted to respond to the known discovery request by packaging detailed database server system information relating to the selected database server system to permit access to databases on the system.

15

The computer program product may further include a software routine for establishing a module associated with the database client to respond to receive the detailed database server system information and to display the information to a user of the database client.

20

The computer program product may also include a software routine for enabling a graphical display interface module (GUI) to respond to user input to initiate the search discovery request and the known discovery request. The software routine module enables the GUI to be responsive to a user to initiate the search discovery request, it displays server system names returned, and is further responsive to the selection of a database server system displayed on the GUI by expanding the display of the selected database server system through initiation of a known discovery request. Detailed addressing information returned is displayed in treelike fashion, identifying the databases controlled by the selected database server system.

25

30

CA9-97-042

8

The GUI may be further responsive to selection of an identified database by auto-configuring for a database access request.

5 In yet another embodiment, the computer program product may include software routines for establishing the database server system with:

discovery manager and connection manager modules adapted to listen for discovery requests on the network, and comprised of:

10 a request receiver module to receive the discovery requests,
a request parser module to parse the requests for processing,
request service module for responding to the parsed requests, and in response to:

15 (a) the receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and
(b) the receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with the specified database server system and databases controlled by the database server instances; and

20 a communications response module adapted to transmit the packaged database server system addressing information and the packaged detailed database server system information to any database client requesting the information.

Brief Description of the Drawings

25 The invention is described herein together with the drawings of which a brief description follows:

Fig. 1 depicts an information handling client server network having a database server in accordance with the prior art;

Fig. 2 depicts an information handling client server network having a database server, constructed in accordance with the present invention herein;

CA9-97-042

9

Fig. 3 depicts a method of search discovery in accordance with the present invention using a broadcast mechanism;

Fig. 4 depicts a method of known discovery in accordance with the present invention in which detailed database server system information is requested and returned to the database client;

Fig. 5 depicts a detailed flow chart of the broadcast method of search discovery;

Fig. 6 depicts a detailed flow chart of the file server method of search discovery;

Fig. 7 depicts a detailed flow chart of known discovery, in which either the search discovery method or an user specifies enough communications information to directly contact an administration server on the database server system.

Detailed Description of a Preferred Embodiment of the Invention

Referring to Fig. 1, the client server database network 1 includes a database client 2, and database server systems 3, linked by a communications link 10. The database server system A, in this example includes a number of database server instances 7, database server instance AA, and database server instance AB. Each database server instance contains its own databases and manages its own database resources. For example, in order for a user of database client 2 to be able to access the databases in database server instance AA, a node directory entry 4, and a database directory entry 5 must be created before a client-server connection can be initiated. The information necessary for the construction of catalogs may be difficult for a user of a database client to obtain under the prior art.

Fig. 2 depicts one embodiment of a client server network in accordance with the invention herein. The network includes a database client 2, and one or more database server systems 3, which include database server instances 7; which in turn contain and control databases (not shown). The database server systems 3, each include a database administration server 11, which includes protocol-specific discovery managers 12, and protocol-specific connection managers 14. The database client includes

CA9-97-042

10

software to initiate requests to the discovery manager(s) 12 and connection manager(s) 14 of each database server system.

5 In general terms, when a user desires to access databases accessible by the network 1, the user instructs the database client, preferably by means of a simple graphical user interface (GUI) 15 to discover what databases are available to the database client, and to access user-selected databases.

10 The first step, shown in Fig. 3, is called the search discovery step. A protocol-specific communications method is used to find database server systems and return a set of database server system names and protocol-specific addressing information to the database client, preferably to be displayed on a GUI for the user. For example, the protocol-specific communications method can be implemented using a broadcast or multicast request, for protocols supporting this type of request,

15 Comments: As will be appreciated by those skilled in the art to which this invention relates, a broadcast request is one that goes to all network nodes which have a datagram receive listening on a particular port to which the request is sent. A multicast request is one that goes to all network nodes which are registered with a multicast
20 group and only the registered nodes will receive the request.

25 The second step, shown in Fig. 4, which may be called the known discovery step, involves allowing the user to select database server systems from those presented on the GUI, for the determination of additional information about the selected database server systems. Additional information about the database server system can be comprised of identification of which database server instances reside on the system, the protocols supported by each database server instance, the protocol-specific communications addressing information for those protocols supported, the databases available on the database server instance, security information required to access the

CA9-97-042

11

database server instance, and any other data relevant for client-server database functionality.

It is preferable that this information be presented by the GUI to the user so that the user may use the well known point and click method of GUI interaction familiar to the users of the OS/2 and Windows GUI systems to select the database server instances of interest and their databases. Preferably this information is presented in a tree form as illustrated in the Client GUI in Figure 4, allowing the user simple selection choices.

The first step in the procedure, shown in Fig. 3, the search discovery step, requests information which is sent back by the database server system discovery managers. It provides the database server system name and enough communications information to allow the database client to communicate with each database server system; so that when a user selects a database server system (step 2), sufficient communications information will be available for the database client to initiate a client-server connection to the selected database server system and retrieve the information necessary to expand the tree form into database server systems, database server instances, and databases.

During step 2, shown in Fig. 4, known discovery, sufficient database and communications information is returned to the database client about each database server instance on a user-selected database server system; so that when the user selects a database to access, the database client automatically catalogs node and database directories, for the user to connect to that database.

In the embodiment of the invention described herein, the administration server 11 in each database server system is adapted to respond to administration tasks, including search discovery and known discovery requests from database clients. As indicated above, each database server system can contain multiple database server instances and each database server instance can contain multiple databases, providing a

CA9-97-042

12

database server system capable of managing and accessing increasingly large amounts of data as required by modern society.

As will be appreciated by those skilled in network communications, different communications protocols exhibit different behaviors and have different conventions. With this in mind, search discovery is performed in accordance with the required protocol using a method that is appropriate for that protocol to find the database server systems on or accessible by the network. For example, using the TCP/IP protocol, a broadcast or multicast request could be issued to find database server systems. Using the NetBIOS protocol, a broadcast could be used to find the database server systems. Using the IPX/SPX protocol, IPX/SPX SAP (known as service advertising protocol) could be used to identify the accessible database server systems, or a NetWare file server attached to the database client could be searched for database server systems that have been previously registered at the file server.

In the case of database server systems using the DB2 database management system, the DB2 server systems would respond to the database client search discovery requests by returning their DB2 system names (for GUI identification purposes) and sufficient communications information about the administration server of the DB2 server system for the database client to attach to the administration server.

For each database server system expanded (graphically, in the case of a GUI) in a tree by the user making a selection on the GUI, a request is sent to the appropriate database server system for detailed server instance and database information. Requests are only sent for database server systems selected by the user.

In the embodiment of the invention described in Fig. 4, when a known discovery request is received by an administration server connection manager 14, the request is processed by calling the Generator 16 to generate the database server system information required by the database client. Options may be provided to a database

CA9-97-042

13

administrator to allow the administrator to configure the database server systems to hide specified database server instances or databases on the system. Server instances and databases that are 'marked' as hidden are not packaged by the generator in the information returned to the database client, so access to these hidden entities will not be accessible to the user from the GUI. This option provides additional security for confidential databases.

As will be readily appreciated from the above description the invention herein results in a simplification of the client configuration required by a user to connect to a desired database.

Fig. 3 illustrates an example of step 1, search discovery using a broadcast method. The client 2 generates a search discovery request to the network. Database server system 3, on the network, such as TEST in this illustration, has a search discovery manager 12 that picks up the search discovery request, and composes a response to be returned to the client, which includes the database server system named, TEST, and administration server communications information. The result of the search discovery request is a list of database server system names and associated communications information for each database server system found on the network.

The next step in the process may be appreciated by referring to Fig. 4 which depicts an implementation of the known discovery process. As a result of the search discovery process the user has been presented through the Client GUI 15 with database server system names, System TEST and System PROD. User selection of the database system named, TEST, returned from the search discovery request causes the database client to issue a known discovery request to the particular database server system selected, to return detailed information. The database server system, TEST, has a protocol-specific connection manager 14 that receives and parses the request, then calls the Generator 16 to package server system information to be returned to the client for display on the database client GUI 15. Now that information from database

CA9-97-042

14

server system, TEST, has been returned, the TEST system on the GUI expands to show all server instances accessible to the database client 2. If the user selects a server instance, it in turn is expanded to display all databases available on that server instance. The information is displayed in a tree format 17. The user of the database client can select one of the databases to have the database client auto-configure a connection to the selected database.

The invention herein is applicable to networks employing different protocols for communications.

For instance, networks using either NetBIOS or TCP/IP could use the Broadcast Flow step 1 depicted in Fig. 5 for search discovery and step 2- Directed Flow depicted in Fig. 7 for Known Discovery. IPX/SPX could use the step 1 - File Server Flow depicted in Fig. 6 for Search Discovery and step 2 - Directed Flow depicted in Fig. 7 for Known Discovery.

[1] Referring to Fig. 5, during DB2 server system boot up, the administration server is started. The administration server brings up a discovery manager and connection manager (see Fig. 7) for each protocol supported and selected,

A NetBIOS discovery listener will listen on a receive broadcast datagram call.

A TCP/IP discovery listener will open up a datagram socket, bind the socket to accept datagram messages from any address that specifies the listener port, set the socket to listen for broadcasts, and listen on a recvfrom call.

A DB2 TCP/IP listener can listen on a port (e.g. 3033/tcp) and accept connections while a discovery listener is listening on the same port (e.g. 3033/udp) and receive broadcasts.

CA9-97-042

15

NetBIOS broadcasts are only received by names that have a receive broadcast datagram call outstanding (i.e.. only NetBIOS discovery listeners). Since a receive broadcast datagram call receives all broadcasts, the NetBIOS discovery parser needs to check the buffer header to ensure that this is actually a discovery broadcast.

5

[2] When a 'discover' request is issued from the GUI, the Interpreter creates a new process and then calls the internal DB2 discover API to service the request asynchronously.

10 Discovery calls sqlccdiscoverinit to perform pre-broadcast tasks. Sqlccdiscoverinit loads the protocol specific library (it may not be loaded yet) then calls the protocol specific discovery initialization function (e.g. sqlccnbddiscoverinit, sqlcctcpdiscoverinit). It returns a discovery handle.

15 Sqlccnbddiscoverinit adds a free NetBIOS name of the form, DB2DSCVR < nname >. This name represents the address of the requesting client and is used to build the DB2GETADDR request.

20 Sqlcctcpdiscoverinit opens a datagram socket, binds the socket to no particular address, and issues a getsockname call to find the free port allocated to the socket by TCP/IP. The TCP/IP address and port number are used to build the DB2GETADDR request.

[3] Discovery then calls sqlccdiscover to broadcast the DB2GETADDR request.
25 Sqlccdiscover calls the protocol specific discover function (e.g.. sqlccnbddiscover, sqlcctcpdiscover).

Sqlccnbddiscover issues a send broadcast datagram call, then issues a sqlccdrecv to receive the reply. A configurable number of receive buffers are issued because
30 NetBIOS does not buffer data between receives.

CA9-97-042

16

Sqlcctcpdiscover sets the socket option for broadcasting and broadcasts the DB2GETADDR request then issues a sqlccdrecv to receive the reply.

- 5 Sqlccnbdiscover and sqlcctcpdiscover will reissue sqlccdrecv calls for a set period of time. The time period will be used to dictate a time when we think discover would have received replies from all systems.

The Discover time-out period is a configuration parameter that is user modifiable.

10

[4] The DB2 server system will receive the DB2GETADDR request via its discovery listener. The discovery parser will parse the message into a DB2GETADDR request.

15

The discovery server calls sqlccgetaddr to retrieve the server system address (for the administration server).

20

Sqlccgetaddr calls the protocol specific get address function (e.g.. sqlccnbgetaddr, sqlcctcpgetaddr). Sqlccnbgetaddr returns the nname of the administration server. Sqlcctcpgetaddr returns the IP address of the system and the port number used by the administration server.

The discovery listener uses the server system address to build the DB2RETADDR reply.

25

The discovery server then uses the client address sent in the DB2GETADDR request to issue a sqlccdsend call to send the DB2RETADDR reply.

[5] When the DB2RETADDR reply is received, each reply is processed synchronously.

CA9-97-042

17

The DB2RETADDR reply is parsed and the system address is saved in a list of known systems. Then, the server system addresses are used to directly retrieve DB2 server system information. See Figure 7 "Step 2 - Directed Flow" for the rest of the discovery flow, and a description of the figure in the following pages.

5

Step 1 - File Server Flow

The following text describes the Step 1 - File Server Flow depicted in Fig. 6:

10 (A) When a 'discover systems' request is issued from the GUI, the Interpreter creates a process and then calls the discover API to service the request asynchronously.

15 Sqlccdiscoverinit loads the protocol specific library (it may not be loaded yet) then calls the protocol specific discovery initialization function (e.g. sqlcctlldiscoverinit). It returns a discovery handle.

Sqlcctlldiscoverinit performs some initial setup tasks.

20 (B) Discovery then calls sqlccdiscover to find the addresses of DB2 server systems s (administration servers). Sqlccdiscover calls the protocol specific discover function (e.g.. sqlcctlldiscover).

25 Sqlcctlldiscover looks for an attached file server. If none is found, the user must specify (via the GUI) a list of one or more file servers to be used. This list is passed to Discover. Even if an attached file server is found, Discover will still use all file servers specified in the list. However, the user specified file server list is optional if an attached file server exists.

30 For each file server, the client attaches to the file server (if not already attached), then scans the objects in the bindery for objects that have the registered administration

CA9-97-042

18

server socket number, 87A2, and the DB2 object type, 0x062B. (This is an object type registered with Novell that is specifically for DB2.) All matches to the criteria are DB2 server system addresses (for the administration server). These addresses are saved in a list of known systems. The server system addresses are used to directly retrieve the DB2 server system information. See Fig. 7 "Step 2 - Directed Flow" for the rest of the discovery flow, and a description of the figure in the following pages.

The administration server addresses located on the file server were originally placed there when the user issued a REGISTER command/API. REGISTERing is a step required on the administration server and all DB2 server instances that are using IPX/SPX file server addressing.

The following text describes the Step 2 - Directed Flow depicted in Fig. 7.

After performing step 1 in discovery SEARCH mode, the client has retrieved enough server system address information to directly contact the administration server on each system. The scenario continues in Fig. 7 with the Step 2 - Directed Flow.

In discovery KNOWN mode, the user specifies enough administration server communications information (via the GUI) for discovery to directly contact the administration server, for each system specified. In discovery KNOWN mode, the scenario starts with the Step 2 - Directed Flow.

Note: Since it is possible for the known DB2 system list to contain duplicate systems (e.g., when discovery on a system supports more than one discovery protocol), duplicates will be removed.

[6] For each address in the list of known DB2 systems, discovery issues an unauthenticated attach API call. This forms an unauthenticated (i.e., no userid/password required) connection to the DB2 administration server.

CA9-97-042

19

[7] Discovery then builds a DB2 GETINFOSZ request with no data and issues a sqlccsend call to send the request.

5 [8] The agent receives the Discover service message and calls Discovery to service the request.

[9] Discovery recognizes the GETINFOSZ request and calls the Generator to package the DB2 server system information and return the size.

10 [10] Discovery builds a RETINFOSZ reply, packaging the information size returned by the Generator, and sends the reply to the client.

[11] The RETINFOSZ reply is received at the client, parsed, and the data is byte-reversed, if necessary.

15

[12] The system information size returned is used to build a GETINFO request.

[13] The GETINFO request is received by Discovery on the server and parsed.

20 [14] Discovery builds a RETINFO reply, packaging the information returned by the Generator, and sends the reply to the client.

[15] The RETINFO reply is received at the client and parsed.

25 [16] Discovery issues a detach API call to cleanup the connection to the DB2 system server.

[17] The data returned is passed to the Interpreter.

CA9-97-042

20

Functions depicted in figures 5, 6, and 7 are detailed in the following pseudo code which is included herein in the Appendix which follows below. Software coding of the invention may be implemented in the database clients and database servers of a client server network using this pseudo code as a base for developing software adapted for a particular software product that runs on particular network hardware. The foregoing has served to describe the structure and operation of the present invention. It will be evident to those skilled in the art that a number of variants of the structure and functionality are substantially equivalent and therefore, are considered within the scope of the present invention.

5

CA9-97-042

21

Appendix**Component SQC**

5

```

1  sqlccmn.h
1  /*=====*/
1  /*          Structure for discover initialization          */
1  /*=====*/

1  typedef struct sqlcc_dinit_struct
1  {
1      unsigned char    ucType;        /* constant SQL_PROTOCOL_xxxxx */
1      void             *protoinfo;    /* protocol specific information */
1  } SQLCC_DINITSTRUCT_T;

1  /*=====*/
1  /*          Structure for DISCOVER request                */
1  /*=====*/

1  typedef struct sqlcc_discover
1  {
1      struct sqlcdaddr *addr;         /* address list */
1      void *           pDscvrProto;
1  } SQLCC_DISCOVER_T;

1  /*=====*/
1  /*          Structure for connectless SEND request        */
1  /*=====*/

1  typedef struct sqlcc_dsend
1  {
1      struct sqlcdaddr *addr;         /* address to send to */
1      char             *data;
1      unsigned long    datalen;
1      void *           pDSendProto;
1  } SQLCC_DSEND_T;

1  /*=====*/
1  /*          Structure for connectionless RECEIVE request  */
1  /*=====*/

1  typedef struct sqlcc_drecv
1  {
1      struct sqlcdaddr *addr;         /* address data is received from */
1      char             *data;
1      unsigned long    datalen;
1      void *           pDRecvProto;
1  } SQLCC_DRECV_T;

```

CA9-97-042

22

```

1  sqlccmn.h (cont'd)
1  /*=====*/
1  /*          Structure for GET ADDRESS request          */
1  /*=====*/

1  typedef struct sqlcc_getaddr
1  {
1      void *          pGetAddrProto;
1  } SQLCC_GETADDR_T;

1  /*=====*/
1  /*          Handle to discover functions and structures          */
1  /*=====*/

1  typedef struct sqlcc_dscvrhandle
1  {
1      short (*dscvrf)
1          (struct sqlcc_dscvrhandle *, SQLCC_COND_T *);
1      short (*dsendf)
1          (struct sqlcc_dscvrhandle *, SQLCC_COND_T *);
1      short (*drecvf)
1          (struct sqlcc_dscvrhandle *, SQLCC_COND_T *);
1      short (*getaddrf)
1          (struct sqlcc_dscvrhandle *, SQLCC_COND_T *);
1      short (*dscvrtermf)
1          (struct sqlcc_dscvrhandle *, SQLCC_COND_T *);

1      unsigned char    ucType;          /* constant SQL_PROTOCOL_xxxxx */

1      SQLCC_DSCVR_T     dDscvr;
1      SQLCC_DSEND_T     dDSend;
1      SQLCC_DRECV_T     dDRecv;
1      SQLCC_GETADDR_T   dGetAddr;
1  #ifndef SQL_STATIC_LINK
1      sqlModuleHandle    modHandle;
1      /* C++ */char       lib_loaded;
1  #endif

1  } SQLCC_DSCVRHANDLE_T;

```

CA9-97-042

23

```

1  sqlcccmn.h (cont'd)
1  /*****
1  /*      Function prototypes      */
1  /*****
1
1  short sqlccdiscoverinit(
1          SQLCC_DINITSTRUCT_T      *pDscvrInit,
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );
1
1  short sqlccdiscover(
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );
1
1  short sqlccdsend(
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );
1
1  short sqlccdrecv(
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );
1
1  short sqlccgetaddr(
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );
1
1  short sqlccdiscoverterm (
1          SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1          SQLCC_COND_T             *pCondition
1      );

```

CA9-97-042

24

```

1  sqlccmn.C
1  /*****
1  *
1  *          DISCOVER INITIALIZATION
1  *****/
1
1  short sqlccdiscoverinit(
1      SQLCC_DINITSTRUCT_T      *pDscvrInit,
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - validate the protocol type
1      - allocate memory for the discovery handle and initialize
1      - copy protocol type to the discovery handle
1      - load protocol specific library, if required
1      - call protocol specific discover initialization
1  }
1
1  /*****
1  *
1  *          DISCOVER
1  *****/
1
1  short sqlccdiscover(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - call protocol specific discover function
1  }
1
1  /*****
1  *
1  *          CONNECTIONLESS SEND
1  *****/
1
1  short sqlccdsend(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - call protocol specific connectionless send function
1  }

```

CA9-97-042

25

```

1  sqlccmn.C (cont'd)
1  /*****
1  *
1  *      CONNECTIONLESS RECEIVE
1  *      *****/
1
1  short sqlccdrecv(
1      SQLCC_DSCVRHANDLE_T    *pDscvrHandle,
1      SQLCC_COND_T          *pCondition
1  )
1  {
1      - call protocol specific connectionless receive function
1  }
1
1  /*****
1  *
1  *      GET ADDRESS
1  *      *****/
1
1  short sqlccgetaddr(
1      SQLCC_DSCVRHANDLE_T    *pDscvrHandle,
1      SQLCC_COND_T          *pCondition
1  )
1  {
1      - call protocol specific get address function
1  }
1
1  /*****
1  *
1  *      DISCOVER TERMINATION
1  *      *****/
1
1  short sqlccdiscoverterm (
1      SQLCC_DSCVRHANDLE_T    *pDscvrHandle,
1      SQLCC_COND_T          *pCondition
1  )
1  {
1      - call protocol specific discover termination function
1      - unload library, if necessary (platform specific)
1      - free discovery handle
1  }

```

[illegible]

CA9-97-042

27

```

1  sqlcctsv.C
1  /*****
1  *          TCP/IP Discovery Manager
1  *****/
1
1  short sqlcctcpdscvrmgr ( SQLCC_CONNMGR_T *pConnMgr,
1                          struct sqlf_kcfd *pDbmConfig )
1  {
1      - initialize connection manager wait/post area
1      - spawn the discovery manager EDU that will listen for discovery requests
1      - wait for status from discovery manager EDU
1      - check postcode from the discovery manager EDU
1  }
1
1  /*****
1  *          TCP/IP Discovery Manager Listener
1  *****/
1
1  void sqlcctcpdscvrmgr_child ( char          *pEDUArgs,
1                                unsigned long ulEDUArgsSz )
1  {
1      - call getservbyname to get the TCP/IP port #
1      - call sqlccddiscoverinit (opens datagram socket, binds to listener port,
1                                and sets socket to send/receive broadcasts)
1      - post RC to discovery manager
1      - do forever,
1          - listen on a recvfrom for discovery requests
1          - if EINTR received,
1              - check termination flag
1              - call sqlccddiscoverterm
1              - cleanup
1          - if data received,
1              - parse data and check for a valid DB2 discovery request
1              - if DB2GETADDR request,
1                  - call sqlccgetaddr to get TCP/IP address info.
1                  - build DB2RETADDR request
1                  - byte-swap DB2GETADDR data as required and use client address
1                    to issue a sqlccdsend call to send the DB2RETADDR reply
1              - if invalid request, ignore the request
1  }

```

CA9-97-042

28

```

1  sqlcctcp.h
1  /*=====*/
1  /*      Structure for DB2GETADDR request      */
1  /*=====*/

1  typedef struct sqlcc_tcp_db2getaddr
1  {
1      char header[14] = "DB2GETADDR";
1      unsigned short seqno;
1      unsigned short relno;
1      struct in_addr addr;
1      unsigned short port;
1  } SQLCC_TCP_DB2GETADDR;

1  /*=====*/
1  /*      Structure for TCP/IP DISCOVER request      */
1  /*=====*/

1  typedef struct sqlcc_tcp_discover
1  {
1      SQLCC_SOCKET sock;
1      unsigned short port;
1  } SQLCC_TCPDSCVR_T;

1  /*=====*/
1  /*      Structure for TCP/IP connectionless SEND request      */
1  /*=====*/

1  typedef struct sqlcc_tcp_dsend
1  {
1      char *data;
1      unsigned long datalen;
1      struct sockaddr *addr;
1  } SQLCC_TCPDSEND_T;

1  /*=====*/
1  /*      Structure for TCP/IP connectionless RECEIVE request      */
1  /*=====*/

1  typedef struct sqlcc_tcp_drecv
1  {
1      char *data;
1      unsigned long datalen;
1      struct sockaddr *addr;
1  } SQLCC_TCPDRECV_T;

```

CA9-97-042

29

```

1  sqlcctcp.h (cont'd)
1  /*=====*/
1  /*      Structure for TCP/IP GETADDR request      */
1  /*=====*/
1
1  typedef struct sqlcc_tcp_getaddr
1  {
1      char ipaddr[255+1];
1      unsigned short port;
1  } SQLCC_TCPGETADDR_T;
1
1  /*=====*/
1  /*      Function Prototypes      */
1  /*=====*/
1
1  short sqlcctcpdiscoverinit(
1      SQLCC_DINITSTRUCT_T      *pDscvrInit,
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );
1
1  short sqlcctcpsend(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );
1
1  short sqlcctcpdrecv(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );
1
1  short sqlcctcpgetaddr(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );
1
1  short sqlcctcpdiscoverterm(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );

```

CA9-97-042

30

```

1  sqlcctcp.C
1  /*****
1  *
1  *      TCP/IP DISCOVER INITIALIZATION
1  *****/
1
1  short sqlcctcpdiscoverinit(
1      SQLCC_DINITSTRUCT_T      *pDscvrInit,
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - perform TCP/IP socket initialization, if required
1      - set up TCP/IP discover function pointers
1      - allocate TCP/IP specific substructures and initialize
1      - open a datagram socket
1      - bind socket to no particular address, but to port specified
1      - call getsockname to find the free port allocated to the socket by TCP/IP
1      and save it
1      - set socket option for broadcasting
1  }
1
1  /*****
1  *
1  *      TCP/IP DISCOVER
1  *****/
1
1  short sqlcctcpdiscover(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - use the TCP/IP address and port # to build the DB2GETADDR request
1      - broadcast DB2GETADDR request
1      - for a particular time period,
1          - call sqlccdrecv to receive DB2RETADDR reply
1          - parse the DB2RETADDR reply and save the admin. server address
1  }
1
1  Support for a DB2DSCVR.TCP file that allows broadcasting to a broader range
1  of workstations may be supported later.
1
1  /*****
1  *
1  *      TCP/IP CONNECTIONLESS SEND
1  *****/
1
1  short sqlcctcpdsend(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - issue a TCP/IP sendto call
1  }

```

CA9-97-042

31

```

1  sqlcctcp.C (cont'd)
1  /*****
1  *
1  *      TCP/IP CONNECTIONLESS RECEIVE
1  *****/
1
1  short sqlcctcpdrecv(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - issue a TCP/IP recvfrom call
1  }
1
1  /*****
1  *
1  *      TCP/IP GET ADDRESS
1  *****/
1
1  short sqlcctcpgetaddr(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - call gethostid to get the 32-bit host id
1      - call inet_ntoa to get a string in dotted-decimal notation
1  }
1
1  /*****
1  *
1  *      TCP/IP DISCOVER TERMINATION
1  *****/
1
1  short sqlcctcpdiscoverterm(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  )
1  {
1      - perform TCP/IP socket cleanup, if necessary (platform specific)
1      - deallocate TCP/IP specific substructures
1  }

```

CA9-97-042

32

```

1  sqlcctl.h
1  /*=====*/
1  /*      Structure file server list      */
1  /*=====*/

1  typedef struct sqlcc_tli_fslist
1  {
1      char          fsname[48+1];
1      struct sqlcc_tli_fslist *nextfs;
1  } SQLCC_TLI_FSLIST_T;

1  /*=====*/
1  /*      Structure for DISCOVER request      */
1  /*=====*/

1  typedef struct sqlcc_tli_discover
1  {
1      SQLCC_TLI_FSLIST_T  fslist;
1  } SQLCC_TLIDISCOVER_T;

1  /*=====*/
1  /*      Structure for GETADDR request      */
1  /*=====*/

1  typedef struct sqlcc_tli_getaddr
1  {
1      char readable_net[8+1];
1      char readable_node[12+1];
1  } SQLCC_TLIGETADDR_T;

1  /*=====*/
1  /*  COMMON PROTOTYPES                      */
1  /*=====*/
1  short sqlcctliddiscoverinit(
1      SQLCC_DINITSTRUCT_T      *pDscvrInit,
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );

1  short sqlcctliddiscover(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );

1  short sqlcctligetaddr(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );

1  short sqlcctliddiscoverterm(
1      SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1      SQLCC_COND_T             *pCondition
1  );

```

CA9-97-042

33

```

1  sqlcctl.c
1  /*****
1  *
1  *          IPX/SPX DISCOVER INITIALIZATION
1  *****/
1
1  short sqlcctliddiscoverinit (SQLCC_DINITSTRUCT_T      *pDscvrInit,
1                               SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1                               SQLCC_COND_T              *pCondition)
1  {
1      - set up IPX/SPX discover function pointers, sqlcctlidsend and
1      sqlcctlidrecv are NULL function pointers
1      - allocate IPX/SPX specific substructures and initialize
1      - if an IPX/SPX file server list was passed in,
1      copy it to the discovery handle
1  }
1
1  /*****
1  *
1  *          IPX/SPX DISCOVER
1  *****/
1
1  short sqlcctliddiscover (SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1                          SQLCC_COND_T              *pCondition)
1  {
1      - call NWGetConnectionList/NWGetServerConnIDList
1      to get a list of attached/connected file servers
1      - for each connection handle in the list,
1      - call NWGetFileName/NWGetServerPlatformName
1      to get the file server name
1      - compile a list of file server names including the ones passed
1      to Discover, remove duplicates
1      - for each file server,
1      - call NWAttachToFileServer/NWAttachToServerPlatform
1      to attach to file server, if not already attached
1      - call NWScanObject to search for all objects with the DB2 object
1      type, 0x062B
1      - for each DB2 object found,
1      - call NWScanProperty to retrieve object properties
1      - call NWReadPropertyValue/NWScanPropertyValue to read the
1      property value (ie. IPX internetwork address)
1      - save addresses that have the registered admin. instance socket #
1      - call NWDetachFromFileServer/NWDetachFromServerPlatform if not
1      previously attached (ie. reset back to original state)
1  }

```

CA9-97-042

34

```

1  sqlcctl.c (cont'd)
1  /*****
1  *
1  *      IPX/SPX GET ADDRESS
1  *****/
1
1  short sqlcctligetaddr (SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1                      SQLCC_COND_T              *pCondition)
1  {
1      - call t_open to open transport endpoint
1      - call t_alloc to allocate memory for t_bind struct
1      - call t_bind to get net and node addresses
1      - call sqlcctliaaddrtox to convert addresses to readable form
1  }
1
1  /*****
1  *
1  *      IPX/SPX DISCOVER TERMINATION
1  *****/
1
1  short sqlcctlidiscoverterm (SQLCC_DSCVRHANDLE_T      *pDscvrHandle,
1                          SQLCC_COND_T              *pCondition)
1  {
1      - deallocate IPX/SPX specific substructures
1  }

```

Note: The behaviour of NetBIOS is similar to TCP/IP and will not be explicitly documented.

```

1  sqledscv.h
1  /*****
1  ** Discover types
1  *****/
1
1  #define SQLE_DSCVR_DISABLED  0
1  #define SQLE_DSCVR_SEARCH    1
1  #define SQLE_DSCVR_KNOWN     2
1
1  /*****
1  ** Discover protocols
1  *****/
1
1  #define  SQLE_DSCVR_IPXSPX      0x0001
1  #define  SQLE_DSCVR_NETBIOS    0x0002
1  #define  SQLE_DSCVR_TCPIP      0x0004
1
1  /*****
1  ** Discover Info. Structure
1  *****/
1
1  struct sqledscvrinfo
1  {
1      void *protoinfo;          /* protocol specific information */
1      struct sqledaddr *addrlst; /* list of admin. server addresses */
1      SQLO_QUE_DESC *qhandle;   /* queue handle to return discovered info. */
1  };
1
1  /*****
1  ** Discover Workstation Address List Structure
1  *****/
1
1  struct sqledaddr {
1      union {
1          struct {              /* well-known socket will be used */
1              char readable_net[8+1];
1              char readable_node[12+1];
1          } ipx;
1          struct {
1              char name[16+1];
1          } nb;
1          struct {
1              char computername[15+1];
1              char instancename[8+1];
1          } np;
1          struct {
1              char ipaddr[255+1]; /* well-known port will be used */
1          } tcp;
1      } protocol;
1      struct sqledaddr *nextaddr;
1  };

```

CA9-97-042

36

```

1  sqledscv.h (cont'd)
1  /*****
1  ** Discover GET INFO. SIZE request
1  *****/

1  struct sqle_db2getinfosz
1  {
1      unsigned long infosz;
1  };

1  /*****
1  ** Discover GET INFO. request
1  *****/

1  struct sqle_db2getinfo
1  {
1      void *info;          /* flattened format of wkstn, instance, */
1                          /* and DB info. - variable length */
1  };

1  /*****
1  ** Discover Error Codes
1  *****/

1  #define SQLE_ERR_BASE          0
1  #define SQLE_ERR_DISCOVER_DISABLED (SQLE_ERR_BASE + 1)
1  #define SQLE_ERR_INV_DISCOVER_TYPE (SQLE_ERR_BASE + 2)
1  #define SQLE_ERR_INV_QUEUE_HANDLE (SQLE_ERR_BASE + 3)
1  #define SQLE_ERR_INV_ADDR_LIST (SQLE_ERR_BASE + 4)
1  #define SQLC_ERR_INV_DISCOVER_COMM (SQLE_ERR_BASE + 5)

1  /*****
1  ** Function Prototypes
1  *****/

1  short sqlediscover (struct sqlcdscvrinfo *discoverInfo);

1  short sqledscvr_getaddr (unsigned short discoverComm,
1                          struct sqlcdscvrinfo *discoverInfo);

1  short sqledscvr_getinfo (struct sqlcdscvrinfo *discoverInfo,
1                          struct sqlcdaddr *addr);

```

CA9-97-042

37

```

1  sqlcdscv.c
1  /*****
1  *
1  *          DISCOVER DB2 WORKSTATION INFO.
1  *****/
1
1  short sqlcddiscover (struct sqlcdscvrinfo *discoverInfo)
1  {
1      - if no queue handle, rc = SQLE_ERR_INV_QUEUE_HANDLE
1      - get DBM config. DISCOVER and DISCOVER_COMM values
1      - if type = SQLE_DSCVR_DISABLED, rc = SQLE_ERR_DISCOVER_DISABLED
1      - if type = SQLE_DSCVR_SEARCH,
1          - if DISCOVER_COMM is not specified, rc = SQLE_ERR_INV_DISCOVER_COMM
1          - call sqlcdscvr_getaddr
1          - for each server workstation found, call sqlcdscvr_getinfo
1      - if type = SQLE_DSCVR_KNOWN,
1          - if admin. server address list is empty, rc = SQLE_ERR_INV_ADDR_LIST
1          - for each server workstation in list, call sqlcdscvr_getinfo
1      - if type is invalid, rc = SQLE_ERR_INV_DISCOVER_TYPE
1  }
1
1  /*****
1  *
1  *          FIND WORKSTATION ADDRESSES
1  *****/
1
1  short sqlcdscvr_getaddr (unsigned short    discoverComm,
1                          struct sqlcdscvrinfo *discoverInfo)
1  {
1      - for each discover protocol,
1          - sqlccdiscoverinit
1          - sqlccdiscover
1          - sqlccdiscoverterm
1          - byte reverse address fields as required
1      - return list of admin. server addresses
1  }
1
1  /*****
1  *
1  *          GET WORKSTATION INFORMATION
1  *****/
1
1  short sqlcdscvr_getinfo (struct sqlcdscvrinfo *discoverInfo,
1                          struct sqlcdaddr    *addr)
1  {
1      - catalog node
1      - unauthenticated attach to remote admin. server
1      - build and send a DB2RA DISCOVER service, GETINFOSZ request
1        to the server, receive the info. size in the reply
1      - byte reverse info. size, if necessary
1      - use the info. size to build a DB2RA DISCOVER service, GETINFO
1        request to the server, receive workstation info. in the reply
1      - put data on the queue, if queue full, sleep for 3 secs. then retry
1  }

```

CA9-97-042

38

```
1  sqledsvs.C
1  /*****
1  *          SERVER DISCOVER AGENT FUNCTION
1  *****/
1  unsigned long sqledscvr_agent (SQLE_DB2RA_T *pDb2ra)
1  {
1      - parse DISCOVER request type
1      - if request = GETINFOSZ,
1          - call Generator to get the workstation info. and info. size
1          - flatten info. generated for sending via the network
1          - build the RETINFOSZ reply and send
1      - if request = DB2GETINFO,
1          - build the RETINFO reply and send
1      - free workstation info. buffer
1  }
```

CA9-97-042

CLAIMS

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

5

1. A client server information handling network having one or more database server systems controlling at least one database, at least one database client and a communications link between said servers and said client; said network including:
discovery means for said database client to identify database servers accessible from said
10 network, said discovery means comprising:

search discovery means associated with said database client for generating a database discovery inquiry comprising a database search discovery request to identify said database server systems;

15

database discovery manager means associated with said database servers, said
database discovery manager means for responding to said database search discovery
request by returning addressing information to permit selection of said database servers
by said database client.

20

2. The network of claim 1 including means associated with said client to respond to
receiving said addressing information provided by said database discovery manager and
to display said information to a user of said client.

3. The network of claim 2 including:

25

known discovery means associated with said client for generating a database known
discovery request to a selected database server system previously identified to determine
detailed database server system information.

4. The network of claim 3 wherein database connection manager means associated
with said selected database server for responding to said known discovery request by

CA9-97-042

packaging detailed database server system information relating to said selected database server system to permit access to databases on said system.

5. The network of claim 4 including means associated with said client to respond to receiving said detailed database server system information and to display said information to a user of said client.

6. The network of claim 5 including a graphical display interface means (GUI) for displaying server system information to a user, said GUI being responsive to user input to initiate said database search discovery request and said known discovery request.

7. The network of claim 6 wherein said GUI is responsive to selection by a user to initiate said search discovery request, displays server system names returned, and is further responsive to selection of a database server system displayed on said GUI to expand the display of said selected database server system by initiating the generation of a known search request and depicting the detailed database server system information returned in treelike fashion, identifying the databases controlled by said selected database server system.

8. The network of claim 7 wherein said GUI is further responsive to a pointer selection of an identified database by auto-configuring a connection to said identified database for a database access request.

9. The network of claim 8 wherein said database server system controls a plurality of database server instances, and wherein each of said database server instances controls at least one database.

10. The network of claim 9 wherein said database server system comprises:
discovery manager and connection manager means for listening for discovery

CA9-97-042

requests on said network, and are comprised of:

request receiver means for receiving said discovery requests,

request parser means for parsing said requests for processing,

request service means for responding to said parsed requests, and in response to:

- 5 (a) receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and
- (b) receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with said specified database server system and databases controlled by
- 10 said database server instances; and

communications response means for transmitting said packaged database server system addressing information and said packaged detailed database server system information to any database client requesting said information.

- 15 11. A method of identifying databases in a client server information handling network having one or more database server systems controlling at least one database, at least one database client and a communications link between said servers and said client, comprising:

generating a database discovery inquiry for said client comprising a database

20 search discovery request to identify said database server systems;

using database discovery manager means associated with said database servers to respond to said search discovery request by returning addressing information to permit access to said database servers by said database client.

- 25 12. The method of claim 11 including means associated with said client to respond to receiving said addressing information provided by said database discovery manager and to display said information to a user of said client.

13. The method of claim 12 including generating a database known discovery request

CA9-97-042

for said client to a selected database server system previously identified to determine detailed database server system information.

14. The method of claim 13 wherein said selected database server responds to said known discovery request by packaging detailed database server system information relating to said selected database server system to permit access to databases on said system.

15. The method of claim 14 wherein said client responds to receive said detailed database server system information and displays said information to a user of said client.

16. The method of claim 15 wherein a graphical display interface means (GUI) displays server system information to a user, and responds to user input to initiate said search discovery request and said known discovery request.

17. The method of claim 16 wherein said GUI responds to selection by a user, to initiate said search discovery request, displays server system names returned, and is further responsive to the selection of a database server system displayed on said GUI to expand the display of said selected database server system by initiating the generation of a known search request and depicting the detailed database server system returned in treelike fashion, identifying the databases controlled by said selected database server system.

18. The method of claim 17 wherein said GUI further responds to selection of an identified database by auto-configuring a connection to said identified database for a database access request.

19. The method of claim 18 wherein said database server system controls a plurality of database server instances, and wherein each of said database server instances controls at least one database.

CA9-97-042

20. The method of claim 19 wherein said database server system comprises:

listening for discovery requests on said network, receiving said discovery requests, parsing said requests for processing, responding to said parsed requests, and in response to:

5 (a) receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and

(b) receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with said specified database server system and databases controlled by
10 said database server instances;

and transmitting said packaged database server system addressing information and said packaged detailed database server system information to any database client requesting said information.

15 21. A computer program product for establishing a client server information handling network having at least one database server system controlling at least one database, at least one database client and a communications link between said servers and said client; said computer program product including a computer readable storage medium and software instructions stored on said storage medium for establishing said network, said
20 medium containing:

software instructions for directing a computer to establish discovery means for said database client for identifying database servers accessible with said network; said discovery means comprising:

25 search discovery means associated with said database client for generating a database discovery inquiry comprising a database search discovery request to identify said database server systems;

database discovery manager means associated with said database servers, said database discovery manager means for responding to said search discovery request by returning addressing information to permit access to said database servers by

CA9-97-042

said database client.

22. The computer program product of claim 21 including software instructions to establish means associated with said client to respond to receiving said addressing information provided by said database discovery manager and to display said information to a user of said client.

23. The computer program product of claim 22 including:

software instructions for establishing known discovery means associated with said client for generating a database known discovery request to a selected database server system previously identified to determine detailed database server system information.

24. The computer program product of claim 23 including software instructions for establishing database connection manager means associated with said selected database server adapted to respond to said known discovery request by packaging detailed database server system information relating to said selected database server system to permit access to databases on said system.

25. The computer program product of claim 24 including software instructions for establishing means associated with said client to respond to receiving said detailed database server system information and to display said information to a user of said client.

26. The computer program product of claim 25 including software instructions for enabling a graphical display interface means (GUI) to respond to user input to initiate said search discovery request and said known discovery request.

27. The computer program product of claim 26 wherein said software instructions enables said GUI to be responsive to selection by a user to initiate said search discovery request, displays server system names returned, and is further responsive to selection of

CA9-97-042

5 a database server system displayed on said GUI to expand the display of said selected database server system by initiating the generation of a known search request and depicting the detailed database server system information returned in treelike fashion, identifying the databases controlled by said selected database server system.

10 28. The computer program product of claim 27 wherein said software instructions enables said GUI to be further responsive to selection of an identified database by auto-configuring a connection to said identified database for a database access request.

15 29. The computer program product of claim 28 wherein said network established by said computer program product includes a database server system which controls a plurality of database server instances, and wherein each of said database server instances controls at least one database.

30. The computer program product of claim 29 comprises software instructions for establishing said database server system with:

discovery manager and connection manager means for listening for discovery requests on said network, and are comprised of:

20 request receiver means for receiving said discovery requests,

request parser means for parsing said requests for processing,

request service means for responding to said parsed requests, and in response to:

(a) receipt of a search discovery request obtains and packages database server system name and protocol - specific addressing information, and

25 (b) receipt of a known discovery request for a specified database server system, obtains and packages detailed information for any database server instances associated with said specified database server system and databases controlled by said database server instances; and

communications response means transmitting said packaged database server

CA9-97-042

system addressing information and said packaged detailed database server system information to any database client requesting said information.

FIG. 1

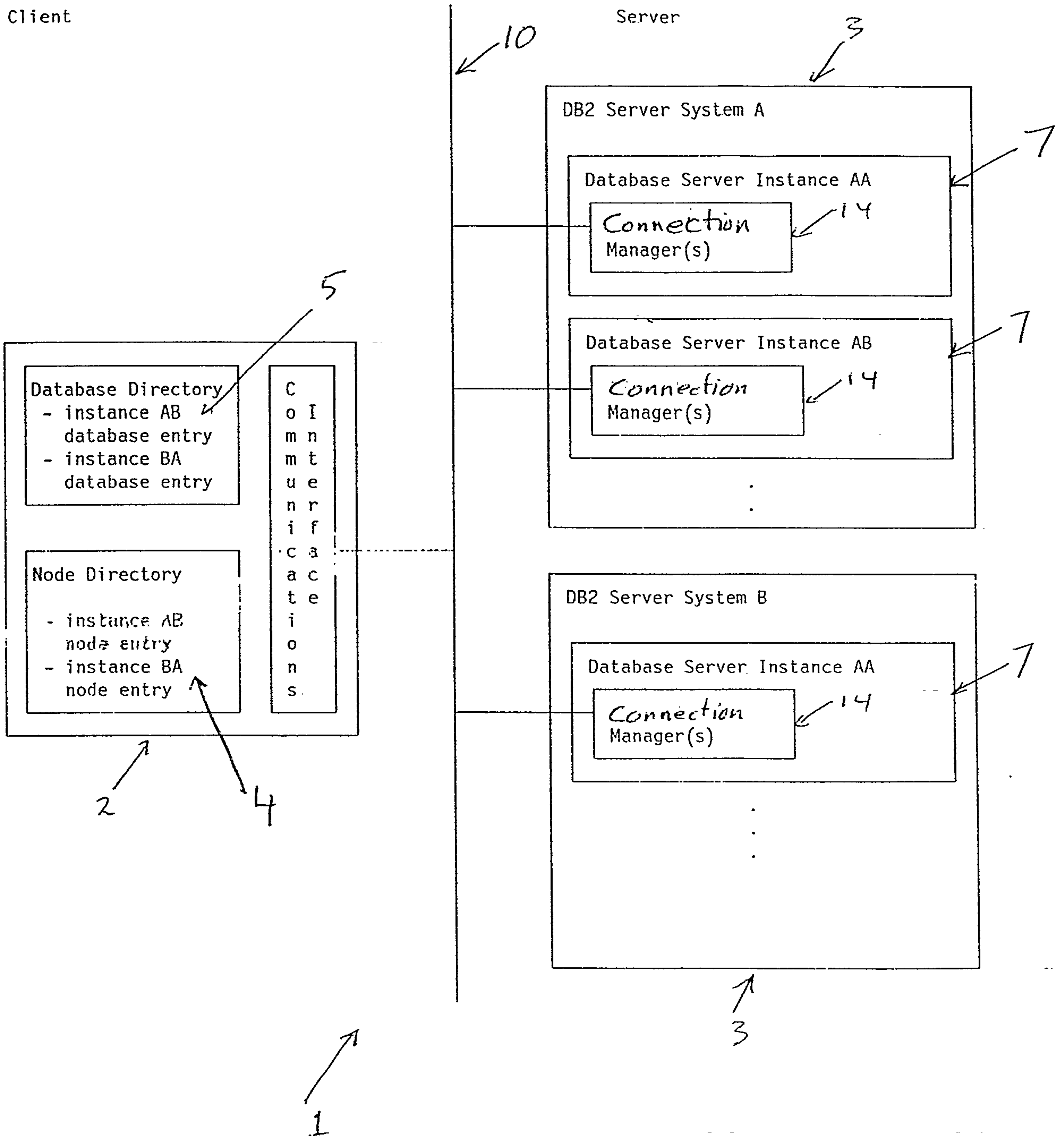


FIG. 2

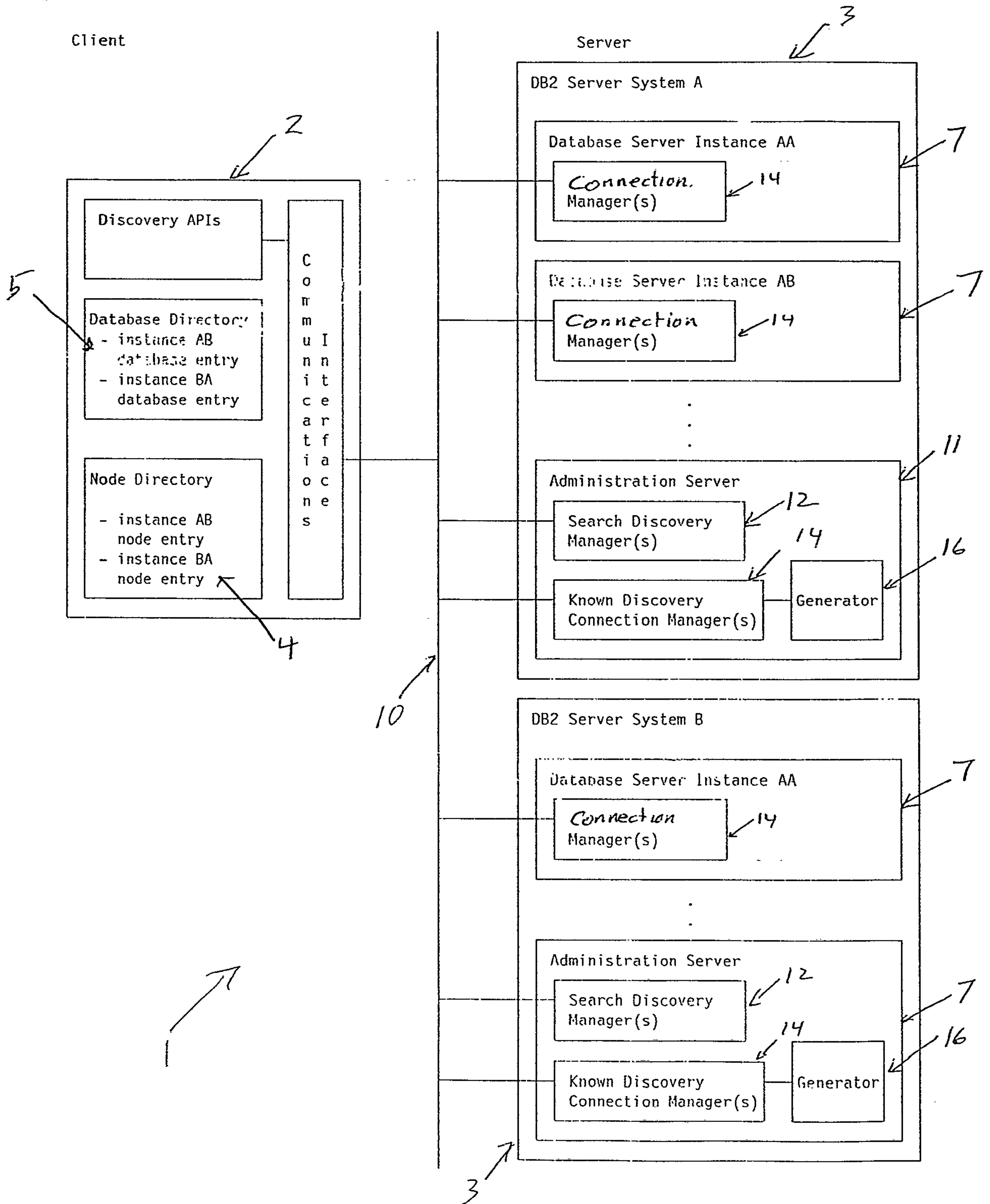


FIG. 3ClientServer

Search discovery request →
to network

Return DB2SYSTEM name & administration
server communications info. to caller

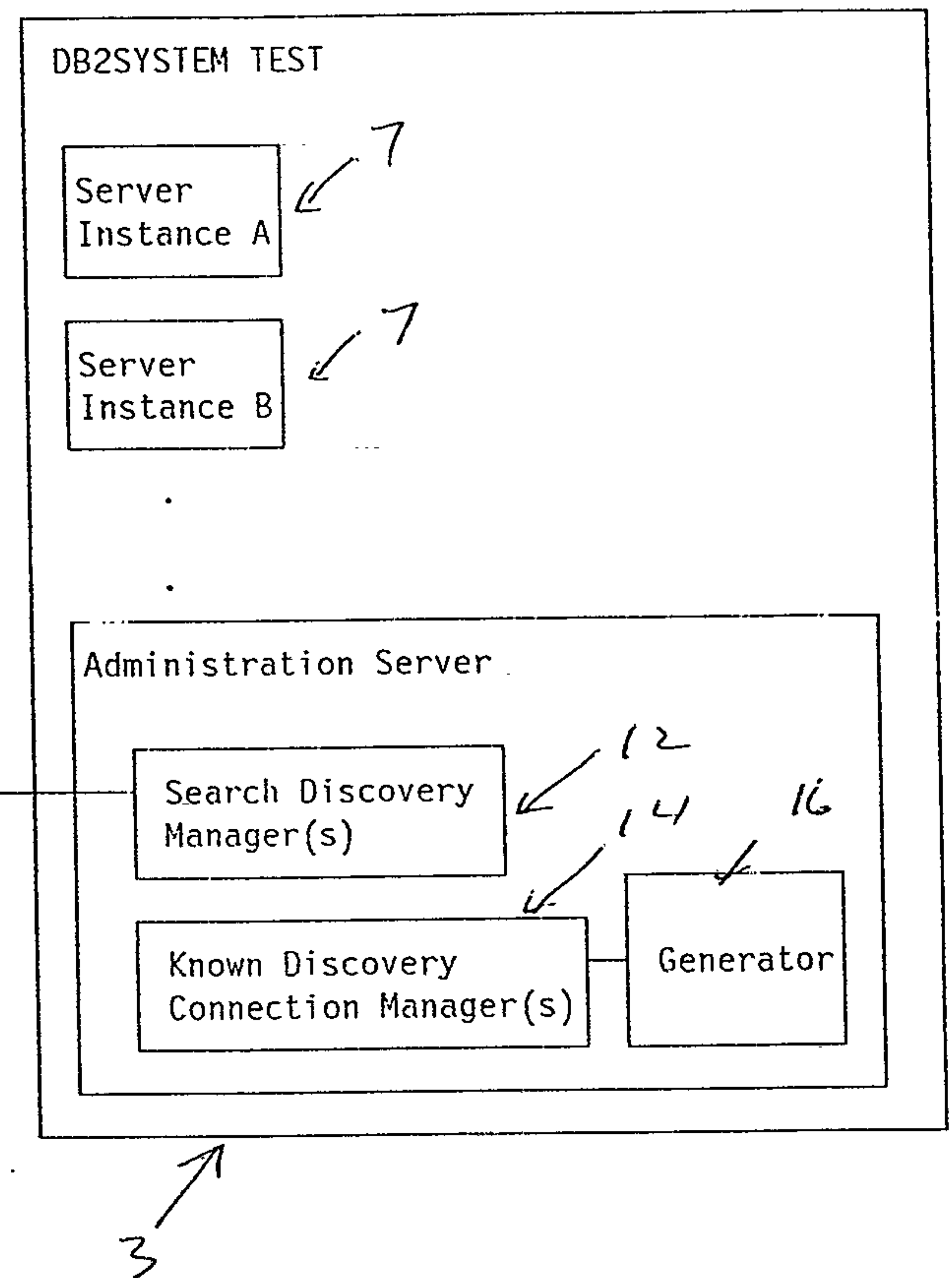


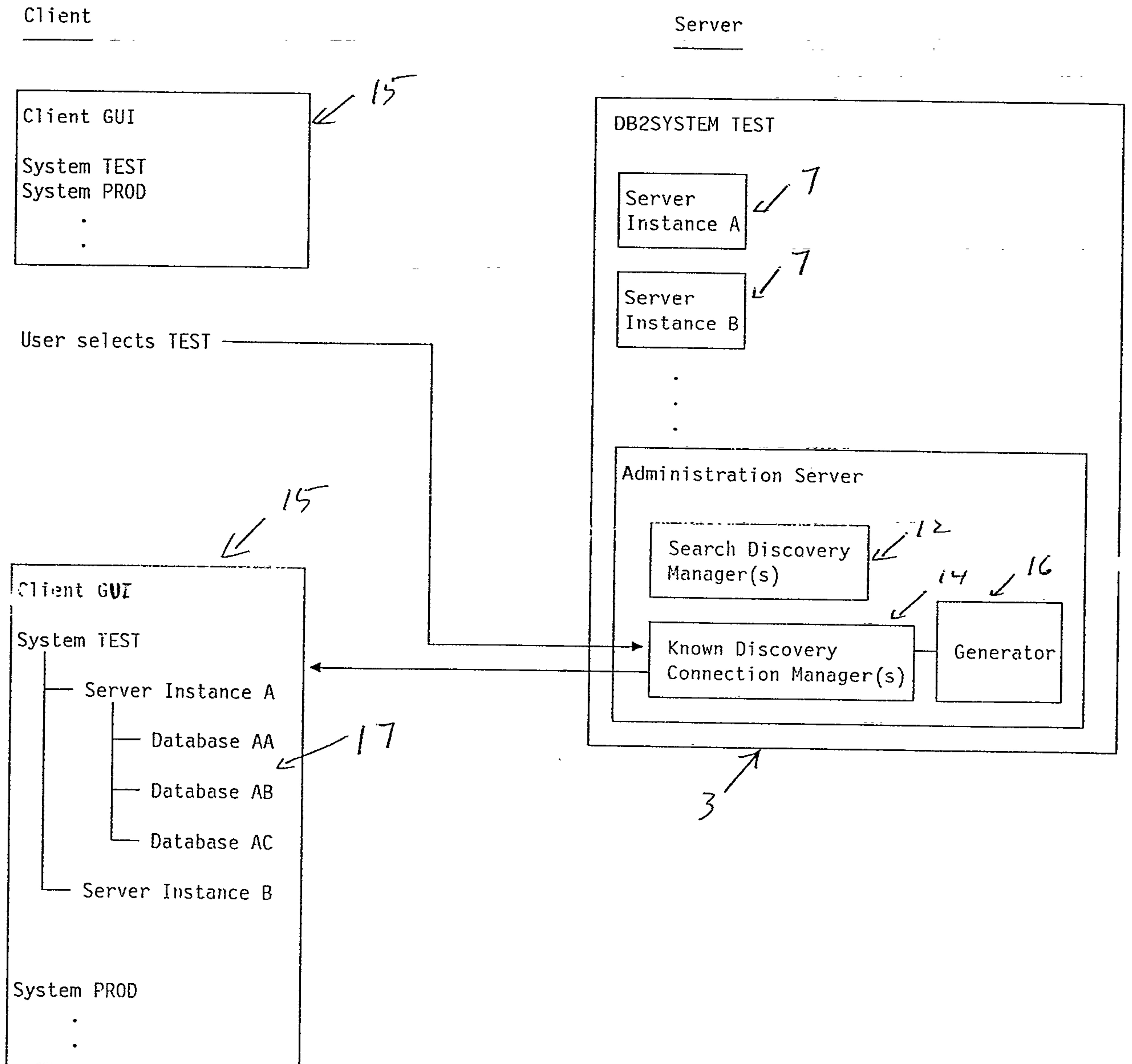
FIG. 4

FIG. 5

Step 1 - Broadcast Flow

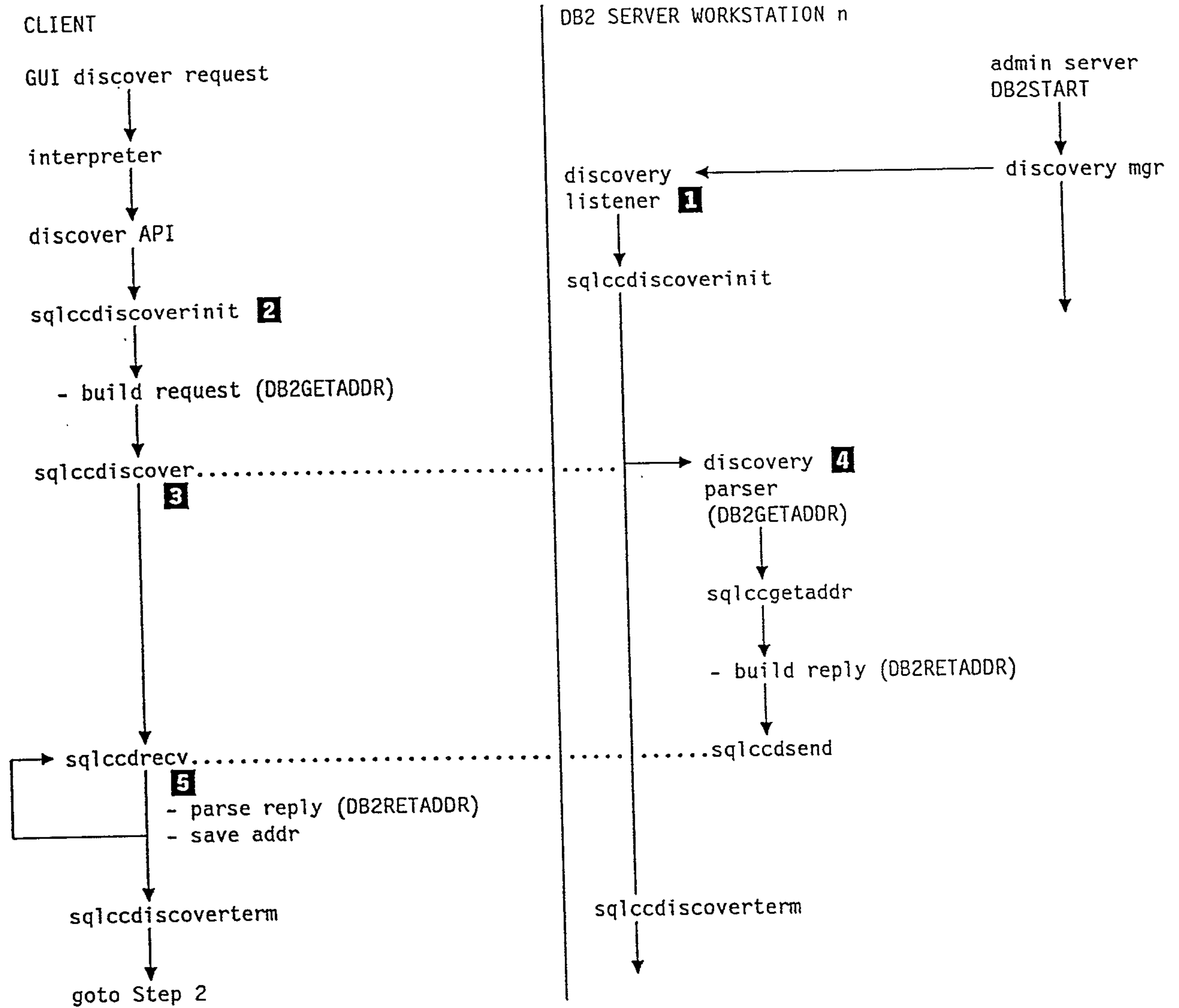


FIG. 6

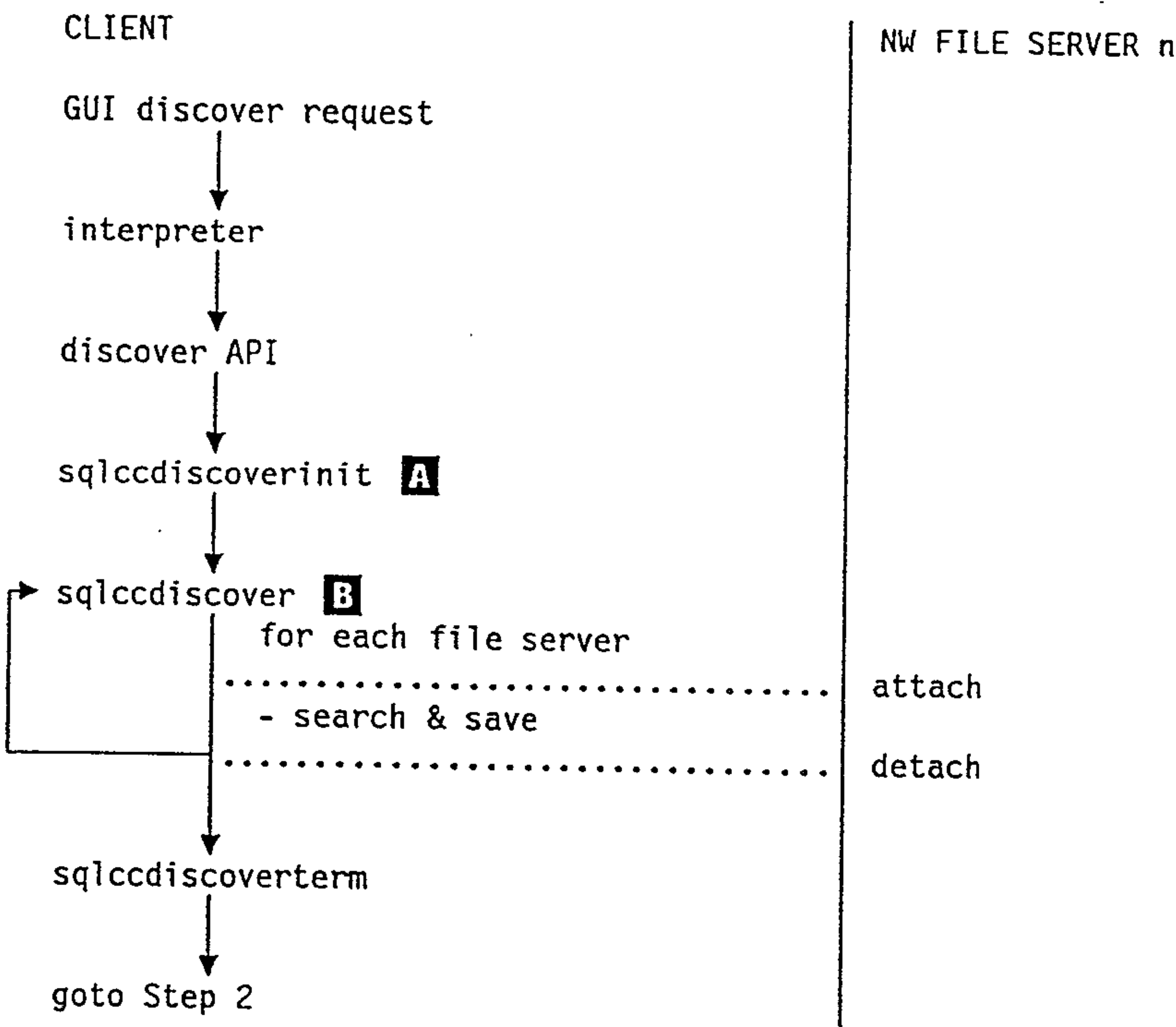
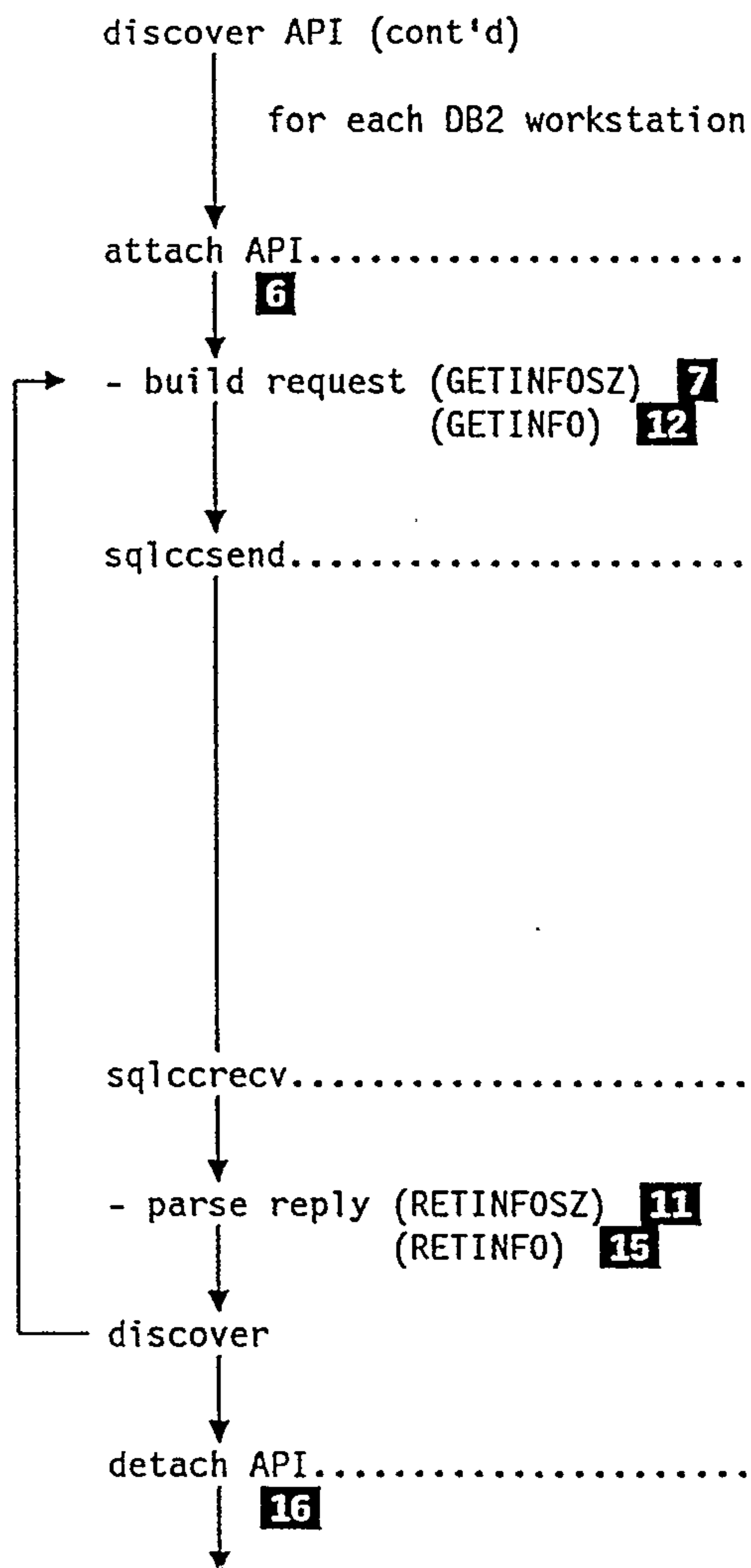


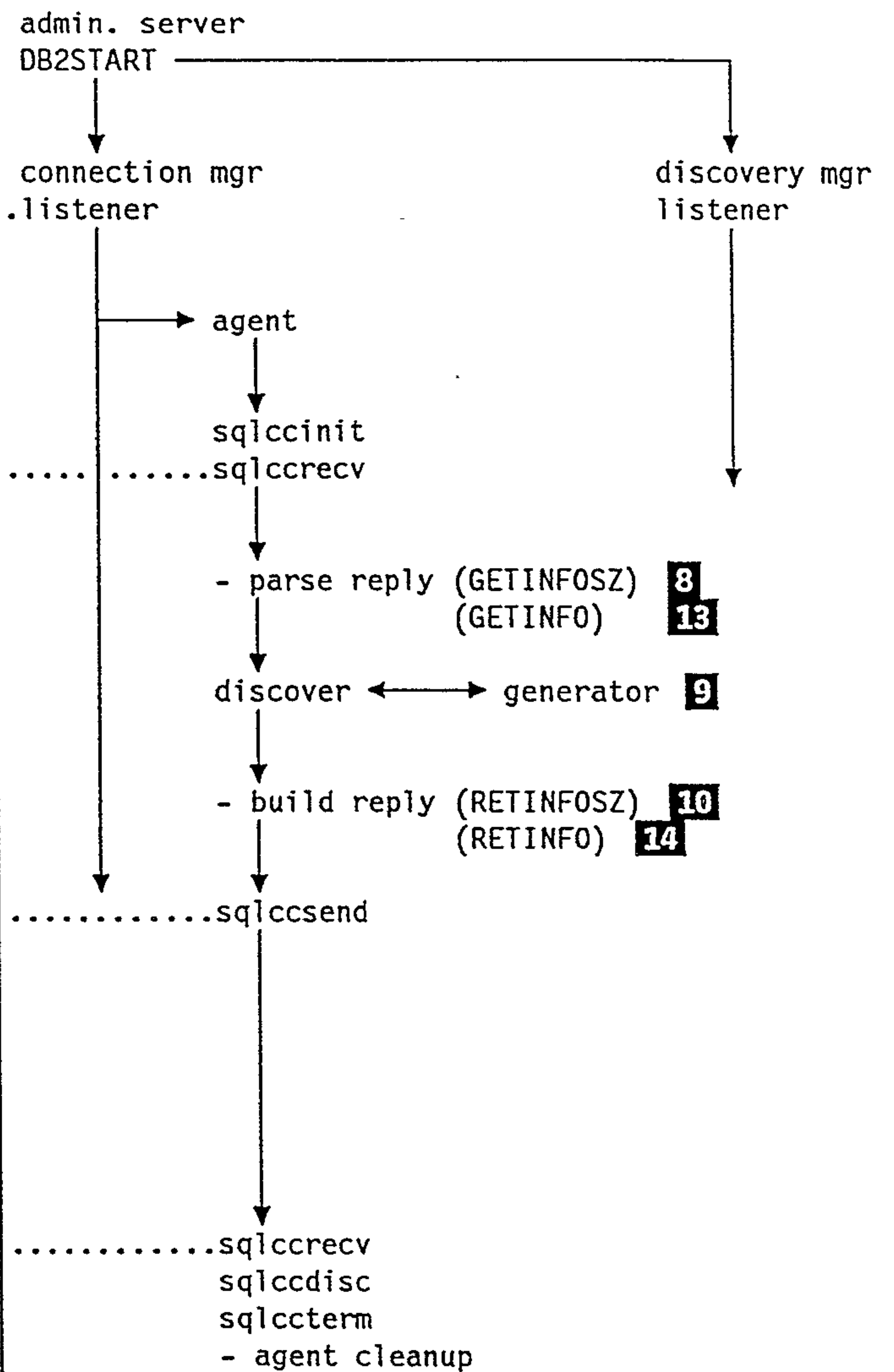
FIG. 7

Step 2 - Directed Flow

CLIENT



DB2 SERVER WORKSTATION n



Client

Server

