Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2616194 C 2015/02/17

(11)(21) 2 616 194

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2006/07/28
(87) Date publication PCT/PCT Publication Date: 2007/02/08
(45) Date de délivrance/Issue Date: 2015/02/17
(85) Entrée phase nationale/National Entry: 2008/01/22
(86) N° demande PCT/PCT Application No.: US 2006/029571
(87) N° publication PCT/PCT Publication No.: 2007/016412
(30) Priorité/Priority: 2005/07/28 (US60/703,687)

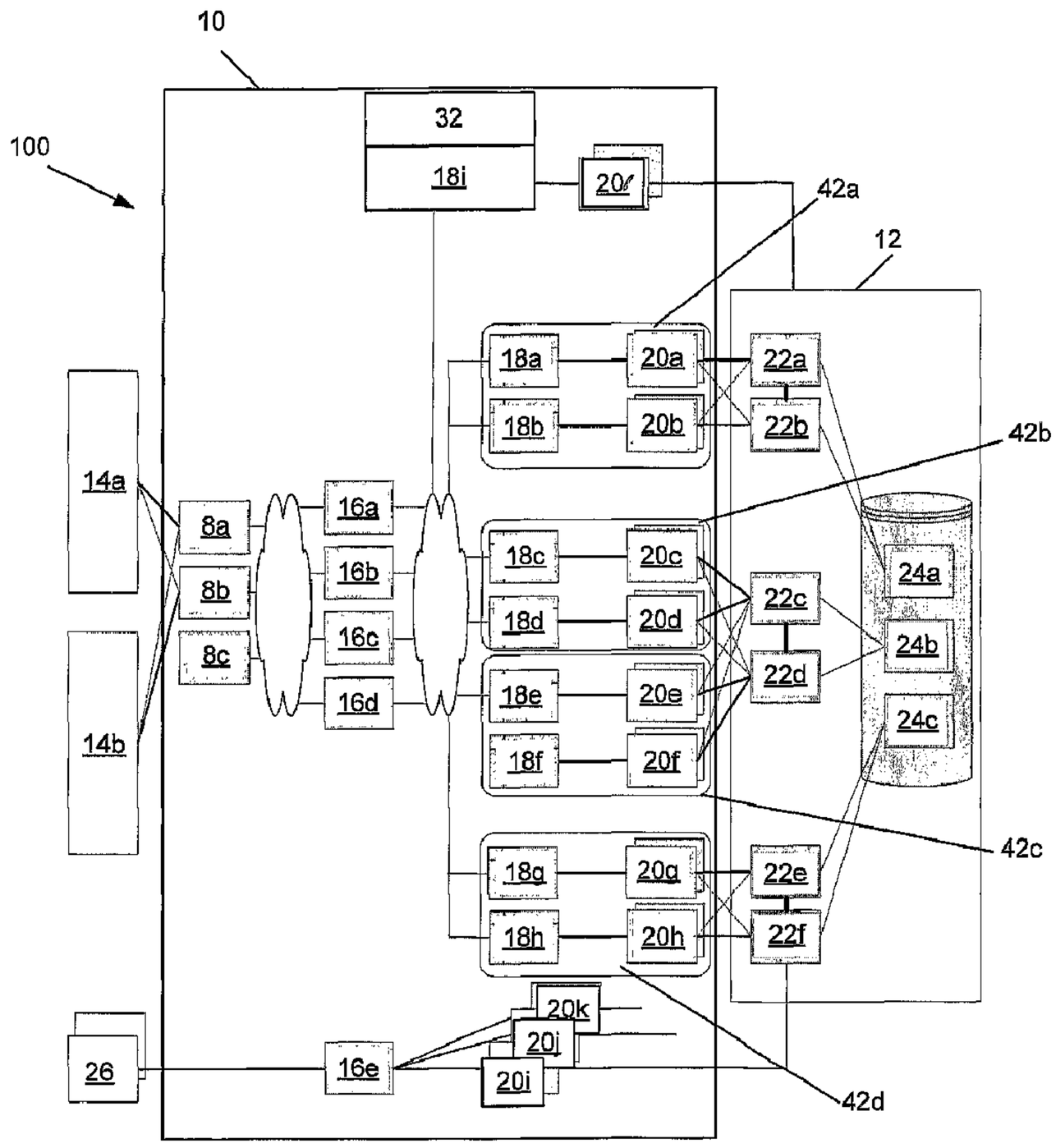
(51) Cl.Int./Int.Cl. *G06F 12/00* (2006.01)

(72) Inventeurs/Inventors:
LABUDA, DAVID SCOTT, US;
KRISHNAMOORTHY, JAYAPRAKASH, US;
HADDOCK, JAMES R., US;
ROCKEL, ALEXANDER, DE;
BREFCZYNSKI, KEITH M., US;
DOUGLAS, GILES, US

(73) Propriétaire/Owner:
ORACLE INTERNATIONAL CORPORATION, US

(74) Agent: RIDOUT & MAYBEE LLP

(54) Titre : SYSTEME ET PROCEDE DE GESTION DES RECETTES
(54) Title: REVENUE MANAGEMENT SYSTEM AND METHOD



(57) Abrégé/Abstract:
A real-time customer relation management system is disclosed. The system can provide increased availability, reduced internal latencies, and reduced data processing and transfer. The system can provide real time processing and batch processing. The

CA 2616194 C 2015/02/17

(11)(21) 2 616 194

(13) C

(57) Abrégé(suite)/Abstract(continued):
system architecture can have an in-memory write-through cache. The cache can store data that would have otherwise been sent to a database. The system can have a backup in-memory write-through cache. The system can use a warm standby, for example, to enhance data backup efficiency.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 February 2007 (08.02.2007)

PCT

(10) International Publication Number
WO 2007/016412 A3

(51) International Patent Classification:
G06F 12/00 (2006.01)

(21) International Application Number:
PCT/US2006/029571

(22) International Filing Date: 28 July 2006 (28.07.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/703,687 28 July 2005 (28.07.2005) US

(71) Applicant *(for all designated States except US)*: **PORTAL SOFTWARE, INC.** [US/US]; 10200 South De Anza Blvd., Cupertino, CA 95014 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **LABUDA, David, Scott** [US/US]; 555 Bryant Street, #604, Palo Alto, CA 94301 (US). **KRISHNAMOORTHY, Jayaprakash** [US/US]; 2252 Lenox Pl., Santa Clara, CA 95054 (US). **HADDOCK, James, R.** [US/US]; 692 Clipper Street, San Francisco, CA 94114 (US). **ROCKEL, Alexander** [DE/DE]; Luttenredder 32, 22457 Hamburg (DE). **BREFCZYNSKI, Keith, M.** [US/US]; 5636 Stevens Creek Blvd., Apt. 381, Cupertino, CA 95014 (US). **DOUGLAS, Giles** [GB/US]; 96 El Dora Dr, Mountain View, CA 94041 (US).

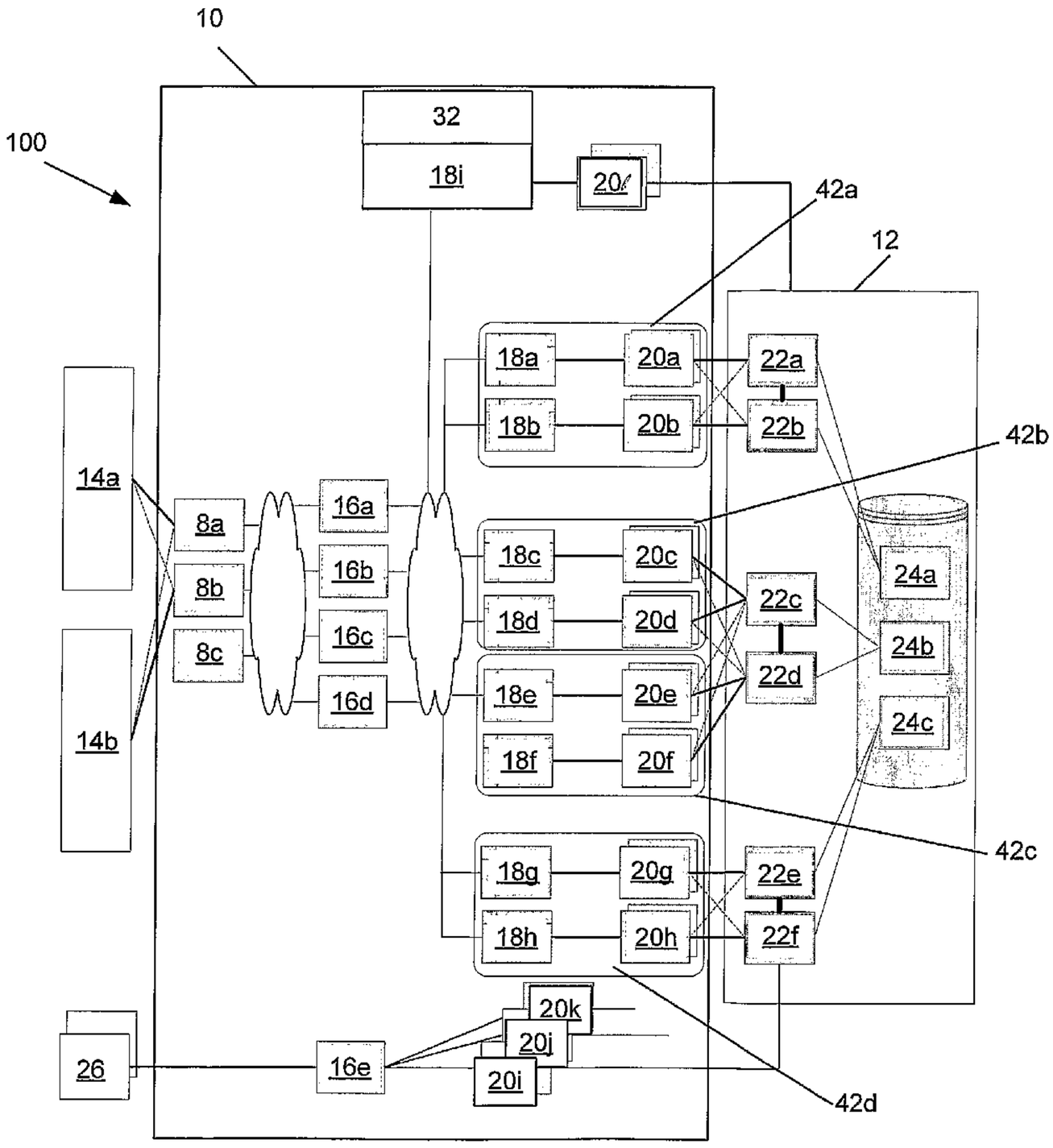
(74) Agent: **LEVINE, David, A.**; Levine Bagade Han LLP, 2483 East Bayshore Road, Suite 100, Palo Alto, CA 94303 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH,

[Continued on next page]

(54) Title: REVENUE MANAGEMENT SYSTEM AND METHOD



(57) Abstract: A real-time customer relation management system is disclosed. The system can provide increased availability, reduced internal latencies, and reduced data processing and transfer. The system can provide real time processing and batch processing. The system architecture can have an in-memory write-through cache. The cache can store data that would have otherwise been sent to a database. The system can have a backup in-memory write-through cache. The system can use a warm standby, for example, to enhance data backup efficiency.

WO 2007/016412 A3

WO 2007/016412 A3

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

Published:

— *with international search report*

(88) Date of publication of the international search report:
12 July 2007

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.

TITLE OF THE INVENTION

REVENUE MANAGEMENT SYSTEM AND METHOD

BACKGROUND OF THE INVENTION

[0002] This invention relates to a revenue management system that has an in-memory write-through cache.

[0003] Prepaid phone accounts are tracked in real-time by billing and time management hardware and software architectures in communication with the phone network switch. The architecture approves the customer's call if there are sufficient funds in the customer's prepaid account. If the customer runs out of time on his prepaid account during a call, the architecture acts to terminate the call.

[0004] These architectures are known as customer relation management (CRM) systems. CRM systems are also used for non-prepaid scenarios, such as for generating bills. CRM systems are also used for other telecommunications, and other network management scenarios.

[0005] Prepaid account CRM systems need to have the ability to track accounts in real-time. Available real-time architectures for managing prepaid customer accounts have some existing limitations.

[0006] First, the available architectures require high performance and data throughput, thereby leading to relatively high hardware requirements. These architectures, along with their storage and maintenance can be expensive and time-consuming.

[0007] Second, requirements for very low system response latencies are difficult to achieve. Transactions in existing architectures involve several round-trips to the disk-based storage subsystem. The data must be processed by a comparatively large software stack to transform from a relational representation into a physical storage format.

[0008] Third, in an available architecture, the data is transferred several times from component to component of the system to retrieve the data, map it from a relational

format to an object format, process it with the desired business logic, and then transfer the response to the client.

[0009] Fourth, currently available architectures can not provide desired levels of data availability to the public phone network (e.g., the switch). The close connection of a prepaid CRM system to the public network increases the data availability requirements. Being part of the public network, some parts of the system need to have carrier-grade availability.

[0010] Also, no single product accomplished both batch processing and real-time processing for telecommunications CRM (e.g., billing) purposes.

BRIEF SUMMARY OF THE INVENTION

[0011] A system and method for managing any numerical account information is disclosed. For example, the system and method can be used for managing revenue for telecommunications system. The system and method can be used to manage account balances, such as user accounts for the telecommunications system. Management of account balances can include altering the balance of the account during use, and/or querying the account (e.g., by the account holder or a customer service representative), and/or querying the account to produce a billing statement or perform other accounting features, and/or querying the account to determine whether to authorize use of the account.

[0012] The system can be used with an account with an existing balance (e.g., prepaid), an account with a maximum use limit (e.g., capped), a current payment account (e.g., now-pay, for example through the use of a credit card), other types of balance management accounts, or combinations thereof.

[0013] The system architecture can be configured to increase performance, and availability and decrease latency. The system and method can manage accounts, for example, for the prepaid wireless markets handling services such as GSM, GPRS and SMS.

[0014] The system can have a rating engine, a billing engine, and a first, high-speed, memory (e.g., transaction in memory object store (TIMOS)). The first memory can be a virtual database cache. The first memory can be a typical on-board RAM storage location.

[0015] The first memory can be a smart cache. The smart cache can treat different object types different ways. For example, the smart cache can treat reference objects, database-only objects, and transient object differently.

[0016] Reference objects can be owned by the database and never updated by the first memory. Reference objects can include dynamic reference objects (e.g., an account balance) that change each call, and static reference objects (e.g., the billing rate for different types of calls) that never or rarely change. Database-only objects can be objects that change one-time or rarely during the call and are not referred to by the connection manager. Transient objects can exist, for example, only in-memory (e.g., in TIMOS). Transient objects can be unwritten to the database. Transient objects can be written to the database, for example, at the end of the call (e.g., credit balance).

[0017] The database can have a data dictionary. The data dictionary can by written by the users. The data dictionary can define an object type and what type of object each other is. Customers can edit the data dictionary if so desired.

[0018] The new revenue management system can have a high availability. The system can have a warm standby operation by referring to any data remaining in TIMOS. During warm standby, in the case of a loss of data, the system can recreate data from the switch and/or TIMOS when the switch sends re-authorization data (e.g., during long calls) or end-of-call data.

[0019] A known failure protection scheme with a high availability (monitor) regularly checking the status of the control manager, TIMOS, data manager, the database manager, and the database is also disclosed.

[0020] A self-container failure protection system is disclosed. Each component of the system can check on the status of its immediately downstream component. If the downstream component has failed, or is passing along a failure message regarding a further downstream component failure, the system can take appropriate action, including alerting a user.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] Figure 1 illustrates a variation of the revenue management system architecture connected to a switch over a network.

[0022] Figure 2 illustrates a variation of the revenue management system architecture connected to a switch over a public network.

[0023] Figures 3 through 5 illustrate variations of the revenue management system architecture.

[0024] Figure 6 illustrates a variation for a method for using the revenue management system.

[0025] Figure 7 illustrates process flows for variations for methods for using the revenue management system.

[0026] Figure 8 illustrates a variation for a method for using the revenue management system.

[0027] Figure 9 illustrates process flows for variations for methods for using the revenue management system.

[0028] Figure 10 illustrates a variation for a method for using the revenue management system.

[0029] Figure 11 illustrates process flows for variations for methods for using the revenue management system.

[0030] Figures 12a through 14 illustrate variations of the revenue management system.

DETAILED DESCRIPTION

[0031] A computer-based system and method for managing any numerical account information is disclosed. For example, the system and method can be used for managing revenue for telecommunications system. The system and method can be used to manage account balances, such as user accounts for the telecommunications system. The management of account balances can include altering the balance of the account during use, and/or querying the account (e.g., by the account holder or a customer service representative), and/or querying the account to produce a billing statement or perform other accounting features, and/or querying the account to determine whether to authorize use of the account.

[0032] Figure 1 illustrates a telecommunication device 2, such as a phone, computer, or fax machine, that can be connected through a public telephone network 4 to a switch 6. The telecommunication device 2 can be communicating with a second telecommunication device through the switch 6. The switch 6 can communicate across a network and through a gateway 8 (e.g., having a protocol translator) to the revenue management system 100. The gateway 8 can communicate directly with a business logic module 10 or business logic application (e.g., Portal Infranet, Portal

Software, Inc. Cupertino, CA). The business logic module 10 can communicate with a database system 12 to determine whether the telecommunication device 2 connected to the switch 6 has permission to connect and/or stay on the line. The database system 12 can have a highly-available Oracle RAC database cluster. The system 100 can utilize Oracle transaction management functionality.

[0033] Figure 2 illustrates that the gateway 8 can be a part of the revenue management system 100. The gateway 8 can interface between the business logic module 10 and an intelligent network (IN) service control point system (SCP) 14. The service control point system 14 can facilitate communication between the switch 6 and the gateway 8. The SCP system 14 can be software or a remote computer database within the network that receives queries, for example from service switching points (SSP), in order to process applications, such as 800 and LNP number lookups and calling card verification. The SCP system 14 can process the applications utilizing the customer management system 100. The gateway 8 can be a high-speed protocol translator from the IN SCP to the remainder of the revenue management system 100.

[0034] Figure 3 illustrates that the business logic module 10 can have one or more rating connection managers (CM) 16a and 16b, a first memory data manager 18 (e.g., TIMOS Data Manager (DMT) from Portal Software, Inc.), and one or more second memory data managers 20a and 20b (e.g., Oracle Data Manager (DM Oracle). The second memory data managers 20a and 20b can communicate with the database system 12 or other second memory system. The database system 12 can have one or more database clusters 22a and 22b (e.g., Oracle Real Application Clusters), for example, providing high availability and scalability for databases running on the cluster. The database clusters 22a and 22b can support one or more databases 24.

[0035] The business logic module can be accessed via the gateway 8 and/or via a manual access application 26. The manual access application 26 can be operated manually or automatically. The manual access application 26 can be configured, for example, to be used by billing software to generate invoices, and/or by a customer service representative to check on account status, and/or by the account-holder to check account status.

[0036] The revenue management system 100 can have a first memory (e.g., TIMOS) and a second memory (e.g., database). The first memory can be, for example, in and/or in communication with the first memory data manager 18. The first memory

can be configured to have faster, slower, and/or the same read, and/or write, and/or re-write speeds (e.g., access speeds) as the second memory. The first memory can be an in-memory data store and database cache dedicated to high-speed rating and authorization requirements.

[0037] The first memory can be solid state memory, such as system memory (e.g., RAM) or one or more hard drives, for example with fast access speeds. Requests for data in the first memory can be processed faster than requests for data in a second memory.

[0038] The first data in a first data object can be stored in the first memory in the format used by the business logic module 10 (e.g., Portal Infranet, Portal Software, Inc.). The first data can be left untranslated before storage in the first memory. The internal search and storage algorithms can be optimized for in-first-memory data. Storing the first data in the first memory can, for example, eliminate the round trip to the second memory (e.g., one or more databases, such as on database servers), and can speed the process of storing, editing and/or querying the first data. Object creation or updates for the first data objects can require no access of the second memory. Updates for the first data objects can be performed in the first memory. The system can have, for example, a reduced throughput and/or latency.

[0039] The first memory data objects (e.g., transient objects) can be stored in the first memory and/or the second memory. For example, the first memory data objects can be stored not in the database and not be persisted in the first memory. The first memory objects can, for example, exist only in the process heap memory of the first memory. The first memory objects can be, for example, managed in a transactional manner (e.g., like the other memory objects).

[0040] First memory data objects can be removed from the first memory by shutdown of the first memory or the business logic executing a delete operation on the first memory data object. The store for first memory data objects can be a fixed size, for example, determined during startup of the first memory process.

[0041] The first memory data manager 18 can be configured to improve access times and latency on moving and/or writing and/or editing and/or deleting and/or querying objects.

[0042] The second memory can be in and/or in communication with the second memory data manager 20. Requests for second memory data objects can be sent to the second memory data manager 20. The second memory, for example, can be a

disk-based (e.g., on one or more hard drives) database. The database can be a relational database (RDBMS).

[0043] The system can have low access second memory data objects (e.g., database-only objects). The low access second memory data objects can be stored primarily and/or exclusively in the second memory (e.g., one or more databases). The low access second memory data objects can be stored in the first memory none of the time, or some of the time.

[0044] The first memory data manager can access the low access second memory data object type via a pass-through mode. For example, requests can be forwarded to the second memory data manager (e.g., DM_Oracle), and responses can be forwarded back to the first memory data manager.

[0045] The high access second memory data objects (e.g., reference objects) can be updated seldom and not during high-speed session processing. The high access second memory data objects can be stored (cached) in a first memory reference object cache (ROC). The high access second memory data objects can grow in number in relation to growth in the subscriber base.

[0046] The high access second memory data objects can exist in the first memory an equal amount of time as length of the first memory process. A newly started first memory instance can contain no high access second memory data objects.

[0047] Updating and creating high access second memory data objects can be performed in the second memory and in the first memory. in the high access second memory data objects can be updated or created asynchronously or synchronously in the second memory and the first memory.

[0048] The high access second memory data objects can be static or dynamic. The static high access second memory data objects can be queried, updated, created, or deleted at irregular intervals. The static high access second memory data objects can be, for example, subscriber information such as the list of subscribed services and the chosen tariff plans.

[0049] The dynamic high access second memory data objects can be touched (e.g., queried, updated, created, deleted) after the completion of each session. The dynamic high access second memory data objects can be, for example, the monetary and non-monetary balances belonging to a subscriber account.

[0050] A standby-first memory (e.g., for a high availability variation that can have an active first memory and a backup, standby first memory) can preload the static high

access second memory data objects. Changes of static reference objects can be propagated from the active first memory to the standby first memory.

[0051] The gateway 8 can directly communicate with the business logic module 10. For example, during a customer's use of the telecommunication network, the gateway 8 can communicate with a first connection manager (CM) 16a. The gateway 8 can pass requests to the CM 16a, for example, calling the appropriate business logic routines depending on the type of request that is indicated from the IN SCP 14. The gateway 8 can be nearly stateless. The gateway 8 can provide fast failover capabilities, for example, accompanied by a degraded mode of operation that is used when the lower architecture layers become unavailable. The gateway 8 can perform authentication, authorization and accounting procedures.

[0052] Events received by the CM can be rated via an embedded rating engine using the data provided from the first memory data manager 18 (e.g., DM TIMOS cache) and the database system 12. The rating engine can produce rates for customer use of the telecommunications network under the specific conditions that apply (e.g., time of day, day of week, network used). The rating engine can cache pricing objects itself, for example, in order to reduce the number of network roundtrips necessary to complete the rating phase. The rating engine can perform zoning and discounting rating.

[0053] Based on the object type, the first memory data manager 18 can pass the request to the database system 12, query the first memory data manager 18 reference object cache or accesses the first memory (e.g., in-memory store) for transient objects. The object types and their locations can be defined in a business logic database (e.g., Infranet Data Dictionary by Portal Software, Inc.), which can be in the database system 12. Traffic for objects not in the first memory can be allowed to bypass the by accessing the database manager 20, for example, in the same way a commonly used system without the first data manager would be configured. The data integrity of the first memory can be ensured by a platform-managed synchronization mechanism that can propagate the necessary updates to the first memory. The first memory can have one or more caches.

[0054] Figures 4a and 4b illustrate variations of the revenue management system 100. The gateway 8 can act as a high-speed protocol translator as well as an SLA monitor with fallback capabilities. The CM 16 can receive requests from the gateway 8. The CM 16 can have the authorization, authentication and accounting business logic (e.g.,

for delivery to the gateway 8). The CM 16 can call operational codes on the first data manager 18. The CM 16 can be replaced with another client, such as a migration tool. The CM 16 can have a realtime pipeline (RTP) 28. The RTP 28 can be configured to adjust the rating, for example by discounting and zoning the rate. The RTP 28 can be optionally used by the CM 16 while rating.

[0055] The first data manager 18 can have a data migratory subsystem 30. The data manager subsystem 30 can be used to fill the high access second memory object cache after start or fail over.

[0056] The first data manager 18 can have a directory server 32. The directory server 32 can be configured to identify the correct first memory/second memory combinations in scaled scenarios with more second memory instances than first memory instances or more first memory instances than second memory instances. The directory server 32 can enable the gateway instances and CM instances to be independent of the number of first memory instances. The number and location of gateway and CM processes can be flexibility and scalability with respect to the number and location of first memory instances.

[0057] Figure 4a shows that the elements of the architecture of the business logic module can all be standalone. Figure 4b illustrates that the numerous elements of the architecture can be integrated.

[0058] Figure 5 illustrates that the first memory 102 can have a reference object cache (ROC) 34 and a transient object store (TOC) 36. The ROC 34 can be managed by a separate set of rules than the TOC 36. The ROC 34 and the TOC 36 can be in the same or different parts of the first memory 102. The first memory 102 can be part of, or separate but in communication with, the first memory data manager 18. The ROC 34 can be configured to cache high access second memory data objects (e.g., reference objects). The TOC 36 can be configured to store first memory data objects (e.g., transient objects).

[0059] Figure 6 illustrates a method for accessing a first memory data in the TOC 36. The CM 16 can send, shown by arrow 38, a request to the first data manager 18. The request can apply to the first memory data. The first data manager 18 can analyze the request 38. The first data manager 18 can conclude that the request applies to the first memory. The first data manager 18 can apply or execute the request on the TOC 36. The first data manager 18 can generate a reply and send, shown by arrow 40, the reply to the CM 16.

[0060] Figure 7 illustrates flows of various requests from the CM 16 and the replies to the requests. The instructions are shown as create, update, delete and search/read (i.e., query). The request from the CM 16 can be, respectively, create the first data object, update the first data object, delete the first data object, and search/read the first data object. (The numbers of the requests and replies illustrate an exemplary chronological order.) The first data manager 18 can convert or otherwise translate the request from the CM 16 to a first data manager instruction, such as add the first data object, change the first data object, remove the first data object, and find the first data object, respectively. The first data manager 18 can apply or execute the first data manager instruction on the TOC 36. The first data manager 18 can then return a reply. The replies can include the data searched, and/or confirmation that the task was completed successfully, and/or an error code and or error explanation.

[0061] Figure 8 illustrates a method for accessing high access memory data in the ROC 34 and in the database system 12. The CM 16 can send, shown by arrow 38, a request to the first data manager 18. The request can apply to the high access second memory data. The first data manager 18 can analyze the request 38. The first data manager 18 can conclude that the request applies to the high access second memory data. The first data manager 18 can determine whether the high access second memory data is in the ROC 34. If the first data manager 34 determines that the high access second memory data is in the ROC 34, the first data manager 18 can apply or execute the request on the high access second data in the ROC 34. The first data manager 18 can send the request to the second data manager 20. The second data manager 20 can apply or execute the request on the high access second data in the database system 12. The database system 12 and/or the second data manager 20 and/or the first data manager 18 can generate one or more replies. The replies can be sent, shown by arrow 40, directly or via the first data manager 18 to the CM 16.

[0062] Figure 9 illustrates flows of various requests from the CM and the replies to the requests. The exemplary instructions are shown as create, update, delete, simple and complex searches/reads (i.e., query). The request from the CM 16 can be, respectively, create the first data object, update the first data object, delete the first data object, and search/read the first data object. (The numbers of the requests and replies illustrate an exemplary chronological order.) The first data manager 20 can convert or otherwise translate the request from the CM 16 to a first data manager instruction, such as add the first data object, change the first data object, remove the

first data object, and find the first data object, respectively. The first data manager can then apply or execute the translated request on the high access second data in the ROC 34.

[0063] The first data manager 18 can send the request to the second data manager 20 and/or the CM 16 can send the request directly to the second data manager 20. The second data manager 20 can convert or otherwise translate the request to a second data manager instruction, such as insert the row of data, update the row of data, delete the row of data, and select the row or rows of data, respectively (with no response shown for a simple search/read, although the second data manager can perform simple searching). The second data manager 20 can apply or execute the request on the high access second data in the database system 12. The second data manager 20, and/or the database system 12 and/or the first data manager 20 can then return a reply. The replies can include the data searched, and/or confirmation that the task was completed successfully, and/or an error code and or error explanation.

[0064] Figure 10 illustrates a method for accessing low access memory data in the database system 12. The CM 16 can send, shown by arrow 38, a request to the first data manager 18, and/or directly to the second data manager 18. The request can apply to the high access second memory data. The first data manager 18 can analyze the request 38. The first data manager 18 can conclude that the request applies to the low access second memory data. The first data manager 18 can send the request to the second data manager 20. The second data manager 20 can apply or execute the request on the high access second data in the database system 12. The database system 12 and/or the second data manager 20 can generate one or more replies. The replies can be sent, shown by arrow 40, directly or via the first data manager 18 to the CM 16.

[0065] Figure 11 illustrates flows of various requests from the CM and the replies to the requests. The exemplary instructions are shown as create, update, delete, and searches/read (i.e., query). The request from the CM 16 can be, respectively, create the first data object, update the first data object, delete the first data object, and search/read the first data object. (The numbers of the requests and replies illustrate an exemplary chronological order.) The first data manager 18 can send the request to the second data manager 20. The second data manager 20 can convert or otherwise translate the request to a second data manager instruction, such as insert the row of data, update the row of data, delete the row of data, and select the row or rows of data,

respectively. The second data manager 20 can apply or execute the request on the high access second data in the database system 12. The second data manager 20, and/or the database system 12 and/or the first data manager 20 can then return a reply. The replies can include the data searched, and/or confirmation that the task was completed successfully, and/or an error code and or error explanation.

[0066] The CM 16 can send requests directly to the desired data manager 18 or 20 and/or the CM 16 can tag the request and the first data manager 18 can analyze the tag to determine whether to apply and/or execute the request and/or whether to send the request to the second data manager. The tag can be the substance of the request (i.e., the requested action) and/or additional data solely to communicate the desired final location of the request.

[0067] The first memory data objects of this category can be created, updated or deleted in the high-speed access path of the revenue management system 100. Examples of the first memory data objects include active session objects and resource reservation objects.

[0068] The first memory data objects can be analyzed using, for example, logical predicates (e.g., equals, not equals). Queries executed on first memory data can specify an index to use to satisfy the query. The index can be a hash to enable fast value lookup. The index can be a single column index. Predicates on other columns can be supported by filtering the result set to find matches.

[0069] Requests for the first memory data objects can be passed to a standard heap memory area. The requests can be created, changed and deleted within transactions.

[0070] The first memory data objects can be limited to particular object, such as business object types.

[0071] The high access second memory data objects can be accessed only in a read-only mode in the high-speed access path. An example of the high access second memory data objects is customer account information.

[0072] The ROC 34 can be filled on demand. This means that requests can be redirected to the database system 12 if the high access second memory data object is not found in the ROC 34. If the request is a read of an entire object, the ROC 34 can be filled or cached by the reply (e.g., as the reply passes through the first data manger 18 on the reply's route back to the CM 16 from the second data manager 20). Partial object requests ('read_fields') of the high access second memory data objects can be cached in a similar manner to that performed for the entire object.

[0073] The high access second memory data objects can be fully queried. Simple queries involving basic logical operators (e.g., equals, not equals) can be performed by the first data manager 18 on the high access second memory data objects in the ROC 34. Complex queries (e.g., involving joins to other objects, or operators such as 'like' or 'in') can be performed by the second data manager 20 on the high access second memory data objects in the database system 12.

[0074] The dynamic high access second memory data objects can be loaded by the data migrator 30 after a failover.

[0075] The static high access second memory data objects can be loaded by the data migrator 30 immediately after the backup first memory system has been started. The static high access second memory data objects can be synchronized with the database via the first memory synchronization system.

[0076] The low access second memory data objects can be absent from the first memory. Requests for the low access second memory data objects can be routed directly from first data manager 18 to the second data manager 20. The low access second memory data objects can be fully queried.

[0077] The first data manager can allow reading of the first data values during a write operation. The first data manager can have the write operation take place on a scratchpad of data that is only visible to the writing transaction. The first data manager can serialize the first data while the update is moved to main memory at the commit time.

[0078] The first data manager can have a read committed isolation. The read committed isolation makes all committed updates available to transactions even if the commit takes place after the transaction is started. Read committed isolation can prevent "dirty" reads (i.e., the first data manager preserves the earlier first data value for reading during pending changes to the first data value).

[0079] The first data manager can support or not support statement or transaction level consistent reads.

[0080] The revenue management system 100 can be configured to route any traffic not related to session handling can be routed to and/or away from the first memory data manager 18. A synchronization system can be used to send updates to the first memory data manager 18. The synchronization system can automatically propagate changes affecting objects stored in the first memory to all the first memory instances caching the particular object or object type.

[0081] The revenue management system 100 can have a convergence system. The convergence system can load batch data via the first data manager 18 into the revenue management system 100, for example, to share any data of batch origin, such as balances between prepaid and postpaid accounts.

[0082] The data capacity of a first memory instance can be lower than data the capacity of a second memory (e.g., database) instance. One second memory instance can support several shared-nothing instances of the first memory. (The commonly used term is m:n (m – first memory instances / n – second memory instances)).

[0083] The first memory data manager 18 can reduce the latency for objects first memory data manager 18 handles, and at the same time enabling increased throughput of the system 100.

[0084] For installation of first memory data manager 18, the first memory data manager 18 can be configured to be inserted between the CM 16 component and second memory data manager 20 component. The introduction of the first memory data manager 18 can change the access characteristics of some object types for a pre-existing revenue management system that did not have the first memory data manager 18. Installation of the first memory data manager 18 can be configured to be transparent (e.g., not change object types). The system 100 can be configured so that the higher-level business logic architecture layers cannot tell first memory data manager 18 is present. However, the business logic can be changed to utilize the first memory data manager 18. These changes can be ignored by the system 100 if the first memory data manager 18 is not present.

[0085] The first memory data manager 18 can be installed in an existing revenue management system. For example, the first memory data manager 18 can be physically installed (e.g., mounting hardware and/or loading software onto the appropriate computer-readable medium) and the base software can be configured.

[0086] After the installation of the base software, the first data can then be migrated into the first memory data manager 18. The first memory data object residencies stored in the data dictionary can take effect, loading the first data onto the first data manager 18 during use. The residencies can be part of the default business logic module 10 installation (having no effect when the first memory data manager 18 is not present) or can be loaded onto the business logic module 10 during the installation of the first memory data manager 18.

[0087] Reference objects can be migrated by loading into the first data manager 18 when accessed for the first time and/or pushed into the first memory data manager 18 by the data migratory 30.

[0088] Data objects can be redefined as first memory data objects (or low access or high access second memory data objects) by deploying the data object via the normal mechanism, and then updating the residency type in the data dictionary.

[0089] After a process startup, the first memory data manager 18 can have an empty ROC 34. A separate data migration thread can push all high access second memory data in the ROC 34. The static high access second memory data objects can be loaded into the ROC 34. For example, a first memory data manager 18 in backup mode can load only the static high access second memory data objects into the ROC 34.

[0090] The data migrator 30 can provide a notification hook to signal the end of the migration and/or startup phase to other processes. The first memory data manager 18 can be operational immediately after start (e.g., before the migratory tool sends the notification hook), for example, with an empty cache. The first request after startup for a specific first data object can trigger that first data object to be loaded into the cache (e.g., if the data migrator 30 has not yet loaded the desired first data object already).

[0091] Upon a system shutdown, the first data manager 18 can close the process log file, and release used memory. The high access second memory data objects can be unaffected by shutdown (e.g., remaining stored on the second memory).

[0092] The revenue management system 100 can provide hooks to verify and monitor performance. The revenue management system 100 can log performance data on a regular basis and/or make performance data available via an embedded web server. A signal can be sent to the second memory data manager to collect desired data (e.g., for some parts of the system).

[0093] The revenue management system 100 can create system logs that can monitor operation of the revenue management system 100. A log monitoring GUI (e.g., Pipeline log viewer) can be used. Business logic style pin-logging can, for example, aid debugging and diagnosis.

[0094] The first memory data manager 18 can have a pipeline framework tracing model. Additional trace information can be collected from subsystems of the revenue management system 100 on a case by case basis.

[0095] Figures 12a, 12b and 12c illustrate that the revenue management system 100 can be scalable to large scale expansion.

[0096] The revenue management system 100 can have multiple second memory locations (e.g., databases 24a, 24b and 24c). The revenue management system 100 can have separate instances of the first memory data manager 18a-18i, and the second memory data manager 20a-20l. Pairs of sets of first memory data managers and second memory data manager, for example, 18a, 18b, 20a, and 20b can be formed into high availability (HA) pairs 42. The HA pairs 42 can have active and backup first data managers 18a and 18b, respectively, for example, and active and backup second data managers 20a and 20b, respectively, for example.

[0097] The revenue management system 100 can have a capacity partitioning scheme.

[0098] Each second memory (e.g., database 24) instance can be associated (i.e., in communication) with one or more first memory data manager 18 instances. The revenue management system can be configured so no data is stored in overlapping second memory instances (*n* Timos instances : *1* database). The business logic module can have several, independent databases (*m*). The combination of TIMOS/databases can be referred to as *m:n* configuration.

[0099] The CM 16a-16e can lookup in the directory server 32 to identify the first memory data manager 18 and second memory data manager 20 (or database 24) combination applicable for a certain object.

[0100] The revenue management system 100 can have account migration tools. The account migration tools can move subscriber data from one first and/or second memory location (e.g., database 24 and/or first memory data manager 18) to another first and/or second memory location.

[0101] Multiple second memory data managers 20 can communicate with the same database clusters 22. All the databases can be managed with one database cluster 22 (e.g., one RAC cluster) (not shown).

[0102] The SCPs 14, gateways 8 and CMs 16 can each be associated to multiple first memory data managers 18. The gateway 8 can support load balancing over several CMs 16. The CMs 16 can use the directory server 32 to route the requests to the correct first memory data managers 18.

[0103] The database system 12 can run multiple database schemes in one RAC cluster 22. The revenue management system 100 can be configured to associate a dedicated set of resources to just one SCP 14 or group of SCPs 14.

[0104] Figure 12b illustrates that the first memory data managers 18c and 18d can communicate directly with the database system 12, for example with the database clusters 22a and 22b.

[0105] Figure 12c illustrates that the revenue management system can have two or more database systems 12a and 12b.

[0106] Figure 13 illustrates a failure protection scheme with a high availability monitor 44 regularly checking the status of the control manager, the first memory in the first memory data manager 18, the second memory data manager 20, the database cluster , and the database 24.

[0107] Figure 14 illustrates a self-contained failure protection system. Each component of the revenue management system 100 can check on the status of its immediately downstream component (e.g., the second memory data manager 20 is immediately downstream of the first memory data manager 18). If the immediately downstream component has failed, or is sending a failure message regarding a further downstream component failure, the revenue management system can take appropriate action, including alerting a user that a failure has occurred. The revenue management system 100 can be absent of a separate monitor component checking for system failures.

[0108] The revenue management system 100 can have a high availability. The revenue management system 100 can have a warm standby operation by referring to any data remaining in the first memory (e.g., TIMOS). During warm standby, in the case of a loss of data (e.g., during a system failure), the revenue management system 100 can recreate data from the switch 6 and/or the first memory when the switch 6 sends re-authorization data (e.g., during long calls) or end-of-call data.

[0109] The database system 24 can store the latest static high access second memory data before a loss of data. The static high access second memory data can be recovered to the first memory from the database system 24 after a loss of data in the first memory.

[0110] In the revenue management system 100, higher layers (i.e., more stable during a system failure, such as the database) of architecture with very high availabilities can partially or completely backup lower layers (i.e., less stable during a system failure, such as a solid state RAM variation of the first memory) with lesser availabilities in case of failures.

[0111] The revenue management system 100 can have spare, unused hardware and software such as backup data managers in the high availability pairs 42, as shown in Figures 12a through 12c. The high availability pair 42 can have active and a backup first data managers 18 and active and backup second data managers 20. The backup data managers can copy from the respective active data managers, for example during a period of no other activity with the active data manager and/or from a sketchpad, and/or the last available data from the active data manager. In case of failure of an element, the backup or other inactive elements will be able to restore data and/or take over the additional load.

[0113] Accessing can include querying, updating, creating, deleting and combinations thereof. Querying, updating, creating, and deleting for any data can be interchanged with each other as disclosed.

[0114] It is apparent to one skilled in the art that various changes and modifications can be made to this disclosure, and equivalents employed, without departing from the scope of the invention. System and architecture are used as interchangeable terms, both referring to one or more hardware and software components in communication. All elements shown herein can be software and/or hardware components. Elements shown with any embodiment are exemplary for the specific embodiment and can be used on other embodiments within this disclosure.

CLAIMS:

1. A computer-based telecommunications network account management system comprising:

a first memory having a first memory access speed and storing one or more transient objects related to a communication on the telecommunications network, wherein the first memory manages the transient objects per communication and wherein the transient objects are not persisted;

a second memory having a second memory access speed and storing one or more reference objects related to the communication, wherein the second memory persists the reference objects; and

a first memory manager configured to route a first request for a first object to the first memory, and configured to route a second request for a second object not to the first memory,

wherein the first memory has a faster access speed than the second memory, wherein routing the first request for the first object to the first memory and routing the second request for the second object to the second memory are performed based on an object type for each of the first object and the second object and wherein the object types for the first object and the second object are defined in a user defined data dictionary stored in the second memory and wherein the data dictionary stores a user defined object type for the first object and a user defined object type for the second object.

2. The system of claim 1, further comprising a second memory manager configured to route the second object to the second memory.

3. The system of claim 1, wherein the second memory comprises a hard drive.

4. The system of claim 1, wherein the first memory comprises solid state memory.

5. The system of claim 1, wherein the first memory comprises Random Access Memory (RAM).

6. The system of claim 1, wherein the second memory comprises a relational database.

7. The system of claim 1, further comprising a database cluster.

8. The system of claim 1, further comprising a gateway configured to interface with a telecommunications network.

9. The system of claim 1, wherein the first request comprises a query.

10. The system of claim 9, wherein the second request comprises a query.

11. The system of claim 1, wherein the first request comprises a request to write.

12. The system of claim 11, wherein the second request comprises a request to write.

13. The system of claim 1, wherein the object type for the first object comprises a transient object.

14. The system of claim 13, wherein the object type for the second object comprises a reference object.

15. A method for managing a telecommunications network account, the method comprising:

storing in a first memory having a first memory access speed one or more transient objects related to a communication on the telecommunications network, wherein the first memory manages the transient objects per communication and wherein the transient objects are not persisted;

storing in a second memory having a second memory access speed one or more reference objects related to the communication, wherein the second memory persists the reference objects;

routing by a first memory manager a first request for a first object to the first memory; and

routing by the first memory manager a second request for a second object not to the first memory,

wherein the first memory has a faster access speed than the second memory, wherein routing the first request for the first object to the first memory and routing the second request for the second object to the second memory are performed based on an object type for each of the first object and the second object and wherein the object types for the first object and the second object are defined in a user defined data dictionary stored in the second memory and wherein the data dictionary stores a user defined object type for the first object and a user defined object type for the second object.

16. The method of claim 15, further comprising routing by a second memory manager the second object to the second memory.

17. The method of claim 15, wherein the second memory comprises a hard drive.

18. The method of claim 15, wherein the first memory comprises solid state memory.

19. The method of claim 15, wherein the first memory comprises Random Access Memory (RAM).

20. The method of claim 15, wherein the second memory comprises a relational database.

21. The method of claim 15, wherein the first request comprises a query.

22. The method of claim 21, wherein the second request comprises a query.

23. The method of claim 15, wherein the first request comprises a request to write.

24. The method of claim 23, wherein the second request comprises a request to write.

25. The method of claim 15, wherein the object type for the first object comprises a transient object.

26. The method of claim 25, wherein the object type for the second object comprises a reference object.

27. A computer-readable storage medium having stored thereon a sequence of instruction which, when executed by a processor, cause the processor to manage a telecommunications network account by:

storing in a first memory having a first memory access speed one or more transient objects related to a communication on the telecommunications network, wherein the first memory manages the transient objects per communication and wherein the transient objects are not persisted;

storing in a second memory having a second memory access speed one or more reference objects related to the communication, wherein the second memory persists the reference objects;

routing by a first memory manager a first request for a first object to the first memory; and

routing by the first memory manager a second request for a second object not to the first memory,

wherein the first memory has a faster access speed than the second memory, wherein routing the first request for the first object to the first memory and routing the second request for the second object to the second memory are performed based on an object type for each of the first object and the second object and wherein the object types for the first object and the second object are defined in a user defined data dictionary stored in the second memory and wherein the data dictionary stores a user defined object type for the first object and a user defined object type for the second object.

28. The computer-readable storage medium of claim 27, further comprising routing by a second memory manager the second object to the second memory.

29. The computer-readable storage medium of claim 27, wherein the second memory comprises a hard drive.

30. The computer-readable storage medium of claim 27, wherein the first memory comprises solid state memory.

31. The computer-readable storage medium of claim 27, wherein the first memory comprises Random Access Memory (RAM).

32. The computer-readable storage medium of claim 27, wherein the second memory comprises a relational database.

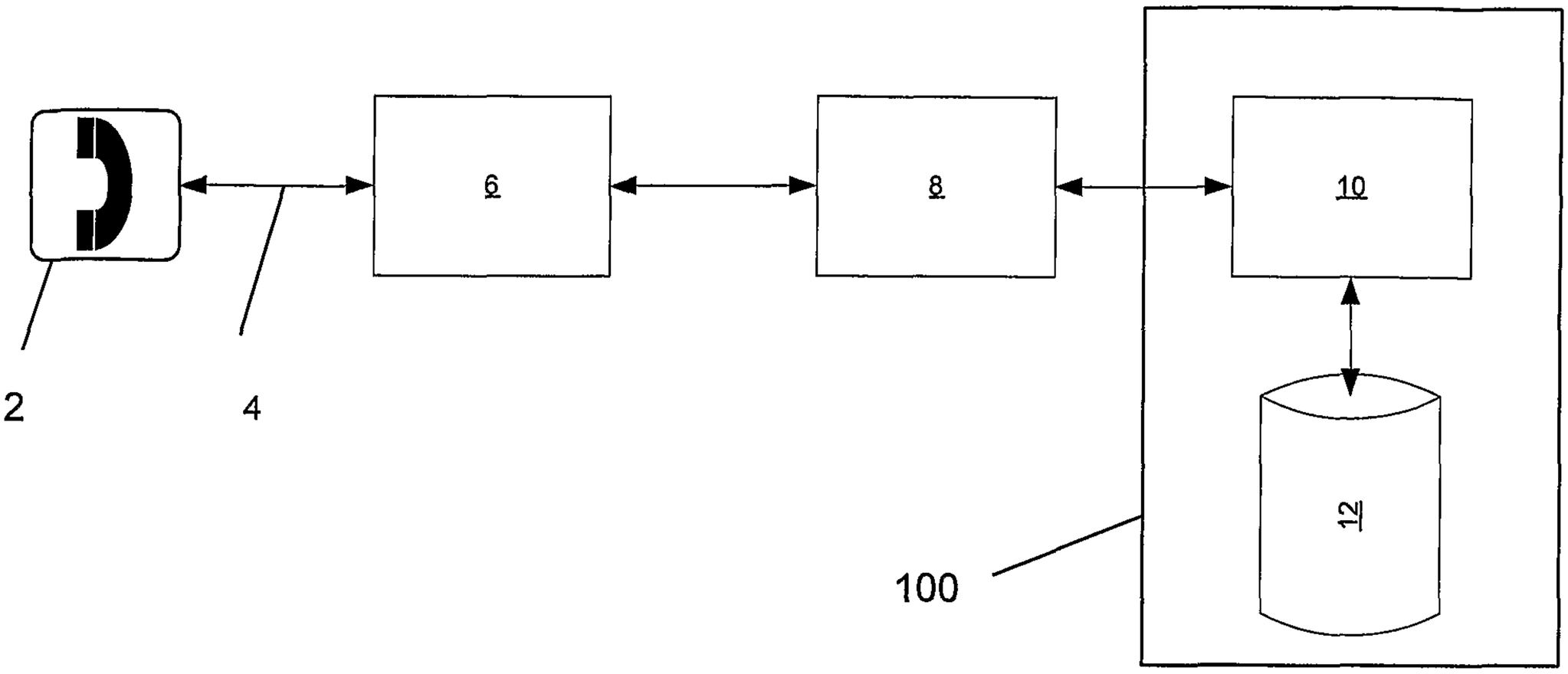


Figure 1

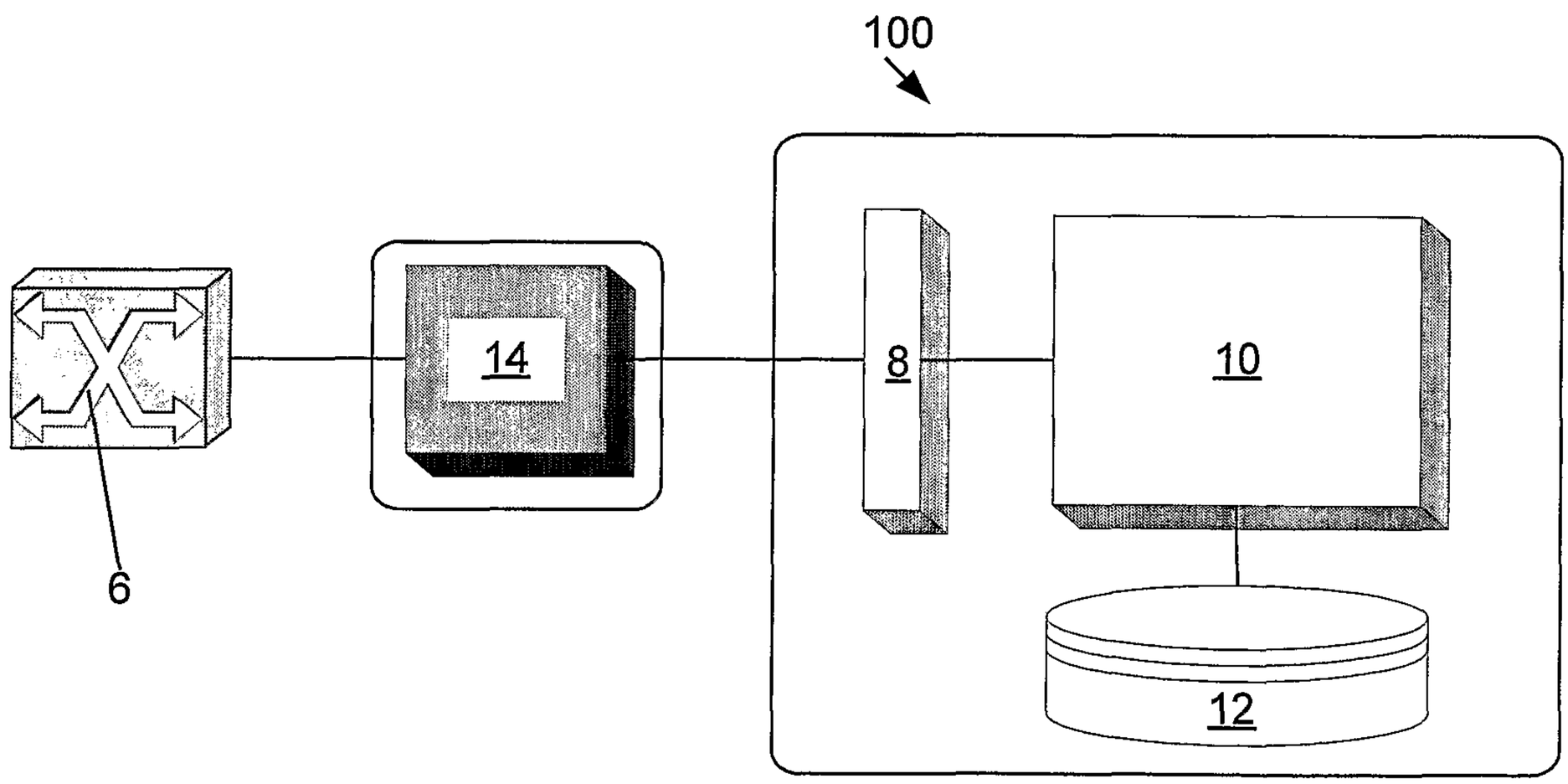


Figure 2

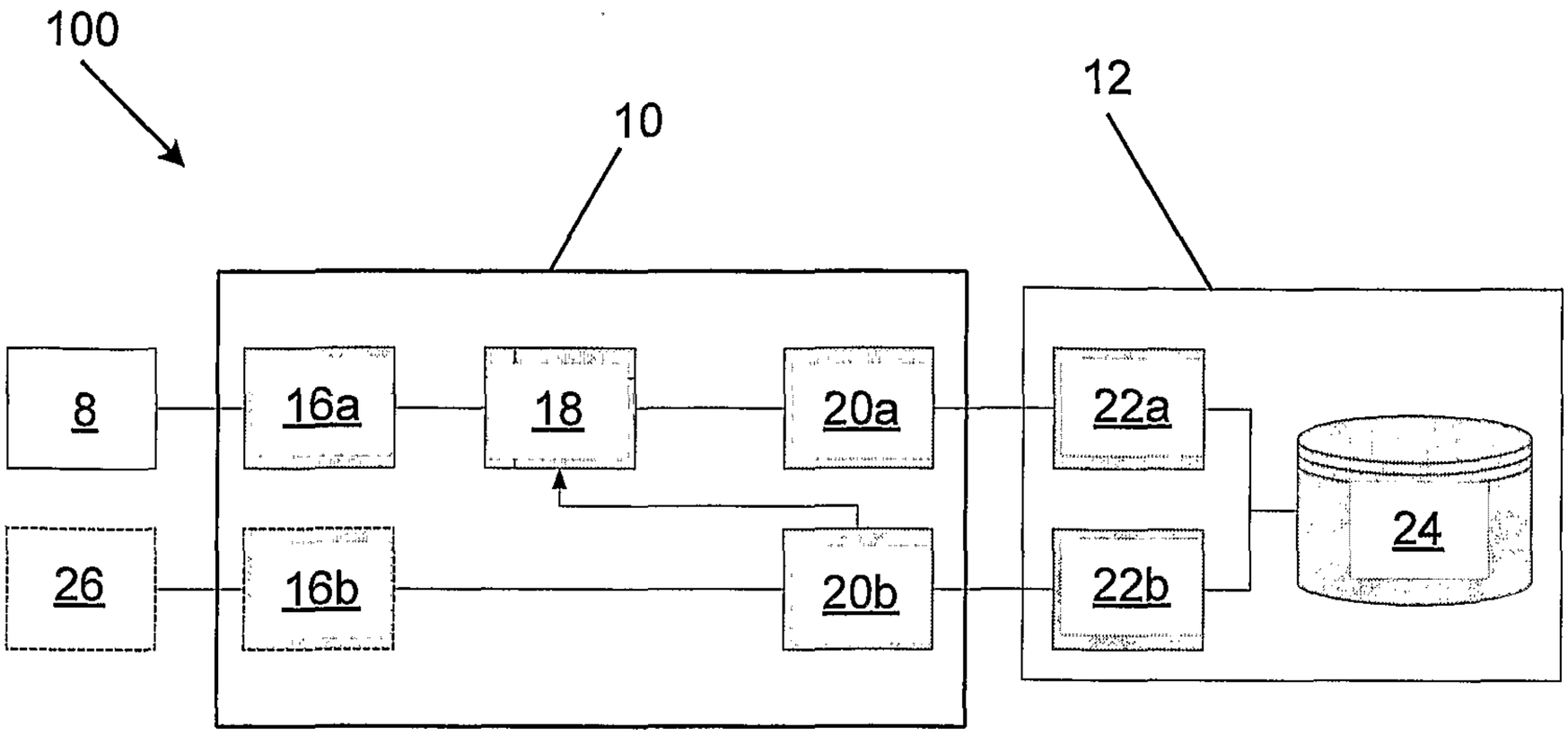


Figure 3

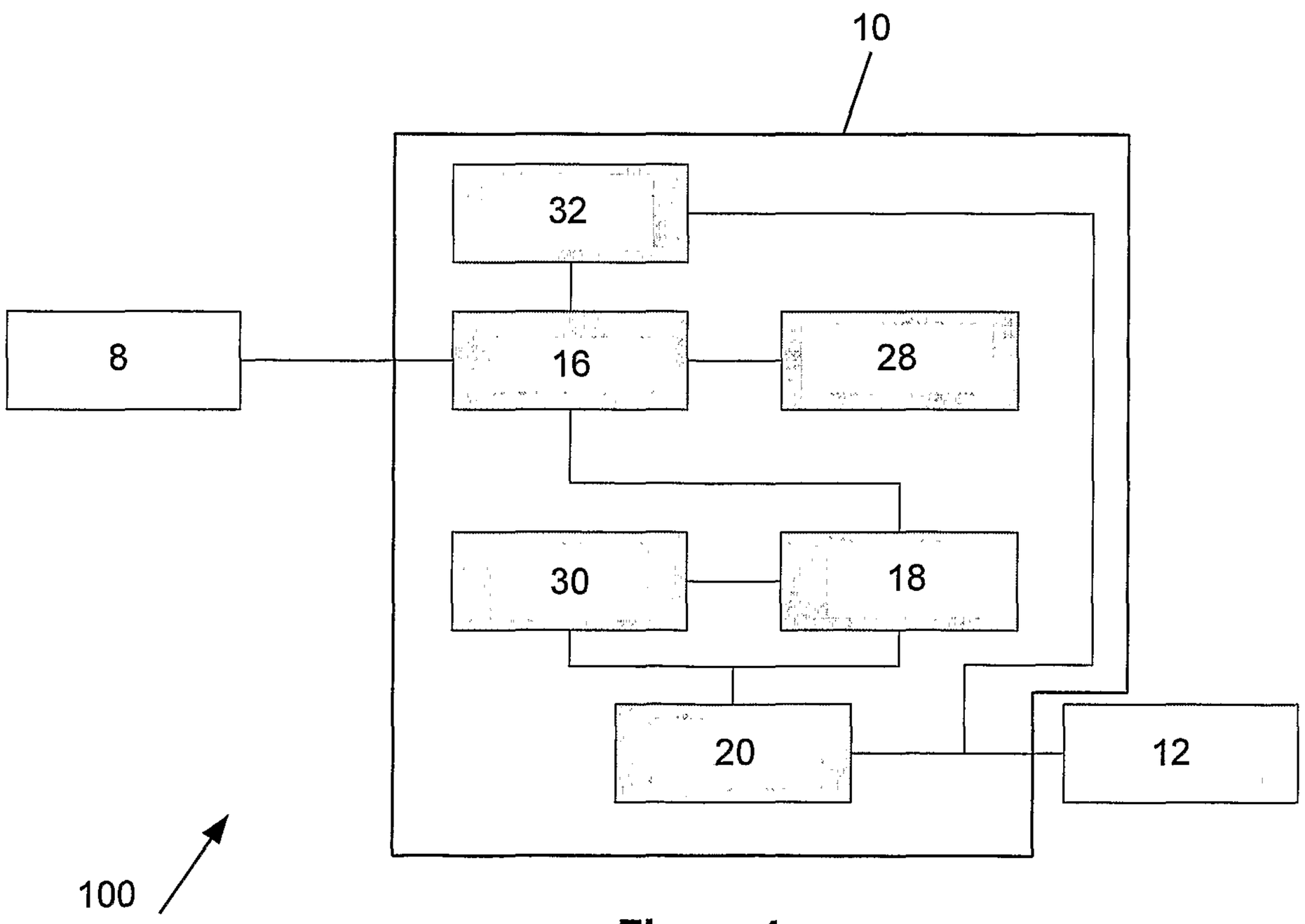


Figure 4a

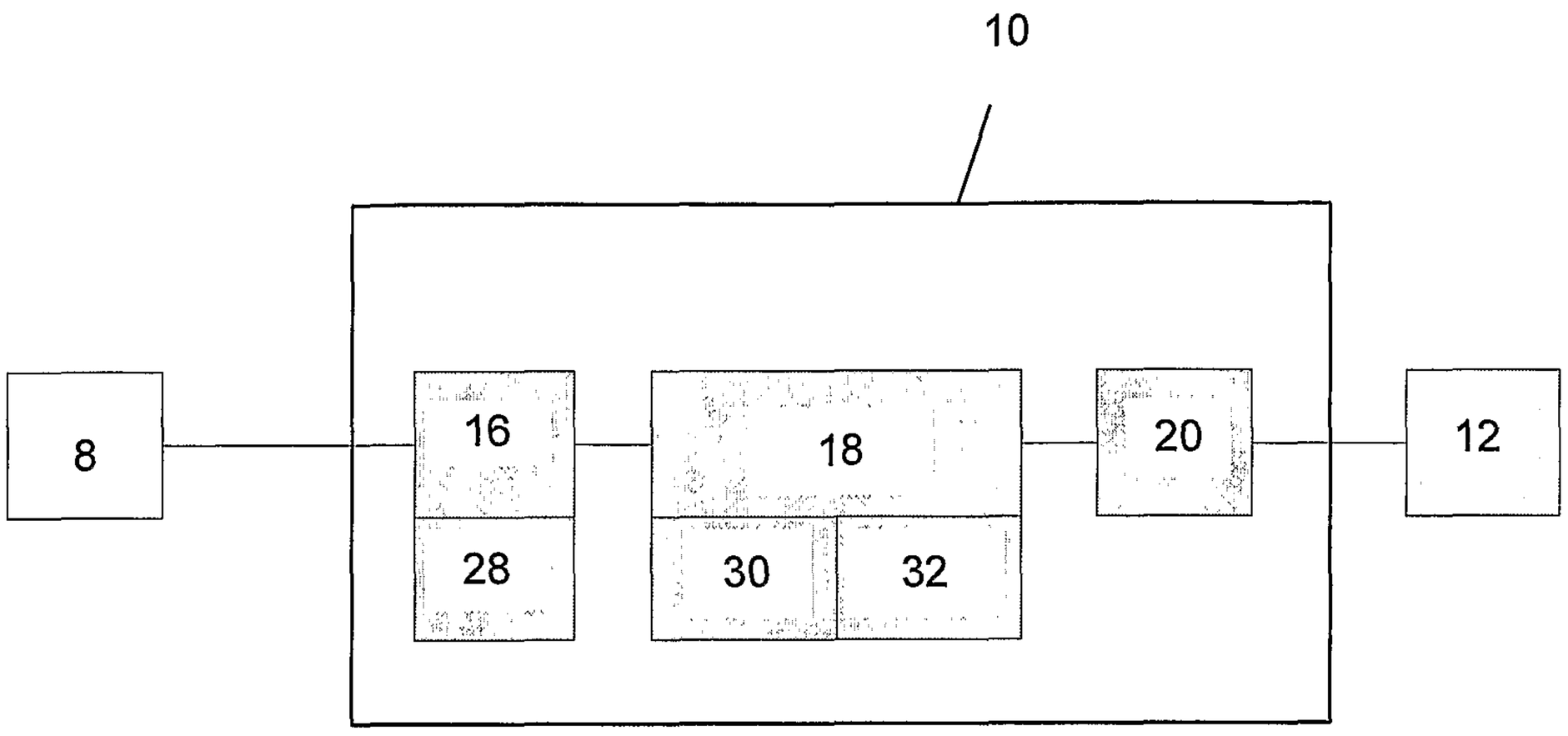


Figure 4b

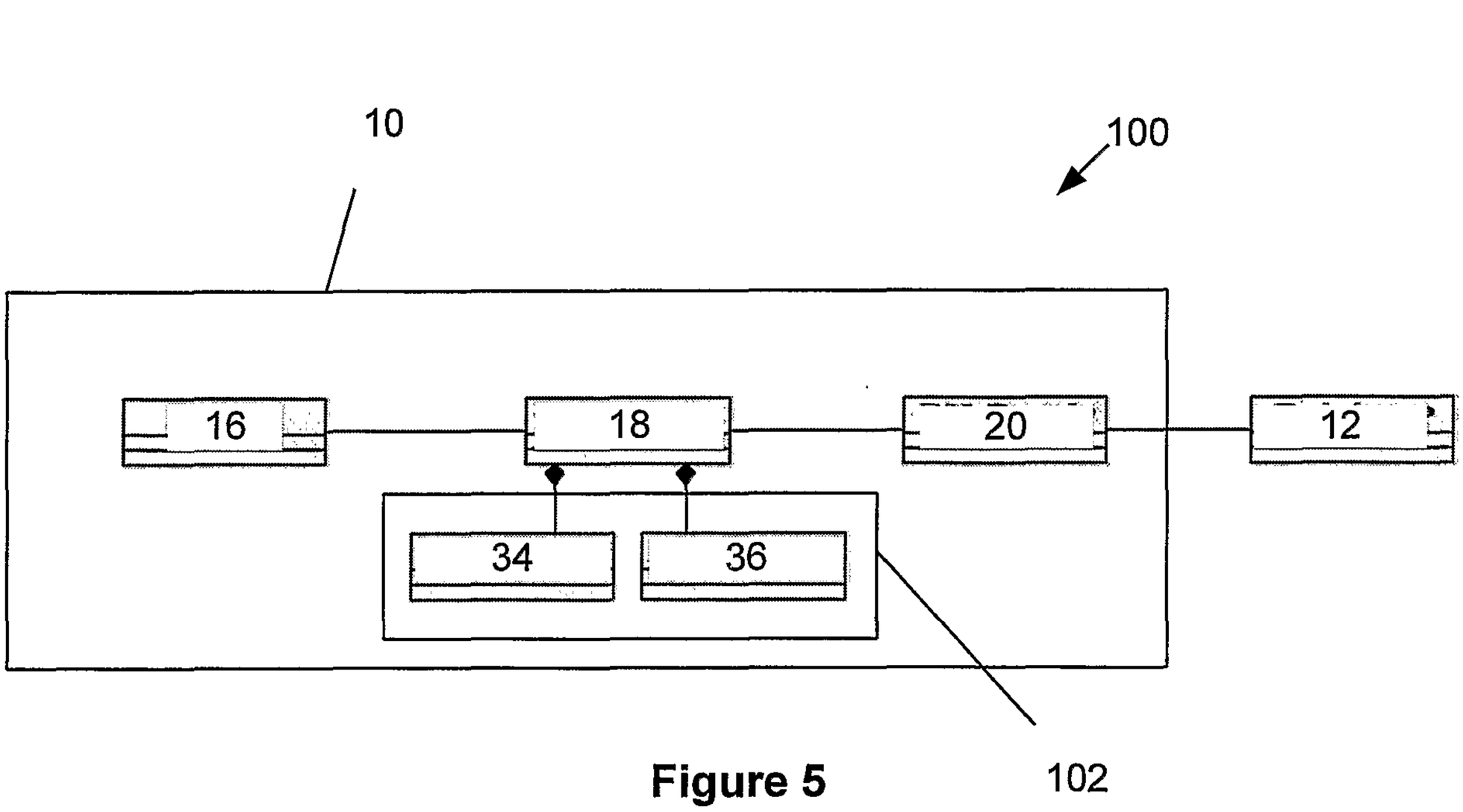


Figure 5

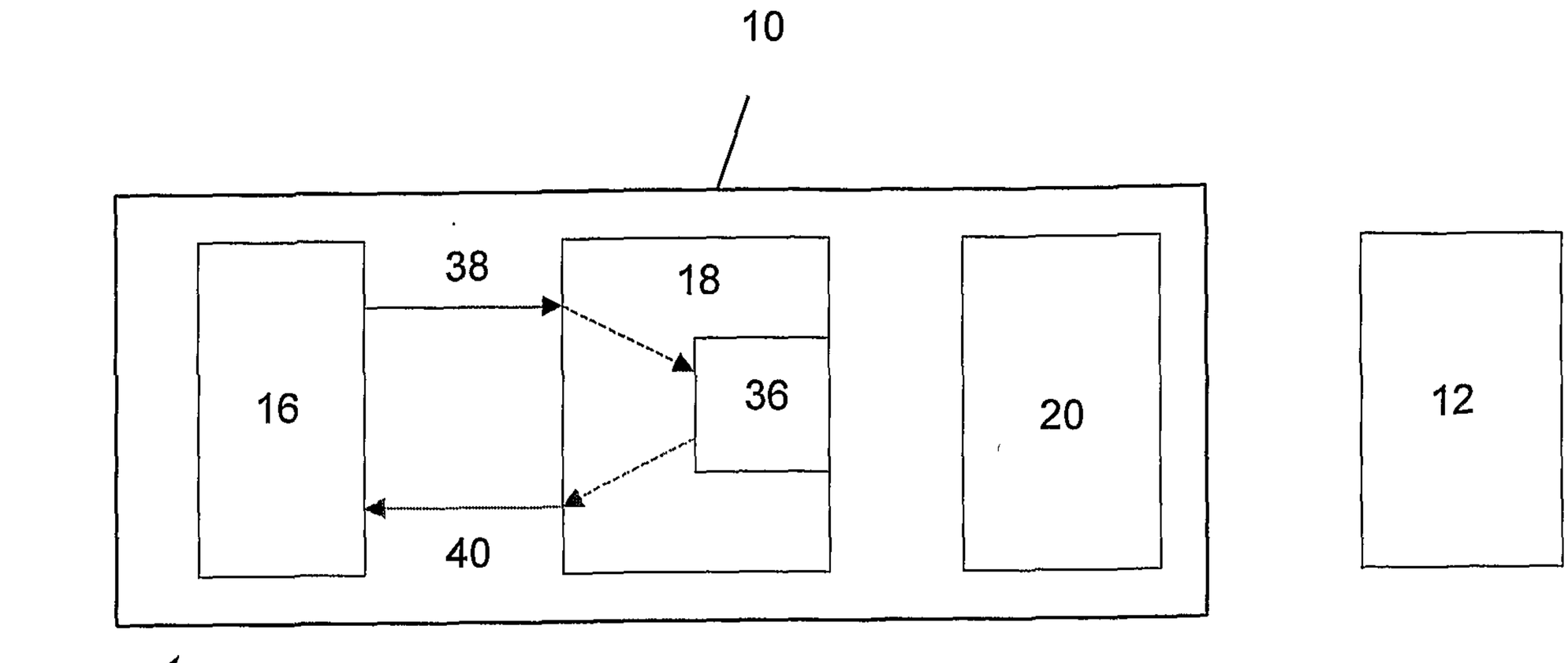
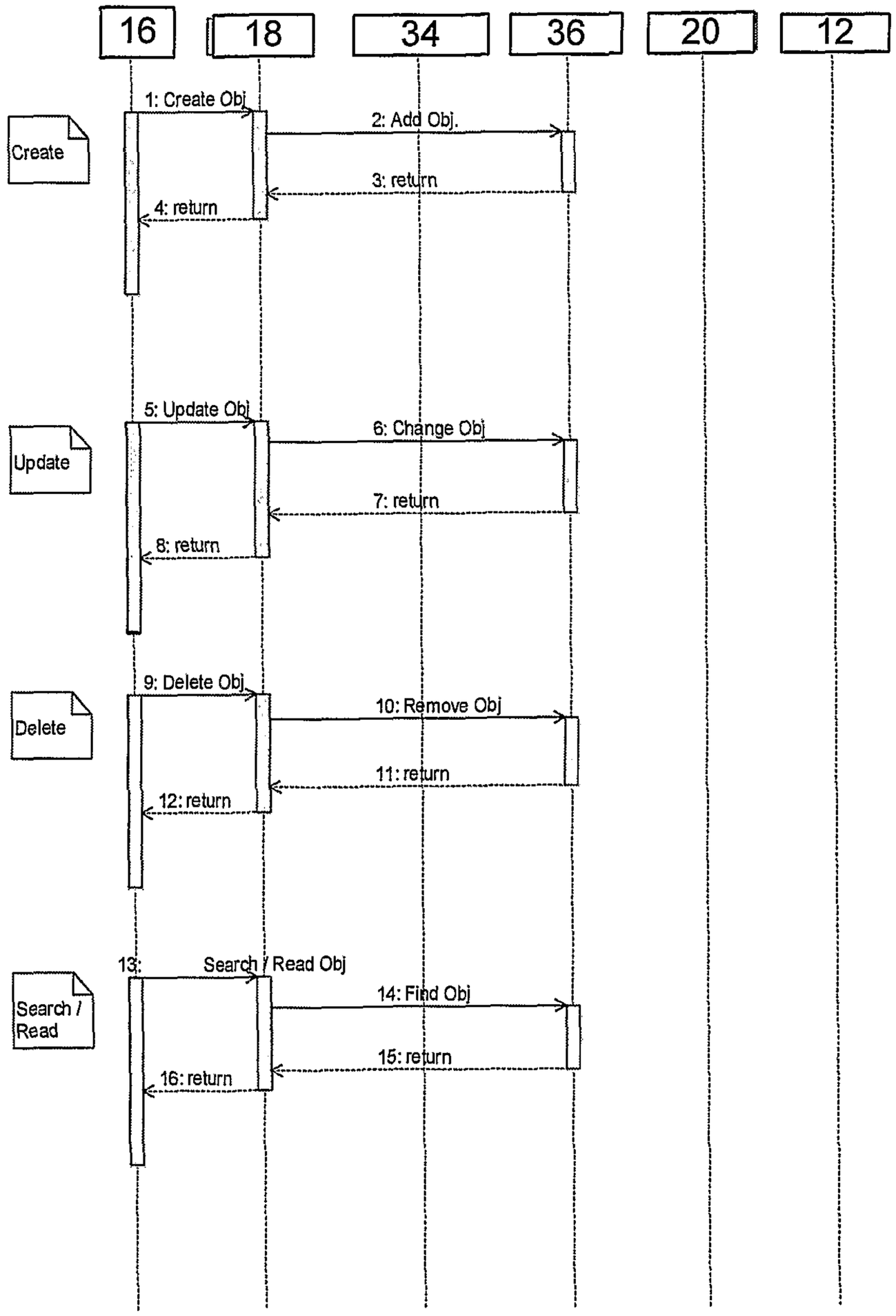


Figure 6

Figure 7



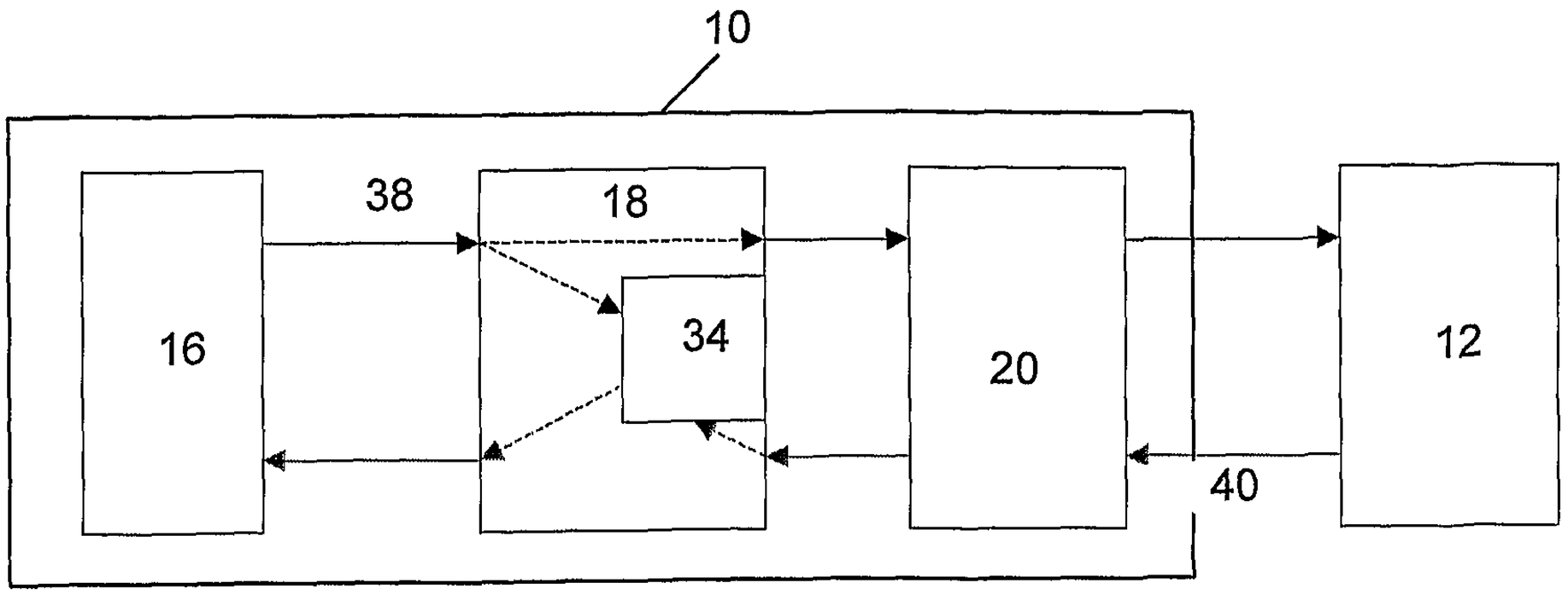


Figure 8

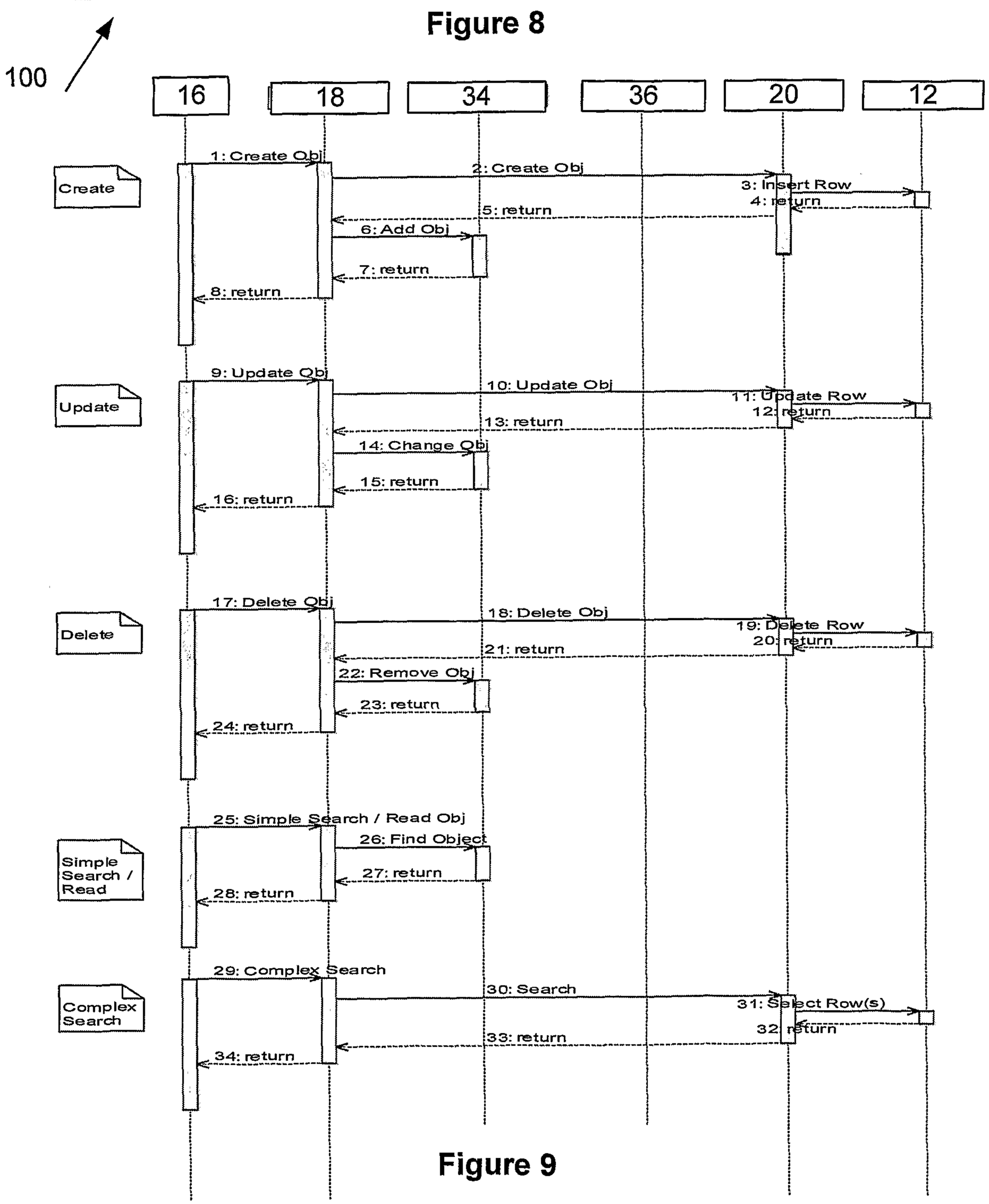


Figure 9

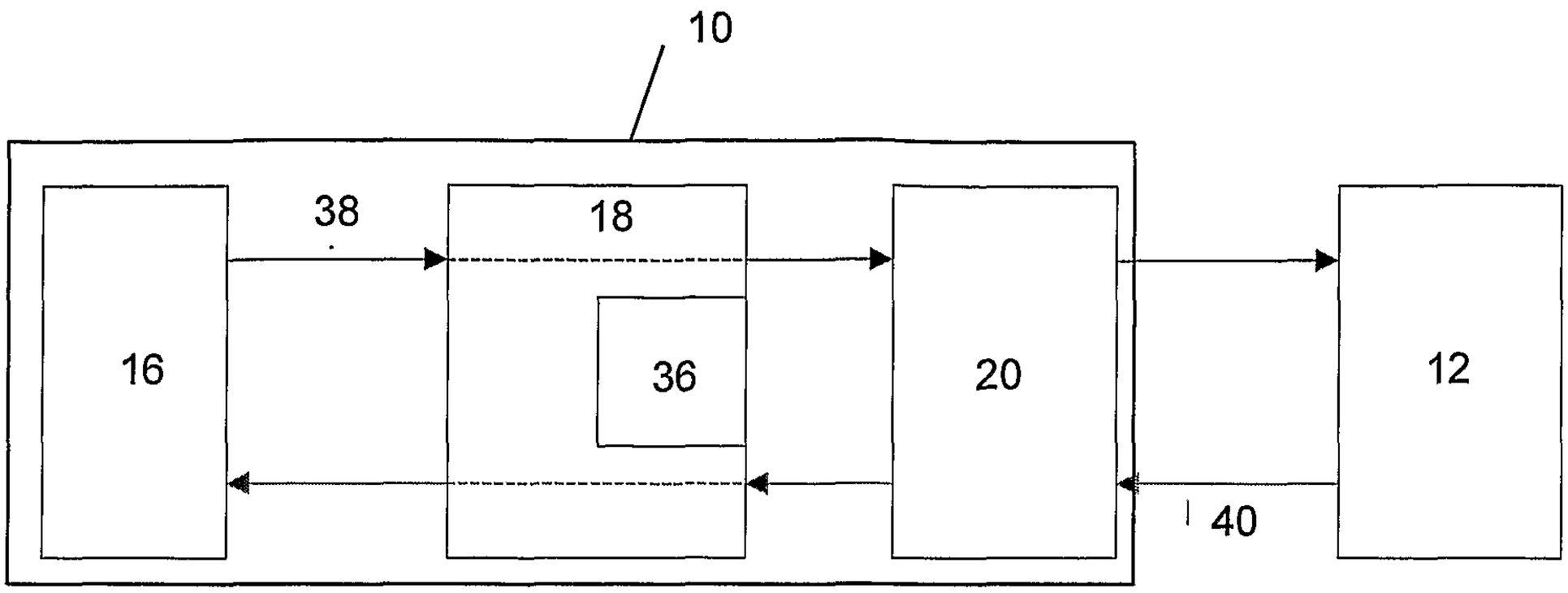


Figure 10

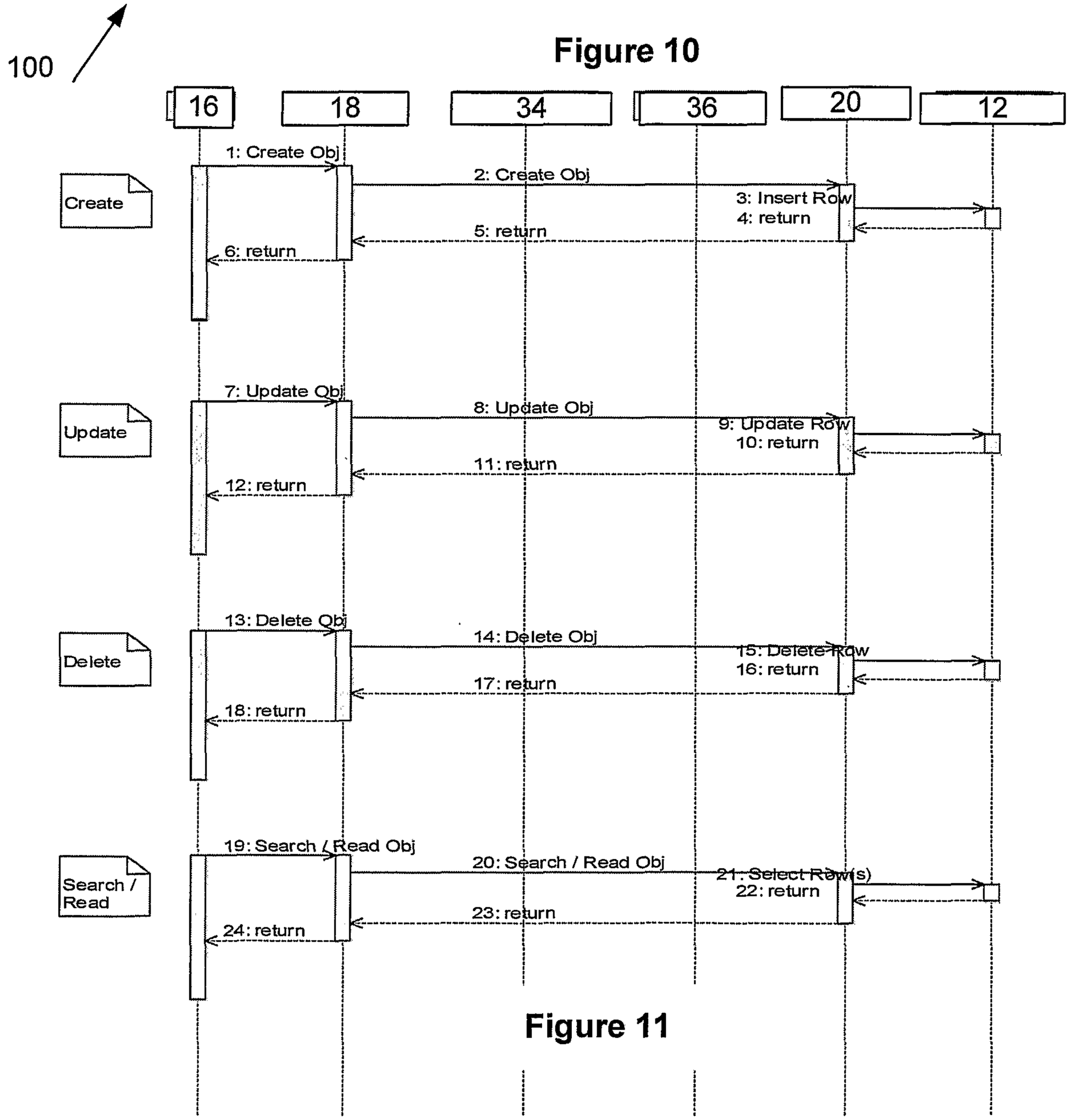


Figure 11

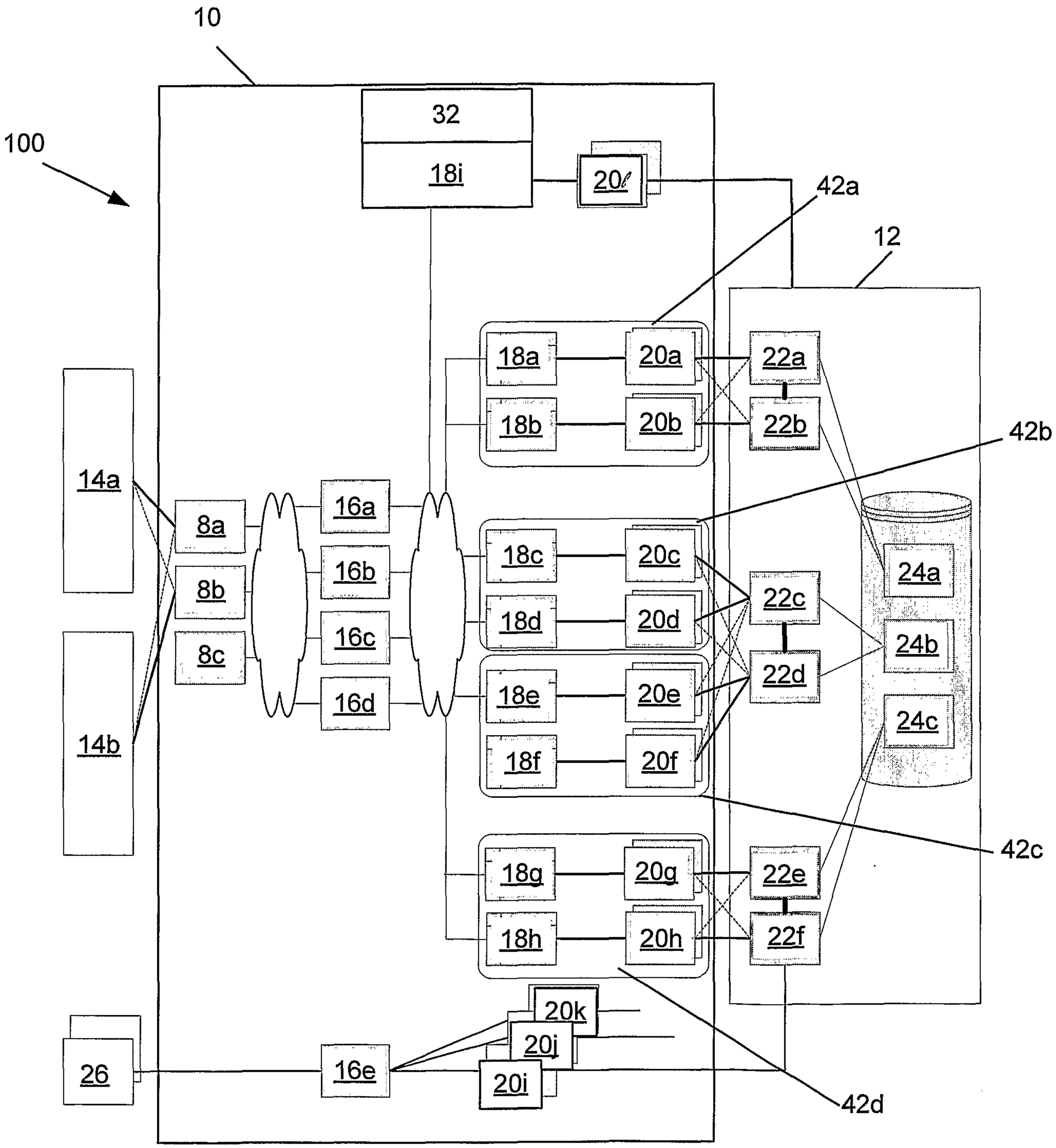

10
100
32
18i
20ℓ
42a
12
18a
20a
22a
18b
20b
22b
42b
14a
8a
16a
24a
18c
20c
22c
8b
16b
24b
18d
20d
22d
8c
16c
24c
18e
20e
14b
16d
18f
20f
42c
18g
20g
22e
18h
20h
22f
20k
20j
26
16e
20i
42d

Figure 12a

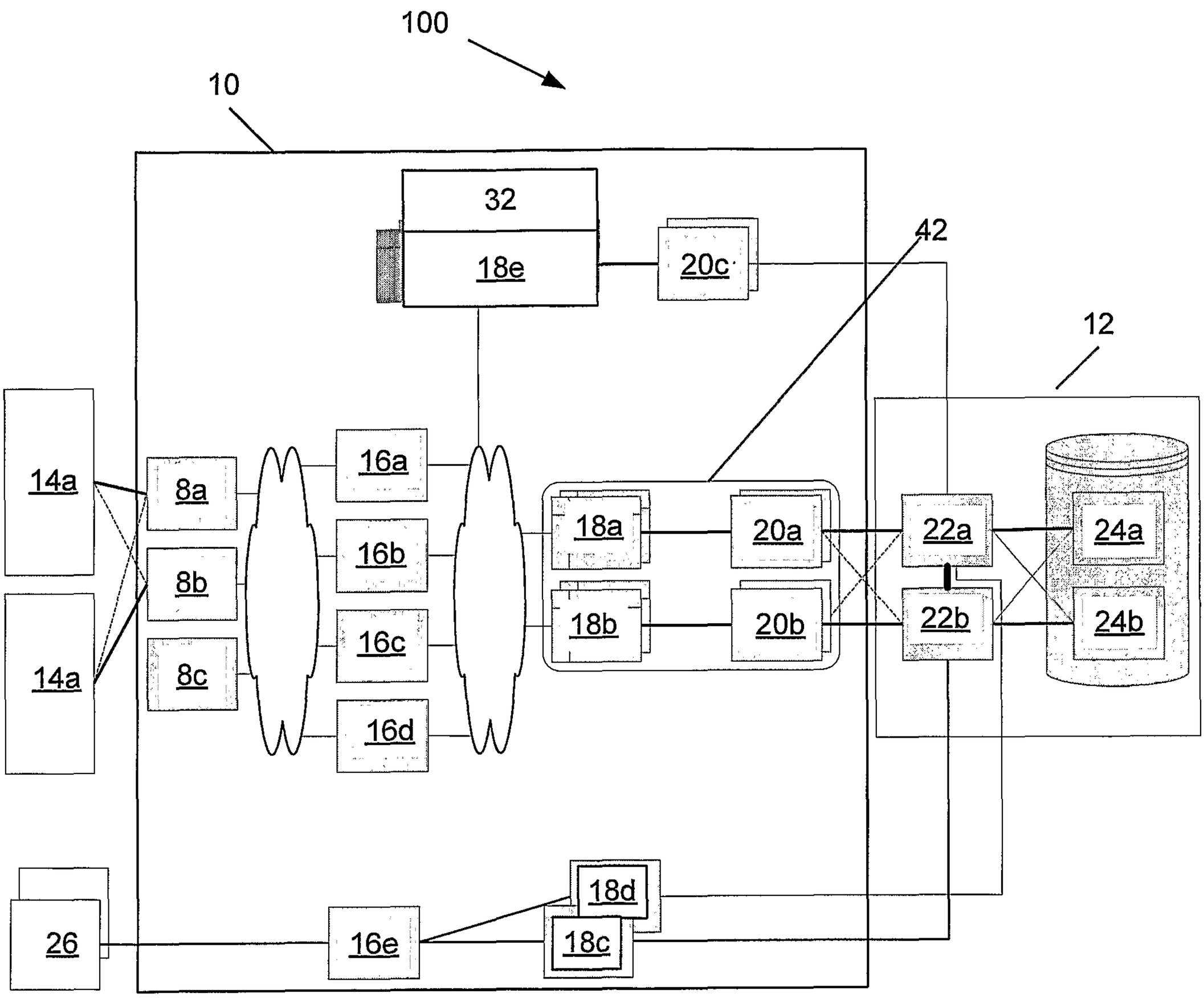

100
10
32
18e
20c
42
12
14a
14a
8a
8b
8c
16a
16b
16c
16d
18a
18b
20a
20b
22a
22b
24a
24b
26
16e
18d
18c

Figure 12b

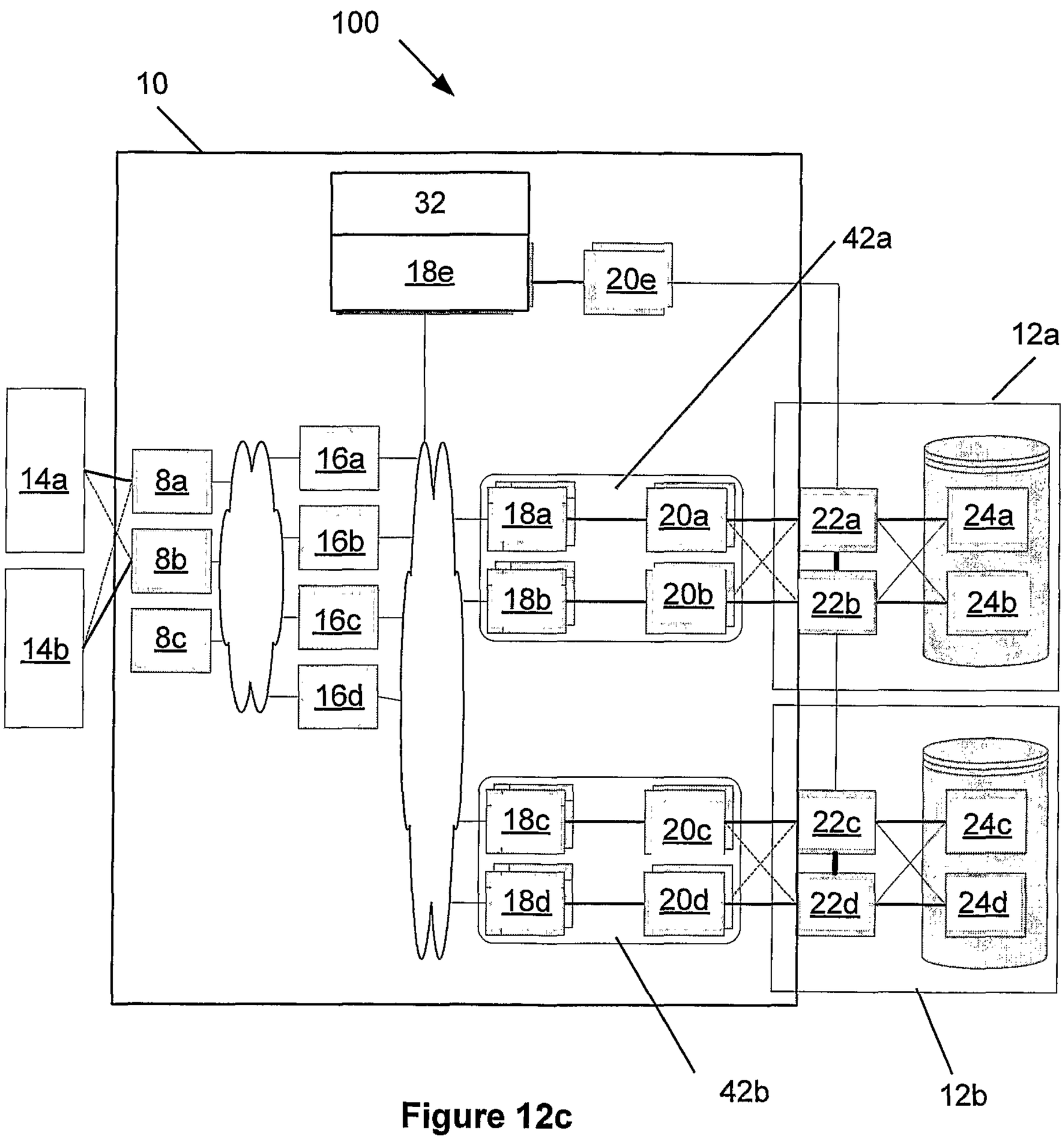

100
10
32
18e
20e
42a
12a
14a
14b
8a
8b
8c
16a
16b
16c
16d
18a
18b
18c
18d
20a
20b
20c
20d
22a
22b
22c
22d
24a
24b
24c
24d
42b
12b

Figure 12c

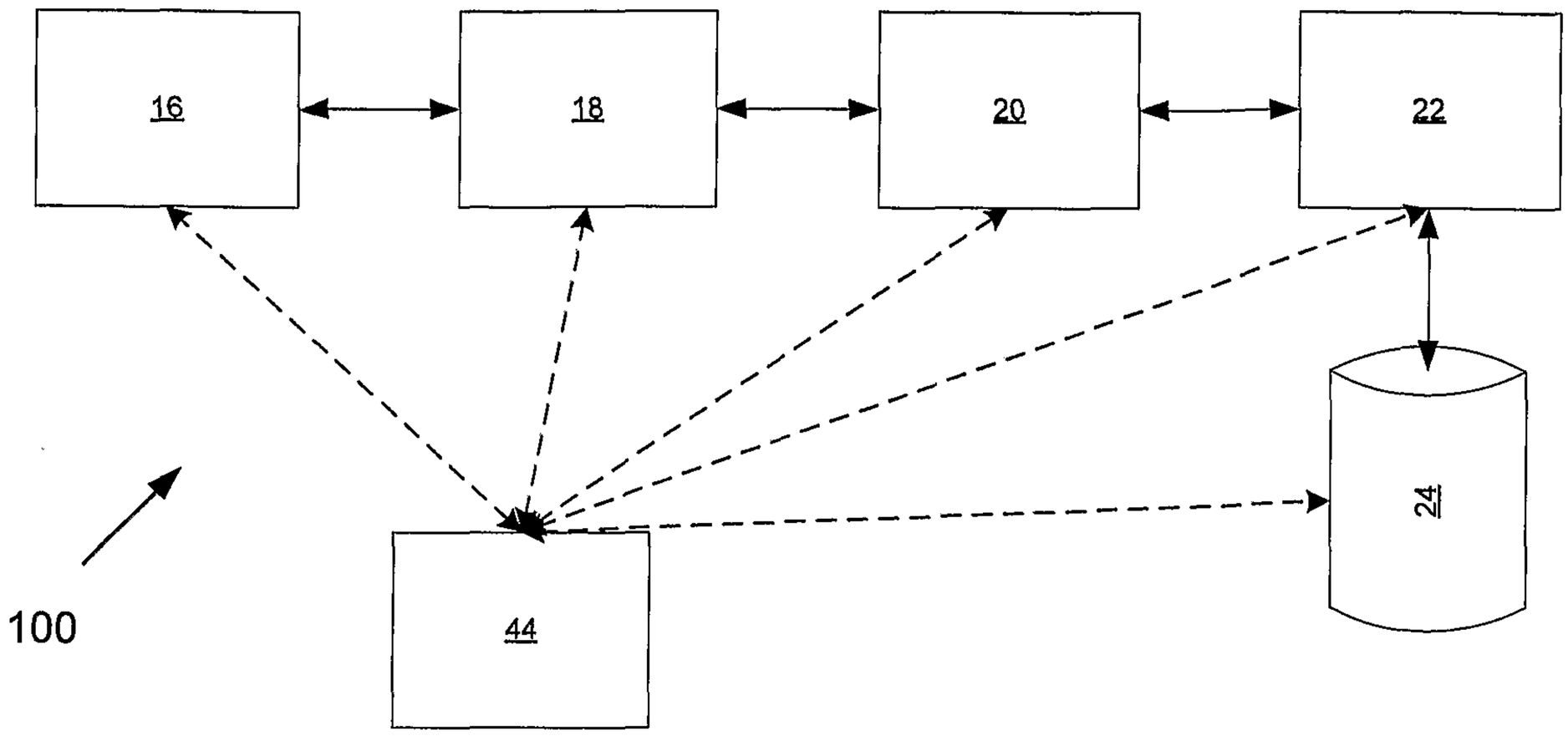


Figure 13

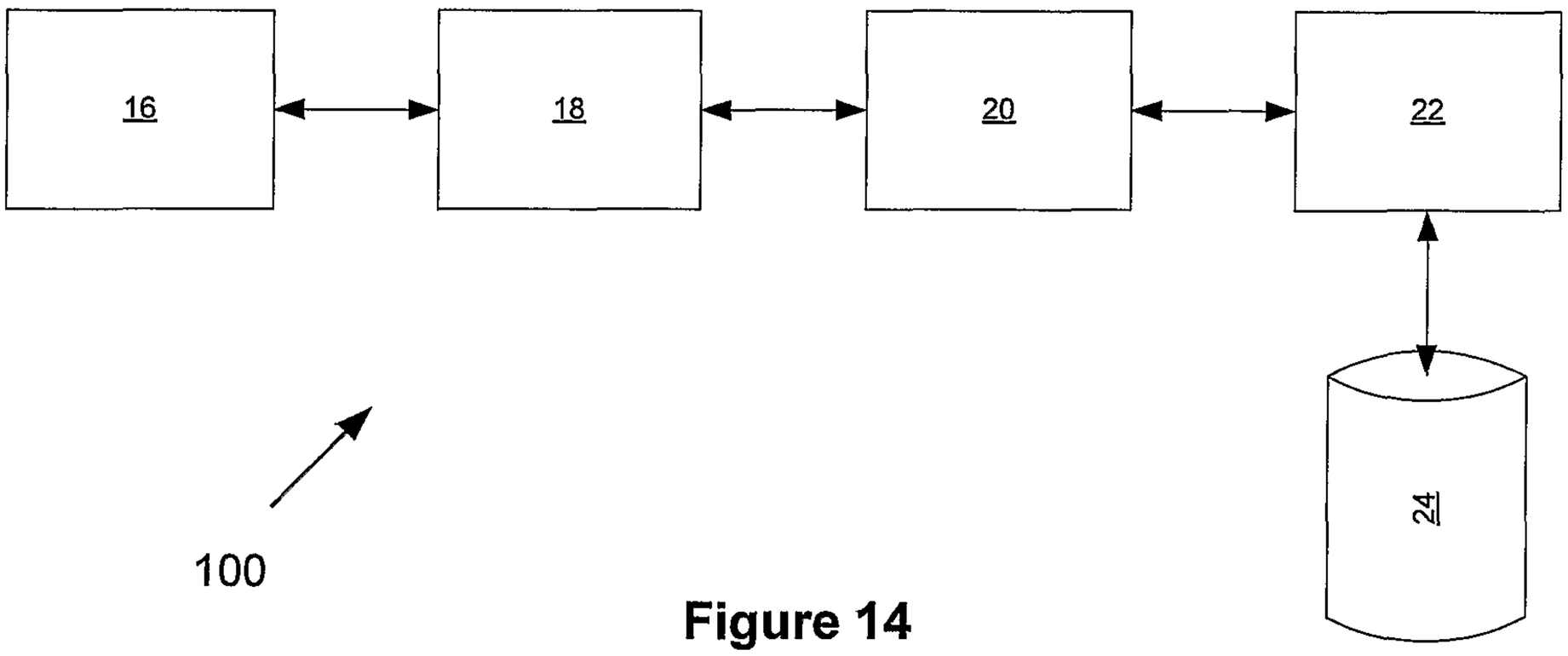
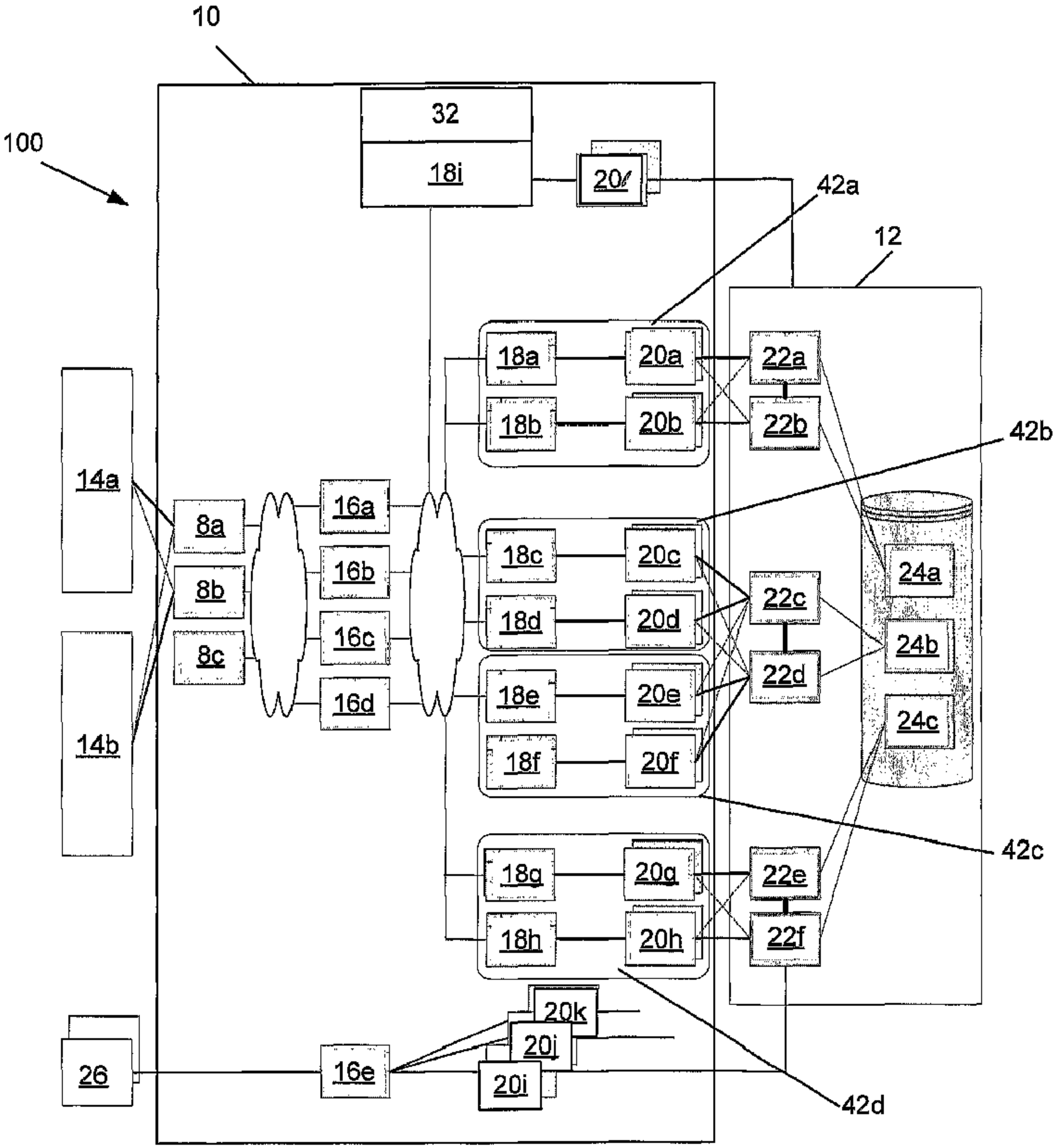


Figure 14


10
100
32
18i
20i
42a
12
18a
20a
22a
18b
20b
22b
42b
14a
8a
16a
18c
20c
24a
8b
16b
22c
18d
20d
24b
8c
16c
22d
14b
16d
18e
20e
24c
18f
20f
42c
18g
20g
22e
18h
20h
22f
20k
20j
26
16e
20i
42d