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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 August 2015 (13.08.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/119934 A1

(51) International Patent Classification:
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/US2015/014219

(22) International Filing Date:
3 February 2015 (03.02.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/935,784 4 February 2014 (04.02.2014) US

(72) Inventor; and
(71) Applicant : **SARKAR, Dipankar** [US/US]; 13044 Houston Court, Saratoga, California 95070 (US).

(74) Agent: **ZWEIG, Stephen**; 224 Vista de Sierra, Los Gatos, California 95030 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, 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, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR RELIABLE MULTICAST DATA TRANSPORT

(57) Abstract: A system and method of providing a reliable and efficient multicast data transfer mechanism in a communication network. The mechanism includes a plurality of computer nodes and one or more data system managers wherein each of computer nodes and each of data system managers are connected through the said communication network. The method generally comprises the steps of setting up of an association among one or more computer nodes as senders and one or more computer nodes as receivers using a combination of unicast and multicast protocols and transmitting one or more packets of data through the said multicast protocol by one or more senders to one or more receivers. The receivers may collaborate among themselves to ensure delivery of said one or more packets of data reliably to the collaborating receivers.

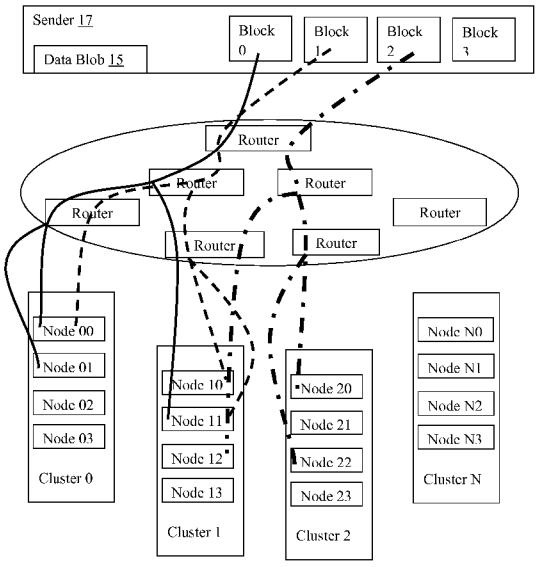

FIG.4

WO 2015/119934 A1

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

SYSTEM AND METHOD FOR RELIABLE MULTICAST DATA TRANSPORT

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of US Provisional Application No. 61/935,784, entitled "Multicast Replication Transport for Distributed Block Storage", Inventor Dipankar Sarkar, filed February 04, 2014, the entire contents of which are incorporated herein by reference.

FIELD OF THE INVENTION

The present invention relates to the field of Network connected Data Transfer System. More specifically, the present invention provides a reliable multicast data transport mechanism.

BACKGROUND OF THE INVENTION

Electronic data systems are implemented using software on a variety of hardware components like computers, disk drives, solid state memory and networking switches. Most data systems store their data in block format where, the sequence of bytes are broken into fixed size blocks and stored in the underlying storage media blocks. This is usually the case as the underlying physical storage is organized into blocks. Read and writes happen in chunks of each data block. The blocks may not be laid out in sequence in the storage media but would be logically linked to form each contiguous file or data blob. The last block may be partially filled. These blocks are stored in the media based on some form of block organization. There is an overlaying data management layer that maintains meta information for the files or data blobs and mechanisms to deploy and retrieve the blocks as required. In a network connected distributed storage, the blocks are spread over clusters of computer nodes connected by data networks. For reliability and disaster recovery, clusters can be distributed over many geographic locations. The blocks are distributed and replicated over these clusters based on various policies specified for that installation. Usually, this includes the same block saved in multiple different computer nodes of geographically separated clusters.

The replication is done to create data redundancy and load balancing. The replication is usually designed to serve two purposes. First, the data is made available closer to the processing units. Second, the replication is such that even if a catastrophe were to strike one location, the data would still be preserved in another location. The data management system

can then work around the lost data blocks. It would then replicate those blocks on other working clusters.

The replication process needs to be reliable as data integrity and data preservation is of utmost importance. The communication mechanism over the network between the nodes has to be reliable. Currently, block storage mechanisms use unicast streams to replicate data across the various computer nodes that are selected for replication. The mechanism is called pipelining. In this mechanism, when the client wants to write data, it queries the data management system for a list of computer nodes where the data is to be replicated. It receives a list of computer node information where the data would be replicated. This is called the pipeline. The client then opens a point-to-point connection to the first computer node in the list. It passes the list to the next computer node in the pipeline and streams the block data to it. The first computer node then opens a second point-to-point connection to the second computer node and streams the block data to it, and so on it goes. The last block receiving the data sends an acknowledgement. The acknowledgement is then cascaded back to the client through the reverse sequence of computer nodes in the pipeline. Alternatively, the sender can open multiple point-to-point connections and unicast the data over these connections.

There are other kinds of replication like Master-Slave configuration and Multi-Master replication where the same data needs to be transmitted to multiple database servers. Such scenarios can benefit from a fully reliable multicast data transfer.

In a multi-user network based remote conferencing system, some of the data from one participant would need to be transmitted to multiple participants. Such a use case can also benefit from a fully reliable multicast data transfer.

Multicast is a class of communication where one entity sends the same data to multiple entities in a single transmission from itself. The data can be sent with multiple transmissions at points that fork into multiple independent paths. The transmission takes on different forms depending upon the underlying network. There is Internet Protocol multicast, ISO CLNP multicast, Network-on-chip multicast and Ethernet multicast and Infiniband multicast. Multicast datagram is normally an unreliable protocol. Where the requirements are strict reliable, reliable unicast mechanisms are used like TCP, TP4 etc. Multicast is used for distributing real time video where it needs to scale with the number of receivers. Limited loss of data show up as glitches and the communication moves on.

Where data needs to be transmitted from one source to multiple receivers, use of multicast transmission is an obvious idea. The validity and viability of a solution based on multicast transmission depends upon the speed, reliability and scalability with error and loss recovery. Reliability using multicast is a domain specific problem. There is no one-solution-fits-all available. Different situations have different types of reliability requirements. Real time audio and video delivery requires sequenced delivery but small amounts of lost data is less important. Small segments of lost data will cause only a slight jitter in the audio or video. In cache updates, time synchronization is more important as validity of cache is important for quick changing data. In data replication, sanctity of the data is more important than speed.

The reliability conditions over wide area networks are different than over local networks. If any of the multicast paths traverse over a wide area network, the issue becomes very important. Over a wide area network, the possibility of packet fragmentation increases. At higher data rates, the possibility of data misalignment during reassembly increases. The number of fragmented packets that can be present in the network at any instance of time is limited by the size of the packet identifier and the data rate. This is described in RFC4963. For IPv4 the packet identifier field is 16 bits. This allows only 64K packets of any protocol between two IP address pairs during a maximum per maximum packet lifetime. At 1Gbps rate, it takes less than one second to fill up this count. Layer 4 checksum can be used to detect and discard wrongly reassembled packets. With a checksum field of 16 bits and well distributed data, the failure rate of layer 4 in filtering out bad datagrams is 1 in 64K. It improves with larger size checksum like 32 bit. Some firewalls allow only known protocol types to pass through. So, many multicast applications tend to use User Datagram Protocol (UDP) which has a checksum size of 16 bits. This analysis indicates that for big data kind of usage, direct interfacing with the network layer with a higher size checksum would be a better option.

Multicast has been used in the distributed file systems to transmit the data to the client side caches. JetFile and MCache are examples of this. JetFile is a distributed file system similar to NFS in functionality. It maintains cache of the files at the computer nodes requesting the file. The files are distributed using Scalable Reliable Multicast (SRM) protocol. In the JetFile system, the sender has no knowledge of the receivers. The sender sends the data to the group multicast address. The receivers are clients who serve files as a peer-to-peer network. Multicast is an unreliable delivery mechanism. In the above two cases, if any receiver does

not get the data, there would not be any damage. A retry will fetch the data with a slight delay. If data caches do not receive the data, it will only delay the fetching of data, not cause a data loss. The problem of data loss can be somewhat mitigated by using published algorithms like SRM & PGM but not completely solved. In all of these algorithms, the responsibility of getting all the data lies completely with the receivers. If any or all receivers fail to get the complete block of data, the sender will never know. In the case of block replication, that would be a failure of operation. In case of data replication, the sender needs to know of any data loss and take corrective action.

Encrypted UDP based FTP with multicast (UFTP) uses multicast to send files to a group of receivers. In this protocol, the sender breaks the data into a sequence of transmittable blocks. Each block has a sequence number. The blocks are grouped in sections. The sender transmits all the blocks in a section and then waits for the negative acknowledgement (NAK) from the receivers. For every block that it receives a NAK, it retransmits the block. If it does not receive any NAK, it closes the session. Again the problem is, if a NAK is lost or receivers fails to get the data, the sender will not get to know. Also, if the NAKs are sent at the end of a big section transfer, it poses a burden on the sender. The sender needs to preserve all the transmitted packets holding up memory or recreate the lost packet by streaming through the original data. This is good for occasional transfer like end of day updates to remote sites. For high load of simultaneous occurring transfers, this can exhaust system resources.

In Distributed File Systems, like Hadoop Distributed File System, there is a need for bytes constituting a file block to be delivered sequentially and reliably. No existing reliable multicast transport has been able to fulfil that requirement. So, such file systems continue to use multiple reliable unicast point-to-point links using Transmission Control Protocol (TCP) till date.

Accordingly, there exists in the art a need for a method for a reliable multicast data transfer with better error recovery and faster loss recovery mechanisms in network connected data distribution systems.

SUMMARY OF THE INVENTION

The following presents a simplified summary in order to provide a basic understanding of some aspects of the disclosed invention. This summary is not an extensive overview, and it is

not intended to identify key/critical elements or to delineate the scope thereof. Its sole purpose is to present some concepts in a simplified form as a prelude to the more detailed description that is presented later.

The present invention provides a more efficient transport by creating a reliable multicast protocol.

One object of the present invention is to provide the existing replication mechanisms of distributed block storage data systems, a faster transport using multicast protocol at the network layer. The network layer can be Internet Protocol multicast, ISO CLNP multicast or Network-on-Chip multicast or any such multicast network.

In this scenario proposed by the present invention, there are one or more senders and multiple receivers. Usually, the receivers are part of the same administrative domain. The receivers can collaborate amongst themselves. So, the unique characteristic of this situation is that the data transfer has to be reliable not to any one receiver or all the receivers but to the aggregate of all the receivers as a whole. This is another object of the present invention.

These as well as other objects of the present invention are apparent upon inspection of this specification, including the drawings and appendices attached hereto.

BRIEF DESCRIPTION OF THE DRAWINGS

In order to describe the manner in which features and other aspects of the present disclosure can be obtained, a more particular description of certain subject matter will be rendered by reference to specific embodiments which are illustrated in the appended drawings. Understanding that these drawings depict only typical embodiments and are not therefore to be considered to be limiting in scope for all embodiments, various embodiments will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

FIG.1 is an illustration of a prior art network connected distributed data system;

FIG.2 illustrates the logical modules in a typical setup of a distributed data system in accordance with an embodiment of the present invention;

FIG.3 illustrates an exemplary method for data block distribution among different computer nodes in a distributed data system;

FIG.4 illustrates multicast data flow paths when the data traverses the network for replication of the blocks in a distributed data system in accordance with an embodiment of the present invention;

FIG. 5 illustrates an initial scenario of a data replication use case as an exemplary embodiment of usage of the present invention;

FIG.6 illustrates an initial scenario of an online conference use case as an exemplary embodiment of usage of the present invention;

FIG.7 illustrates an association setup for data transfer between a sender and a data system using a combination of unicast and multicast protocols in accordance with an embodiment of the present invention;

FIG.8 illustrates a data write process and the data transfer process using multicast protocol to the computer nodes in accordance with an embodiment of the present invention;

FIG.9 illustrates a process of error recovery from data loss in accordance with an embodiment of the present invention;

FIG.10 illustrates a process of tear down of the association set up for a data transfer in accordance with an embodiment of the present invention;

FIG.11 illustrates the essential elements of the header of the protocol in accordance with an embodiment of the present invention;

FIG.12 illustrates the design of a typical fragment of the state machine in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

FIG.1 is an illustration of a network connected distributed data system (100) generally known in the art. The system 100 includes one or more network hosts such as Client (1) and plurality of Clusters (6) each comprising plurality of Computer Nodes (5) located at same or different

geographic locations etc. The system 100 also includes one or more Data System Managers (2) running on computers, a communication network (3) and other necessary networking devices such as plurality of Network Switches (4) etc. The Client (1) can be a reader/receiver or writer/transmitter. Client (1) accesses the Data System (100) over the Network (3) via the Application Programmer Interface (API) provided by the Data System Manager (2). There are clusters (6) of computer nodes (5) in various geographic locations. Some of the clusters (6) can be located in the same data center and some in remote locations. The above modules connect to the communication network (3) via the nearest Network Switch (4).

FIG.1 illustrates just an example of a suitable computing system environment in which the present invention may be implemented. The computing system environment is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment be interpreted as having any dependency requirement relating to any one or combination of components illustrated in the exemplary operating environment.

The present invention is operational in numerous other general purpose or special computing system environments or configurations other than shown in FIG.1. Examples of well known computing systems, environments, and/or configurations that may be suitable for implementing the invention include, but are not limited to personal computers, server computers, laptop devices, multiprocessor systems, microprocessor-based systems, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or the like.

FIG.2 illustrates the logical modules of network distributed Data system (100) in a typical setup employing the principles of the present invention implemented through a combination of hardware and software components. The hardware components are programmed to execute one or more instruction for the data transfer mechanism by using the software components proposed by the present invention. The Data System Manager (2) has a module called Data Manager (7) that manages the attributes and meta-data of the files or data blobs. The Block System Manager (8) is responsible for distributing and replicating the data across the blocks. Each Computer Node (5), apart from the standard modules of a computer, has various modules dealing with the block management. Typically, it would have a Control Module (9) responsible for listening to instructions from the central Block System Manager (8) and acting upon them in the computer node. If a new address needs to be listened on for block

data, the Control Module (9) instructs the I/O Module (10) to listen and receive data. The Block Management module (11) is responsible for managing the Data Blocks (12) within the Computer Node (5). The I/O Module (10) receives the incoming block data and moves it to the appropriate blocks. All the communication among Client (1), Computer Node (5) and Data System Manager (2) and inter-node communication happen using a set of Transport Protocols (13). This set of Transport Protocol (13) depends upon the type of underlying network. FIG.2 also illustrates the introduction of an Address Allocator Module (14) in accordance with an embodiment of the present invention. The Address Allocator (14) is responsible for maintaining a list of multicast addresses and if needed, corresponding Transport layer Service Access Point (TSAP).

FIG.3 illustrates an example of how the data blocks (16) of a file/data blob (15) are distributed around amongst the computer nodes (5). The Data Manager (7) has, amongst its meta-data, a pointer to its various data blocks (40). These blocks (16) have the pointer to the various locations where the block has been stored. In the example shown in FIG.3, the replication count is three and so, each of the blocks (16) are copied in three different locations in various Computer Nodes (5). If a Computer Node (5) were to go down, the blocks (12) stored there would get replicated in other Computer Nodes (5).

The present invention proposes modification to the multicast address allocator modules in the Block System Manager (8) or in its equivalent in other embodiments like Conference Member Manager (68) as shown in FIG.6 to maintain a cache of the group of computer nodes and the corresponding multicast address. For address allocation to the same group of computer nodes, the recently used multicast address should be re-allocated. Since the network switches and routers already have the routes in their cache, significant efficiencies will be achieved at the network level.

The present invention is different from Scalable Reliable Multicast (SRM), Pragmatic General Multicast (PGM) and Reliable Multicast Transport Protocol (RMTP) in the sense that they are designed to deliver data to all the receiving group members at a high throughput with the best effort reliability. The present invention is designed to deliver all the data reliably to the aggregate of all receiving group members at the best effort throughput. The present invention is different from Encrypted UDP based FTP with multicast (UFTP) and Xpress Transport Protocol in the sense, the UFTP and XTP receivers do not collaborate and so cannot be treated as an aggregate whole.

SRM and UFTP use only negative acknowledgement mechanism. A negative acknowledgement identifies the data that has not been received. There are two ways to identify data. One is to identify a named portion of data. The second one is to use a sequence number. This invention uses a combination of positive acknowledgement (ACK) and negative acknowledgement (NACK) for reliability. The positive acknowledgement helps the sender to recycle the buffers containing the data it knows it has transmitted successfully. The negative acknowledgement helps in filling up the gaps in the data received and avoid unnecessary retransmissions.

In the preferred embodiment, the protocol proposed by the present invention is run directly on top of the layer providing multicast service like Internet Protocol network layer with configured values including but not limited to the protocol number and checksum size. There can be constraints that may lead to running this invention on top of another datagram service like UDP on top of the multicast layer like IP network layer.

The solution offered by the present invention is in the space of Open Systems Interconnection (OSI) Transport layer as defined by ITU-T X.214, ISO 8072. The terminology, such as T-Data request, T-Data response etc. , used herein are to be interpreted as per definitions provided for those terms in ITU-T X.214, ISO 8072 and adds to the terminology the following two items:

T-ASSOCIATE: The interaction between the sender and the receivers is not strictly connection oriented but a loosely coupled association. The error recovery and flow control functions depend upon the relationship and association parameters. This kind of relationship is termed T-ASSOCIATE.

T-DISSOCIATE: To get out of the relationship state created by T-ASSOCIATE, the T-DISSOCIATE primitive is used. It does a graceful release of the relationship between the sender and one or more receivers.

FIG.5 and FIG.6 illustrate initiation process for reliable multicast data transport as per the present invention in two different exemplary use cases. FIG.5 shows the initiation process for a distributed data system whereas FIG.6 shows the initiation process for an online conference system. Reference to FIG.5, there is a Block System Management Module (8) responsible for block distribution and allocating the multicast address through Address Allocator Module

(14) (shown in FIG.2) for each data transfer association. For the online conference system of FIG.6, the functions of Block System Manager (8) can be carried out by a Conference Member Manager (68) which is a module included in Conference System Manager (2), Conference System Manager (2) being an equivalent of Data System Manager (2). Depending upon the role it plays, at the different phases of the process described herein, any of the computer nodes (5) or Client (1) can act like a Sender/transmitter (17) or a Receiver (18). In the present examples, hereinafter,

Client (1) is referred to as sender (17) as Client (1) is initiating a write operation or an online conference. Reference to FIG.5 and FIG.6, when the sender (17) makes an initiation request i.e. data write request (19) or conference start request (619), Data Manager (7) or conference manager (67) transmits Node Request (20) for obtaining node information to the Block System Manager (8) or to the conference member manager (68). The Block System Manager (8) or the conference member manager (68) works out which computer nodes (5) would receive data and the corresponding multicast network address. If the underlying transport is the network layer, the protocol number would be decided at setup time and configured. If the underlying protocol is an OSI transport layer datagram protocol like UDP or TP0, the port number or TSAP needs to be chosen. The responsibility of assigning these for each session falls on an Address Allocator module (14). This Address Allocator module (14) allocates the service parameters making sure they do not infringe on other associations. The Block System Manager (8) or the conference member manager (68) unicasts this provisioning information Node information (21) back to the sender (17) and also sends multicast address information (22) to the Control Modules (9) in the computer nodes (5) hosting the replication blocks (12) of FIG.3, either directly or through a proxy. If the data transfer needs to be encrypted, the Address Allocator Module (14) also generates one or more encryption key(s) and distributes those with the pipeline information. These provisioning messages i.e. messages (21) and (22) are unicast request-response to prevent leakage and unauthorized distribution. The unicast messages can run on top of a secure client server channel like Transport Layer Security (TLS), Secure Socket Layer (SSL) and HTTPS.

The sender (17) gets a complete list of computer nodes that comprise the replication pipeline through unicast message (21) as shown in FIG.5 and FIG. 6. In a preferred embodiment, a subset of that replication pipeline can be defined synchronous and remaining asynchronous. The synchronous set comprises computer nodes that need to be updated with guarantee. A

failure of that guarantee results in the replication process deemed to be a failure and as a result is aborted.

Reference to FIG.5 and FIG.6, when the Receivers (18), which are plurality of Computers Nodes (5) belonging to various Clusters (6), included in the multicast network address information (22) receive the multicast network address and port information (22), they start listening on that address and port (23). The switches and routers get the multicast address subscription information from the network and build the multicast routes.] Please suggest.

The Sender(s) (17) create(s) a hierarchical state machine corresponding to the transmission session. One of the Senders acts as the initiator of the association. Reference to FIG.7, the sender (17) sends a T-ASSOCIATE Request message multicast (24) to each of the computer nodes participating in the association. The sender (17) also sends an initial sequence number for the data it will be sending. As shown in FIG.12, a timer is started after sending the T-ASSOCIATE Request. The timer retries the association process until the association is set up or the retry count is exhausted and an exception is signaled. When the association is set, the timer is cancelled. The receiving computer nodes or Receivers (18) respond with a T-ASSOCIATE Response (25). This can be configured to be unicast or multicast. The receivers of the association request also send an initial sequence number for the data they might be sending. The initiating Sender (17) then sends a T-ASSOCIATE Confirm to the receivers (18). If T-ASSOCIATE Response (25) is unicast, it is the onus of the Sender (17) to propagate the list of receivers to other receivers. It does so with the T-ASSOCIATE Confirm (26) message multicast to all the computer nodes participating in the session. This contains all the participant computer node information and their corresponding initial sequence number. One of the advantages of the receivers (18) knowing about each other is that if any repair needs to be done, it can be done from the nearest receiver. The receivers (18) respond with T_ASSOCIATE Confirm message (26) as an acknowledgement. If any receiving node (5) does not respond within a pre-configured time, the Sender (17) checks the corresponding aggregation policy and the synchronous list. Depending upon that, action might be retries by sending another T-ASSOCIATE Request message (24). After the configured number of retries are over, the sender (17) deals with all the receivers that have succeeded in setting up the association. If this does not include any one of the receivers in the synchronous set or if the aggregation policy so dictates, the sender (17) raises an exception and aborts the transmission. Once the association is established as described through the processes shown

in FIG.7, any of the sender (17) now initiates the data transmission to the receivers (18) that are a part of the association.

The association set up phase is also used to negotiate operational parameters of the association. The sender proposes the parameter values in the T-ASSOCIATE Request call. The receivers respond with the values they want to override in the T-ASSOCIATE Response. The so modified values are then propagated to the receivers using the T_ASSOCIATE Confirm. Amongst the parameters is a type of service parameter that is not negotiable. The type of service parameter can have two values, reliable-sequence-stream and reliable-sequence-packet. Both the settings provide reliable and sequenced delivery. The former accepts and sends the data as a stream to its client layer and the latter accepts and sends the data as finite sized packets.

Reference to FIG.8, the Sender (17) uses the T-DATA Request (27) multicast to transfer packets of data to the Receivers (18). It breaks the datagram data (16) to a list of packets with the identifying information, offset, length and the data segment. The size of the packet including the header is maximum transport layer Protocol Data Unit (PDU) size allowed on that interface. This is commonly referred to as the Maximum Transfer Unit (MTU). As shown in FIG.12, a timer is started after sending the T-DATA Request. The sender (17) then waits on the T-DATA-Response (T-DATA RSP) (28) from the receivers (18). A T-DATA-RSP (28) with the latest sequence number of the data (ACK) is sent by each of the receivers (18) receiving the correct and complete data. A sliding window is defined which is essentially either a count of packets or a count of octets that can be sent without waiting for a T-DATA Response (28). The window is moved forward on state machine corresponding to each receiver from which the T-DATA Response (28) acknowledgements are received. Only when the sender (17) has all the required number of T-DATA Response, does the sender (17) conclude that a particular packet has reached the intended receivers (18). The sequence number is always processed in the context of the Sender. In a scenario, where there are multiple Senders, there are that many sequence number progression at any receiver.

The T-DATA Response (28) can be configured to be sent either to the unicast address of the sender (17) or to the multicast address of this association of sender (17) and receivers (18) as response (29). In other scenarios where multicast is used for the data response, this would cause scalability problems. In this scenario, scalability of T-DATA Response (28 or 29) handling is not an issue as the number of computer nodes involved in a session is a small

finite number. If the T-DATA Response (28) is sent to the multicast address as response (29) and any receiver has missed the data packet, it quickly gets to know about the missed packet as soon as it sees the T-DATA Response (29) from other computer nodes. Otherwise if the response is sent unicast, a receiver will get to know about the missing packet only when it sees the gap after it receives the next transmission or the sender times out for the acknowledgement and solicits a data response.

Reference to FIG.9, if a receiver (18) finds one or more gap (32), on receiving a message i.e. on receiving a Block Packet T-DATA request or end-of –block multicast (31), the receiver (18) adds a negative acknowledgement (NACK) in the T-DATA Response (33) to the Sender (17) if the lower edge of the transmission window from the Sender (17) has not exceeded the beginning of the gap. If the lower edge has moved past that number, the receiver (18) turns to the other receivers for repair request. The receiver (18) multicasts a negative T-DATA-Response (33) to a subnet specific multicast address. Since all the computer nodes involved in the session know about each other, they start a timer to send a repair response. The nearest computer node sends the repair response packet unicast to the receiver wanting it and sends a repair report to the others on the multicast address (34). If the repair report is not seen within the timeout, all the participants restart the timer and the next nearest one sends the repair response and the repair report. This is done till all the retries are exhausted and failure conditions are invoked. The failure handling depends on whether the failing computer node belongs to the synchronous list or not as described above. The above mechanism is good for a scenario where the computer nodes know about their network topology to figure out the distance and cost of transmission. Alternatively, the computer node requesting missing data packets can send the request in an escalating manner. It first sends it to a subnet limited multicast address. If that does not solicit a response, it sends to the regular multicast address but with a Time-To-Live (TTL) network field set to a small value and then increasing the value to escalate the request to computer nodes many hops away. The highest level of escalation will be when it requests the Sender (17) to send the repair packets. The escalation level can be set in the options field of the header.

After all the data is successfully transmitted, the Sender sends a T-DATA Request with no data to indicate a temporary end of data. A T-DATA Request with no data is also sent to solicit acknowledgement for the previously sent data. The rest of the handling happens as before.

For flow control, the T-DATA Request packets go through a transmission window control. The other packets are not subject to that control. The window control constitutes an upper bound of octets that can be transmitted without waiting for an acknowledgement via T-DATA Response packet. The T-DATA Response packet can contain information about missing data segments. It can also contain a receive window size that can further restrict the overall window size as determined by load on the receiver.

In general, to handle lost or missed packets, the packets are retransmitted (31) whenever the gap is reported. The retransmission scenarios are attached to timers that are invoked when a packet is transmitted. The receivers can identify the retransmitted data from the sequence numbers that it has already received. It then drops the data if it already has it. Some of the deserving cases are explicitly described in the above sections.

Reference to FIG.10, if a receiver is to be shutdown or detects a failure from which it cannot recover, the I/O Module (10) informs (35) the Control Module (9) about that and the Control Module (9) sends a T-DISSOCIATE Request (36) to the Sender's unicast address or multicast address as configured. The Sender responds with a T-DISSOCIATE Confirm (37) to the Receiver's unicast address or multicast address as configured.

Whenever any of the senders or the receivers wants to terminate the association, it sends a T-DISSOCIATE Request (38) to the multicast address and expects the T-DISSOCIATE Confirm (37) from the others to unicast to it. This is also tied to a timer. If the confirm message is not received, it will retry the configured number of times and then close down the session at its end.

Congestion control is implemented by a combination of three mechanisms. The first is the rate control that specifies the maximum rate of data that can be sent from the sender, on a per association basis and a total of all associations. The second is the transmit window that allows the maximum amount of data that the sender can transmit before waiting for an acknowledgement. The third is the receive window sent from each receiver in T-DATA-Response packet as the acknowledgement for data. It is a subset of the configured transmit window. It is dependent upon the system resources at the Receiver. The Round Trip Time (RTT) used by the Sender is a function of the RTTs of the computer nodes in the synchronous part of the pipeline with some overage for the remote computer nodes that are farther away. The type of function can be, but not limited to, maximum, average, mode etc.

The flow control is implemented using a combination of congestion control and configured parameters like the transmission window and receive window. These configured parameters are administrative domain specific and is usually but not limited to interface bandwidth, link bandwidth, time of day, network traffic etc.

As is evident from the discussions above, for the replication mechanism–of the present invention, the data transfer from sender (17) is required to be reliable only to an aggregate of receivers (18). In other words, the client (1) need not be responsible for transmitting data to all the computer nodes (5) but to an aggregate of computer nodes only. The aggregate of computer nodes can be defined by a policy. The policies can be any one of the following but not limited to:

1) All or None: The client considers the transmission to be successful, only if all the computer nodes in the replication pipeline have received the data.

2) First N: The client considers the transmission to be successful if the first N count of computer nodes in the replication pipeline has received the data.

3) Any N: The client considers the transmission to be successful if any N computer nodes out of all the computer nodes in the replication pipeline have received the data.

4) Synchronous N: The client considers the transmission to be successful if the synchronous N computer nodes out of all the computer nodes in the replication pipeline have received the data.

The aggregate of computer nodes defined by any of such policies collaborate among themselves to ensure that the data is transmitted reliably from the sender (17) to each of the receivers (18) participating in the association. In the present example the protocol is implemented as a state machine for the receiver and a hierarchical state machine for the transmitter or sender. In the case of the transmitter, the hierarchy is of two levels. The lower level corresponds to each receiver. The upper level corresponds to the aggregate of all receivers.

FIG.11 shows the various elements of the header (900), not necessarily in the same order, in the messages of this protocol of the present invention. Some of the elements may be used or not used and with varying sizes. The information of the usage and their sizes are specified in

a profile configuration for a specific implementation. The Source TSAP (901) and Destination TSAP (902) are used to multiplex between different users of this protocol. The Header Length (903) refers to the length of the header (900). In the preferred embodiment, the header (900) would of fixed length and would be configured in the profile as mentioned above. In that case, Header Length (903) field will not be present. Other embodiments that have a variable sized header (900) will have the Header Length (903) field. The Checksum (904) would ideally be at least 32 bit length but smaller size can be used. The algorithm for the checksum should be specified in the profile. The Version (905) is the version of the protocol. The PDU Type (906) field specifies the type of packet like Association request or Association response or Data request etc. The Session ID (907) is the one generated for this session. It has to be unique in the network at any point of time. The Flags (908) field carries the various binary valued fields representing the various settings in the protocol. The Options (909) carries the various non-binary valued fields. The Sequence Number (910) field has a context dependent meaning. In the association set up packets, it refers to the initial sequence number. In the data request packets, it refers to the starting sequence number of the data octets in that packet. In the data response packet, it refers to the cumulative sequence number. In the data response solicitation packets, it refers to the largest sequence number of the data octets already transmitted. In the dissociate packets, it refers to the largest sequence number of the data octets already transmitted. In the data repair packets, it refers to the starting sequence of the missing data. The sequence number field wraps to zero and progresses forward every time it has reached the maximum field value. Each iteration of the wrapped sequence number has a position value higher than the sequence number prior to that wrap. The length field (911) specifies the total length of the data portion.

Fig.12 shows the design of the state machine through a fragment. It demonstrates the incorporation of the aggregation policy. The state machine at State 1 (1001) sends a Request (1002) and starts a timer waiting for Response (1004). The state machine is now in a Wait (1003) state. The timer can also have a retry counter that is reset. If the Response (1004) is received, the machine goes into Evaluate Policy (1006). Depending upon the result of the evaluation, it goes into State 2 (1007) or goes back to Wait (1003) state waiting for further Response(s) (1004). Depending upon the case, it might trigger another Request (1002) and then go into the Wait (1003) state. If the timer expires, it may trigger another request. If the retry count is exhausted, it might have an Exception (1008) situation and go into a State 3 (1009) after evaluating the policy.

Process or method charts is used to describe the steps of the present invention. While the various steps in this process chart are presented and described sequentially, some or all of the steps may be executed in different orders, may be combined or omitted, and some or all of the steps may be executed in parallel. Further, in one or more of the embodiments of the invention, one or more of the steps described above may be omitted, repeated, and/or performed in a different order. In addition, additional steps, omitted in the process chart may be included in performing this method. Accordingly, the specific arrangement of steps shown in FIG.5 through FIG.8 should not be construed as limiting the scope of the invention.

What is claimed is:

1. A method of providing a reliable and efficient multicast data transfer mechanism in a communication network, said mechanism including a plurality of software components running on a plurality of computer nodes and one or more data system managers running on computers, wherein each of said plurality of computer nodes and each of said one or more data system managers are connected through said communication network and is operable to respond to at least one type of instruction, said method comprising the steps of:

setting up of an association by one or more computer nodes of said plurality of computer nodes as senders with one or more computer nodes of said plurality of computer nodes as receivers using a combination of unicast and multicast protocols; and

transmitting one or more packets of data through said multicast protocol by said one or more senders to said one or more receivers.

2. The method of claim 1, wherein said setting up of an association step comprises the steps of:

finding out computer node information and assigning multicast network address information corresponding to said association for said one or more receivers by said data system manager on receipt of initiation request from said one or more senders;

transmitting unicast said computer node information and said corresponding multicast network address information for said one or more receivers by said data system manager to said one or more senders and to said one or more receivers;

transmitting multicast request for association by said one or more senders to said one or more receivers;

receiving of response sent from said one or more receivers by said one or more senders against said request for association; and

confirming association between said one or more senders and said one or more receivers by sending multicast association confirmation.

3. The method claim 2, wherein said assigning of multicast network address information is done by an address allocator module included in said one or more data system managers.

4. The method of claim 3, wherein said address allocator module further generates one or more encryption keys and distributes said one or more encryption keys with said multicast

network address information to make said transmission of said one or more packets of data encrypted.

5. The method of claim 2, wherein some of said one or more receivers are defined as synchronous and rest as asynchronous.

6. The method of claim 1, wherein said transmitting one or more packets of data step comprises the steps of:

multicasting one or more data requests by said one or more senders to said one or more receivers; and

responding to said one or more data requests by said one or more receivers.

7. The method of claim 6, wherein said responses from those of said one or more receivers against receipt of said one or more data requests are positive acknowledgements.

8. The method of claim 6, wherein said responses from those of said one or more receivers against a non-receipt of said one or more data requests are negative acknowledgements.

9. The method of claim 6, wherein said responses from said one or more receivers are sent to unicast address of said one or more senders.

10. The method of claim 6, wherein said responses from said one or more receivers are sent to multicast address of said one or more senders and said one or more receivers.

11. A method of providing an efficient error recovery mechanism in a communication network, said error recovery mechanism including one or more senders, and one or more receivers, wherein each of said one or more senders and each of said one or more receivers are in an association relationship connected through said communication network and is operable to respond to at least one type of instruction, said method comprising the steps of:

detecting a data gap by any of said one or more receivers on receiving a packet data request or an end of transmission message from said one or more senders;

transmitting a negative acknowledgement by those of said one or more receivers which detect said gap; and

transmitting repair packet data to those receivers against said negative acknowledgement said error recovery.

12. The method of claim 11, wherein said negative acknowledgement is unicast to said one or more senders if lower edge of transmission window from said one or more senders has not exceeded beginning of said gap.

13. The method of claim 11, wherein said negative acknowledgement is multicast to said one or more receivers if lower edge of transmission window from said one or more senders has exceeded beginning of said gap.

14. A reliable data transport system in a communication network, said system including a plurality of software components running on a plurality of computer nodes and one or more data system managers, wherein each said computer node and said one or more data system managers are operable to respond to at least;

working out multicast network address information for one or more receivers from said plurality of computer nodes by a data system manager on receipt of an initiation request from one or more computer nodes as senders;

transmitting unicast said multicast network address information to said one or more senders and to said one or more receivers;

transmitting T-Associate request multicast by said one or more senders to said one or more receivers;

responding to said T-Associate request multicast by said one or more receivers by sending T-Associate Response;

transmitting T-Associate Confirm multicast by said one or more senders to said one or more receivers and transmitting back said T-Associate Confirm multicast by said one or more receivers;

sending T-Data Request multicast by said one or more senders to said one or more receivers;

returning T-Data Response by said one or more receivers;

sending negative acknowledgement T-DATA Response for repair request by any of said one or more receivers on detection of data gap for missing data;

transmitting unicast said missing data by said one or more senders and by neighbors of said one or more receivers and transmitting multicast repair report;

transmitting T-Dissociate Request by said one or more receivers on completion transmission and on failure to receive packets; and

transmitting T-Dissociate Confirm by said one or more senders on receipt of T-Dissociate Request.

15. The system of claim 14, wherein said plurality of computer nodes create a state machine corresponding to a transmission session.

16. The system of claim 15, wherein said state machine is a hierarchical state machine corresponding to a transmission session created by said one or more senders.

17. The system of claim 16, wherein lower level of said hierarchical state machine is created corresponding to each of said one or more receivers.

18. The system of claim 16, wherein upper level of said hierarchical state machine is created corresponding to an aggregate of said one or more receivers.

19. The system of claim 15, wherein said state machine is created by said one or more receivers.

20. The system of claim 15, wherein a timer is set on initiation of each of said request corresponding to said state machine as per policy.

21. The system of claim 20, wherein said timer is cancelled if a configured number of said responses are received as per said policy.

22. The system of claim 20, wherein said timer triggers repeat of said request until a number of count configured in said timer gets exhausted if a configured number of said responses are not received.

23. The system of claim 14, wherein said returning of T-Data Response is unicast to said one or more senders.

24. The system of claim 14, wherein said returning of T-Data Response is multicast so that said one or more receivers know and collaborate with each other to repair said missing data by responding to said repair request transmitted as multicast.

25. The system of claim 14, wherein said T-Data Response is multicast to said multicast network address information.

26. The system of claim 14, wherein said T-Data Response is unicast to said one or more senders.

27. The system of claim 14, wherein said T-Dissociate Request and said T-Dissociate Confirm are unicast.

28. The system of claim 14, wherein said T-Dissociate Request and said T-Dissociate Confirm are multicast.

29. A method of providing a mechanism for reliable and efficient transmission of data to an aggregate of receivers in a communication network, said communication network connecting one or more computer nodes as senders, one or more computer nodes as said aggregate of receivers and one or more data system managers, said method comprising the steps of:

setting up of an association between said one or more senders and said aggregate of receivers; and

multicasting one or more packets of said data by said one or more senders to said aggregate of receivers;

wherein said aggregate of receivers collaborate among themselves to ensure successful transmission of said data from said one or more senders to said aggregate of receivers.

30. The method of claim 29, wherein said association between said one or more senders and said aggregate of receivers is done through a combination of unicast and multicast messages.

31. The method of claim 29, wherein said transmission of said data is considered successful by said one or more senders only when said aggregate of receivers has received said data.

32. The method of claim 29, wherein a policy defines said aggregate of receivers from said one or more receivers participating in a transmission session.

33. The method of claim 32, wherein as per said policy said aggregate of receivers are all the receivers participating in said transmission session.

34. The method of claim 32, wherein as per said policy said aggregate of receivers are any specified number of the receivers of said transmission session.

35. The method of claim 32, wherein as per said policy said aggregate of receivers are synchronous receivers of said transmission session.

(1/12)

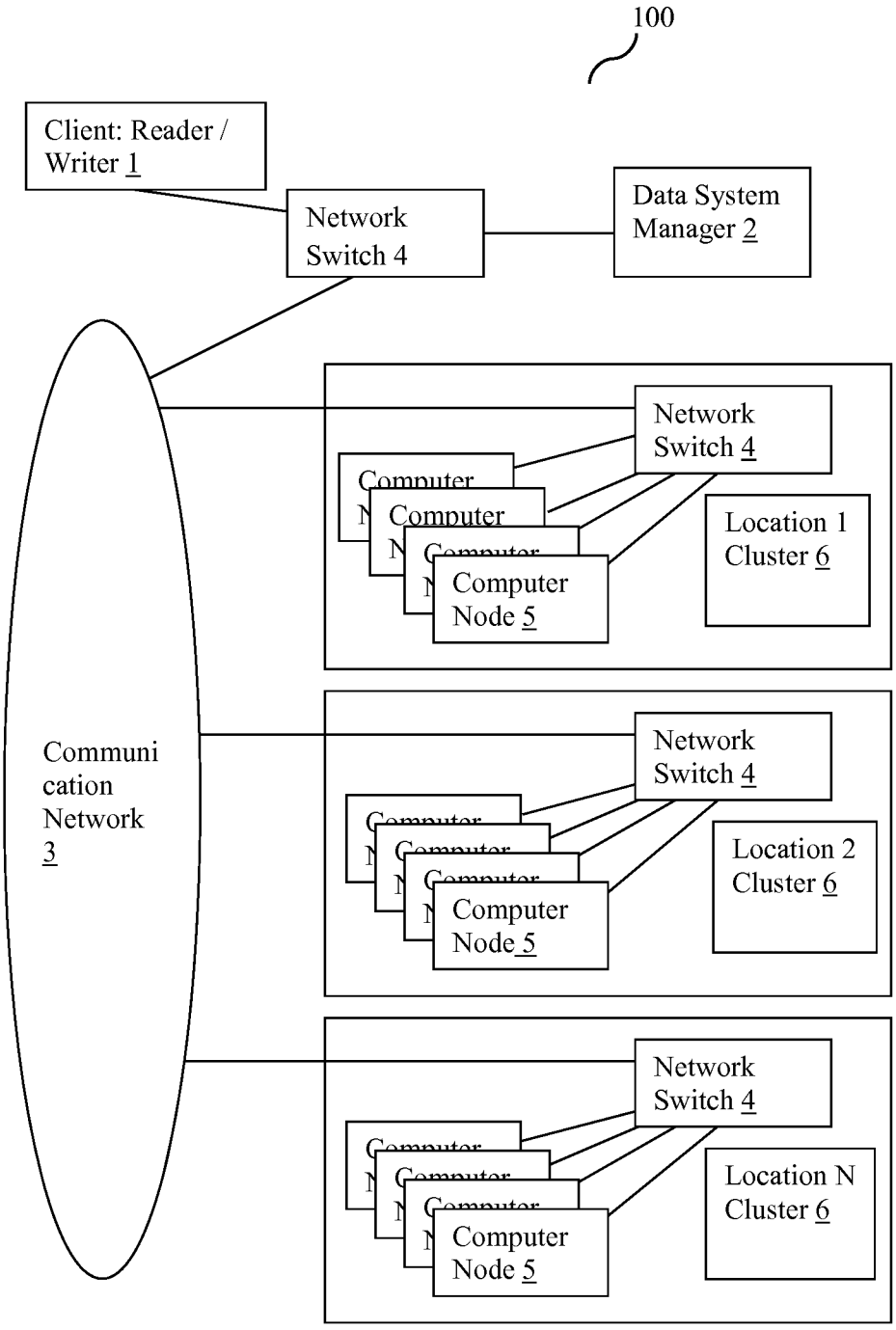
100
Client: Reader / Writer 1
Network Switch 4
Data System Manager 2
Communi cation Network 3
Network Switch 4
Computer Node 5
Location 1 Cluster 6
Network Switch 4
Computer Node 5
Location 2 Cluster 6
Network Switch 4
Computer Node 5
Location N Cluster 6

FIG. 1

(2/12)

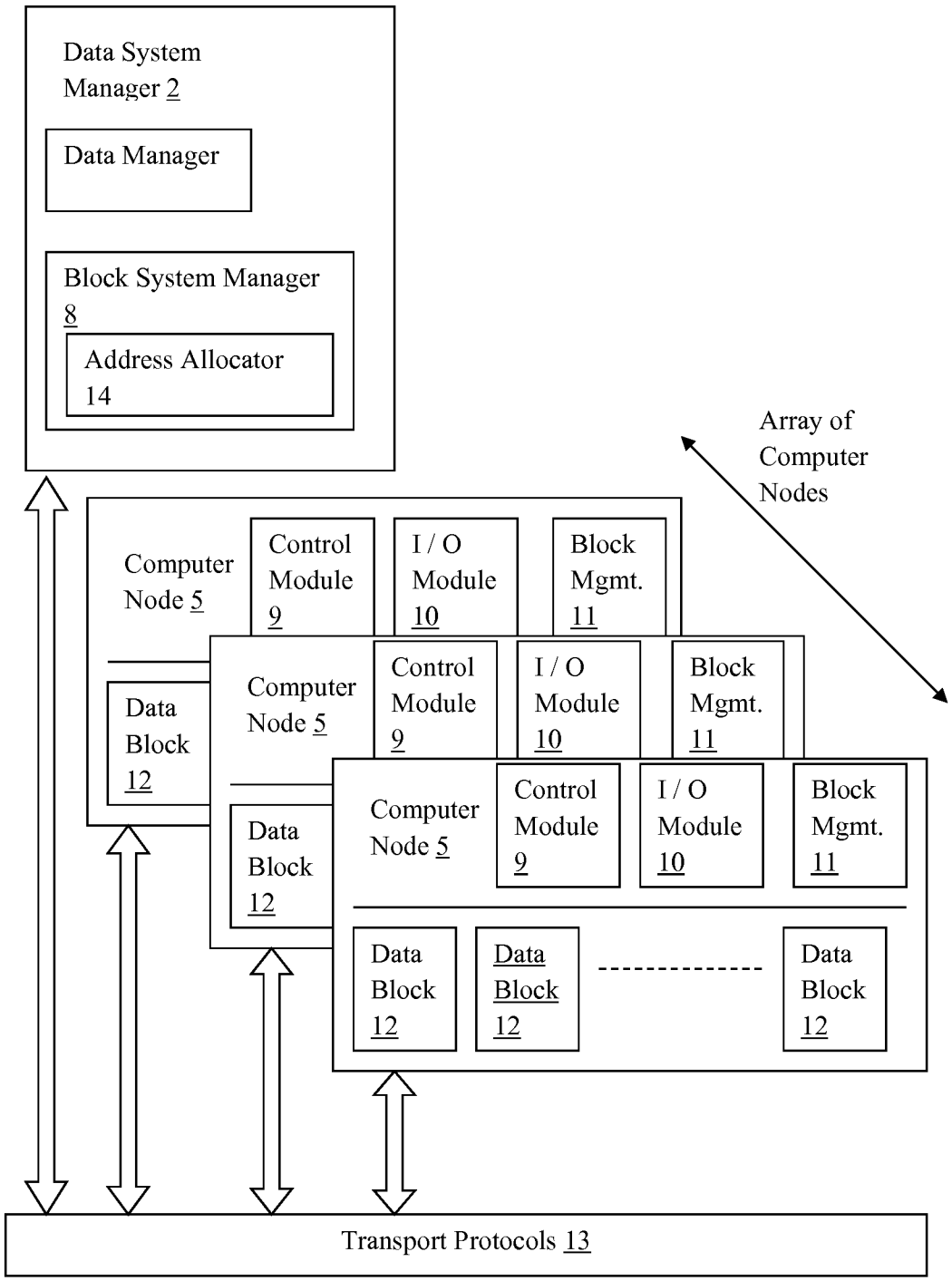
Data System Manager 2
Data Manager
Block System Manager 8
Address Allocator 14
Array of Computer Nodes
Computer Node 5
Control Module 9
I / O Module 10
Block Mgmt. 11
Data Block 12
Transport Protocols 13

FIG.2

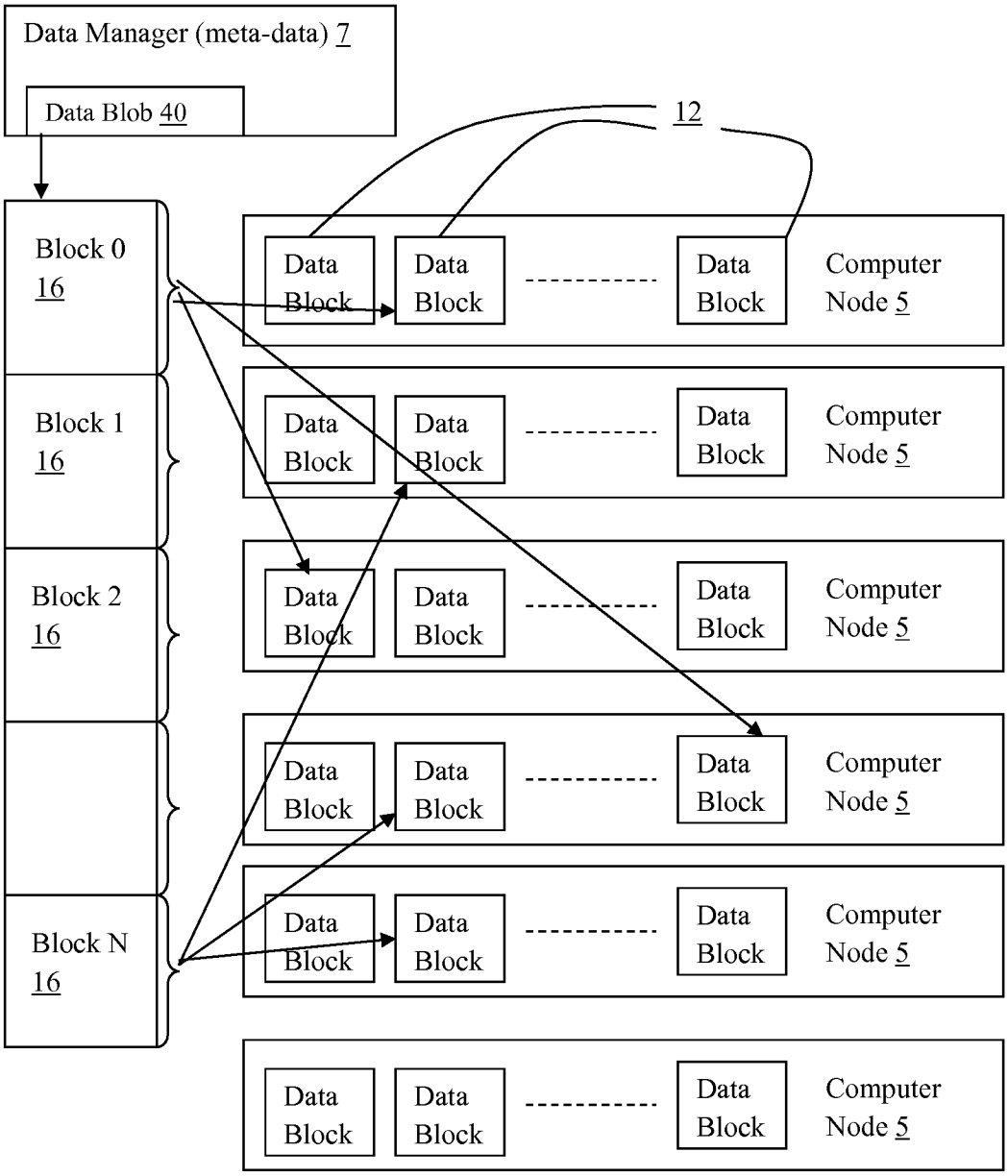
Data Manager (meta-data) 7
Data Blob 40
12
Block 0
16
Block 1
16
Block 2
16
Block N
16
Data Block
Computer Node 5

FIG.3

(4/12)

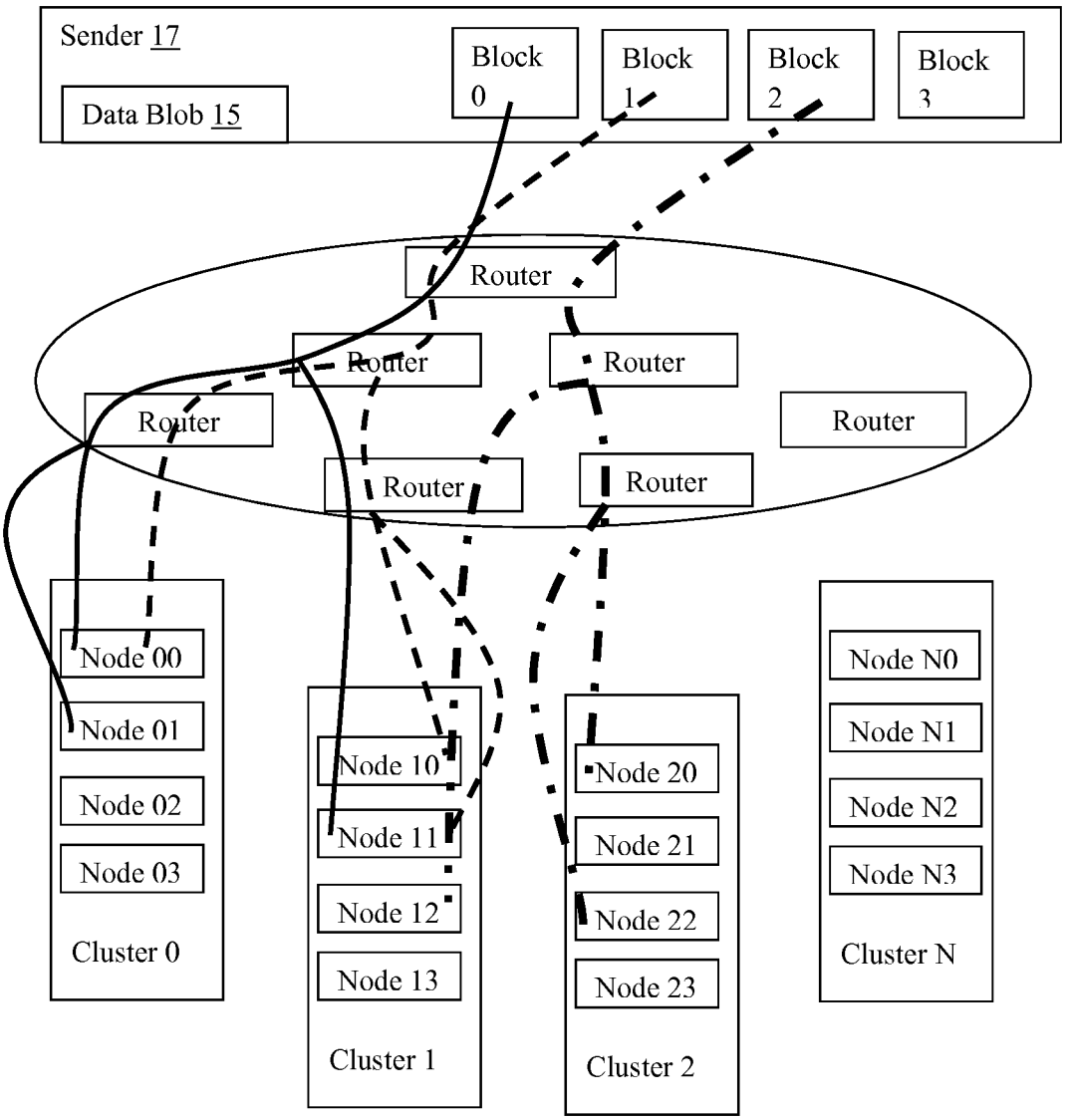
Sender 17
Data Blob 15
Block 0
Block 1
Block 2
Block 3
Router
Router
Router
Router
Router
Router
Router
Node 00
Node 01
Node 02
Node 03
Cluster 0
Node 10
Node 11
Node 12
Node 13
Cluster 1
Node 20
Node 21
Node 22
Node 23
Cluster 2
Node N0
Node N1
Node N2
Node N3
Cluster N

FIG.4

(5/12)

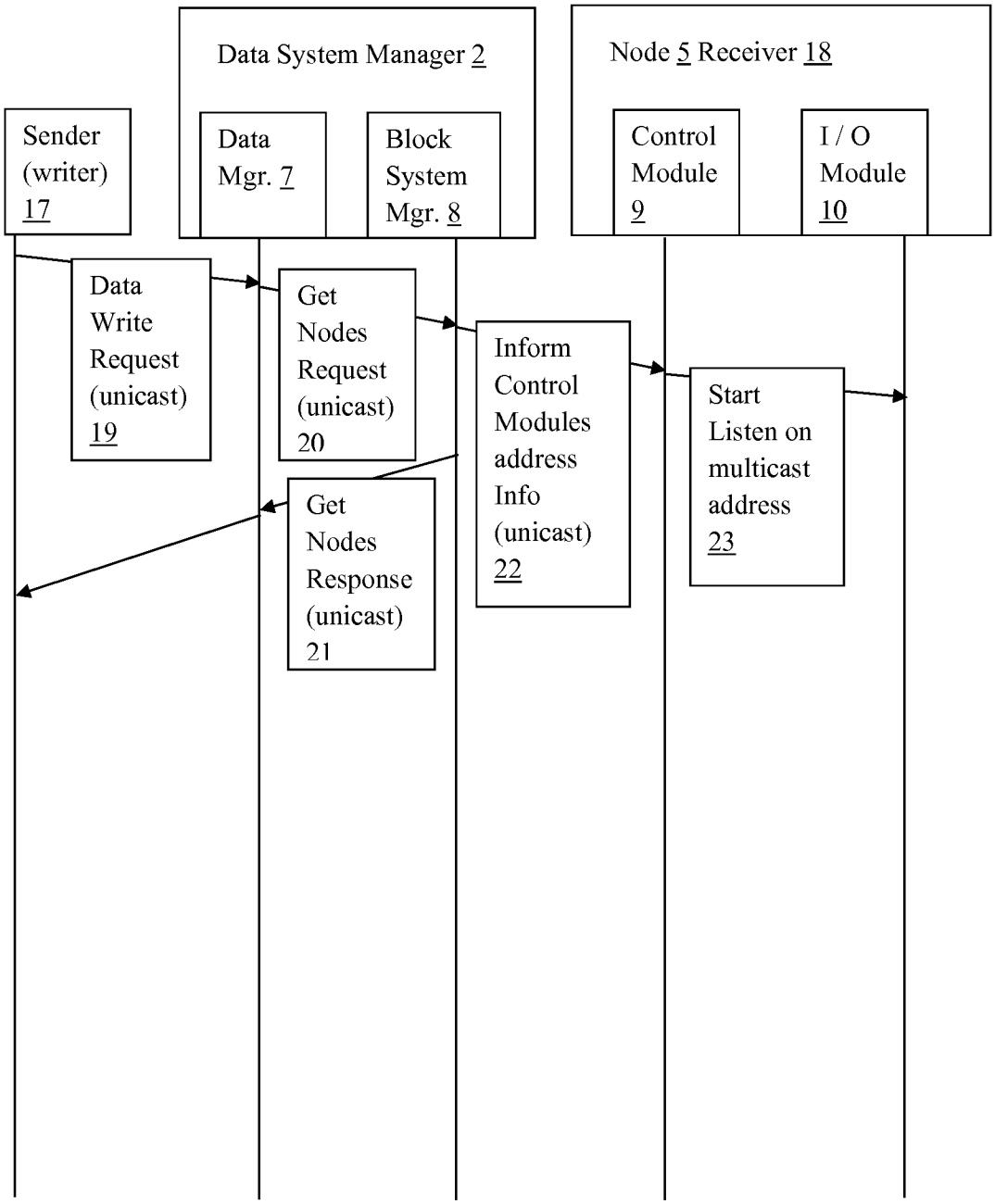
Data System Manager 2
Node 5 Receiver 18
Sender (writer) 17
Data Mgr. 7
Block System Mgr. 8
Control Module 9
I / O Module 10
Data Write Request (unicast) 19
Get Nodes Request (unicast) 20
Inform Control Modules address Info (unicast) 22
Start Listen on multicast address 23
Get Nodes Response (unicast) 21

FIG.5

(6/12)

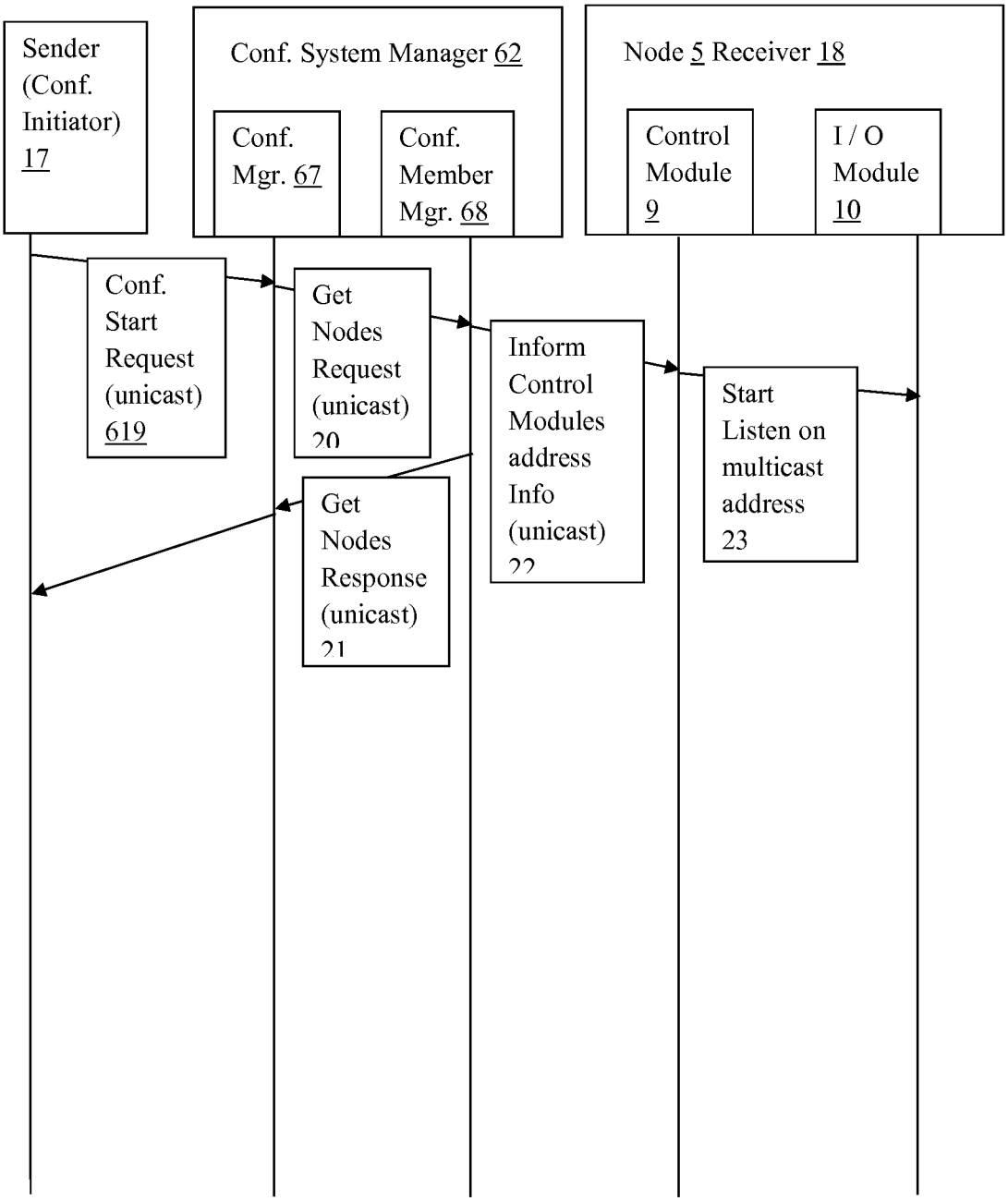
Sender (Conf. Initiator) 17
Conf. System Manager 62
Conf. Mgr. 67
Conf. Member Mgr. 68
Node 5 Receiver 18
Control Module 9
I / O Module 10
Conf. Start Request (unicast) 619
Get Nodes Request (unicast) 20
Inform Control Modules address Info (unicast) 22
Start Listen on multicast address 23
Get Nodes Response (unicast) 21

FIG.6

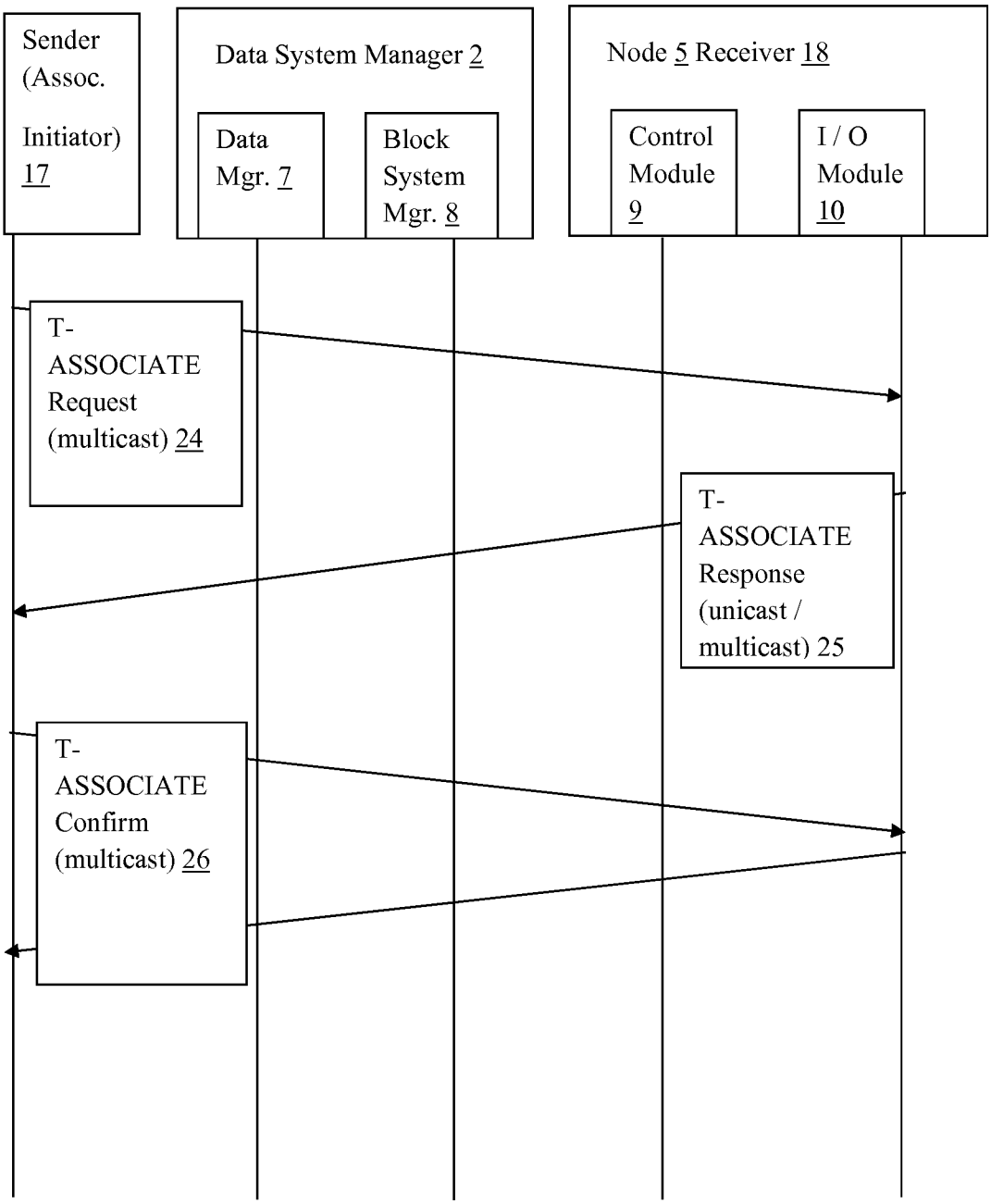
Sender (Assoc. Initiator) 17
Data System Manager 2
Data Mgr. 7
Block System Mgr. 8
Node 5 Receiver 18
Control Module 9
I / O Module 10
T-ASSOCIATE Request (multicast) 24
T-ASSOCIATE Response (unicast / multicast) 25
T-ASSOCIATE Confirm (multicast) 26

FIG.7

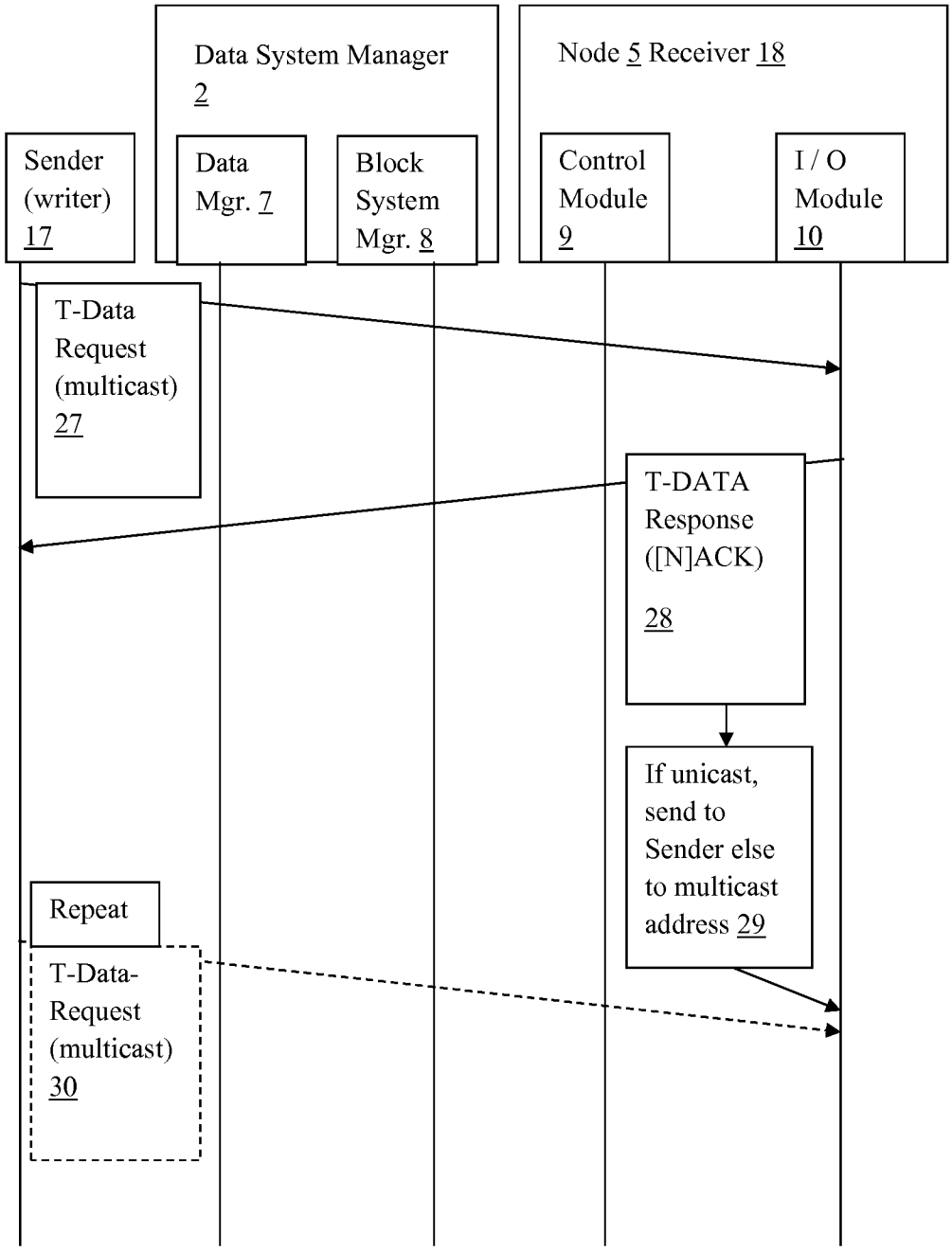
Data System Manager 2
Node 5 Receiver 18
Sender (writer) 17
Data Mgr. 7
Block System Mgr. 8
Control Module 9
I / O Module 10
T-Data Request (multicast) 27
T-DATA Response ([N]ACK) 28
If unicast, send to Sender else to multicast address 29
Repeat
T-Data-Request (multicast) 30

FIG.8

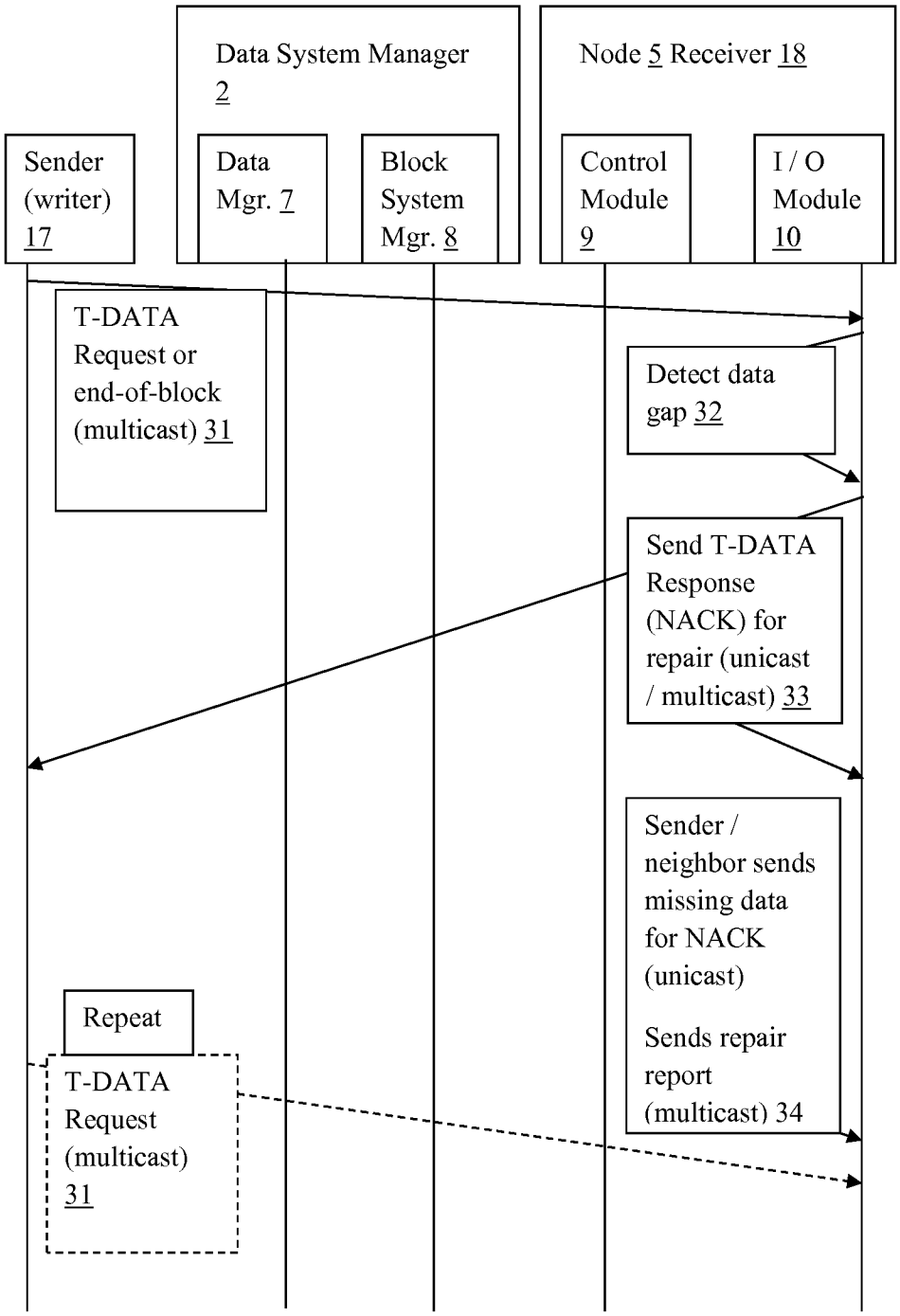
Data System Manager 2
Node 5 Receiver 18
Sender (writer) 17
Data Mgr. 7
Block System Mgr. 8
Control Module 9
I / O Module 10
T-DATA Request or end-of-block (multicast) 31
Detect data gap 32
Send T-DATA Response (NACK) for repair (unicast / multicast) 33
Sender / neighbor sends missing data for NACK (unicast)
Sends repair report (multicast) 34
Repeat
T-DATA Request (multicast) 31

FIG.9

(10/12)

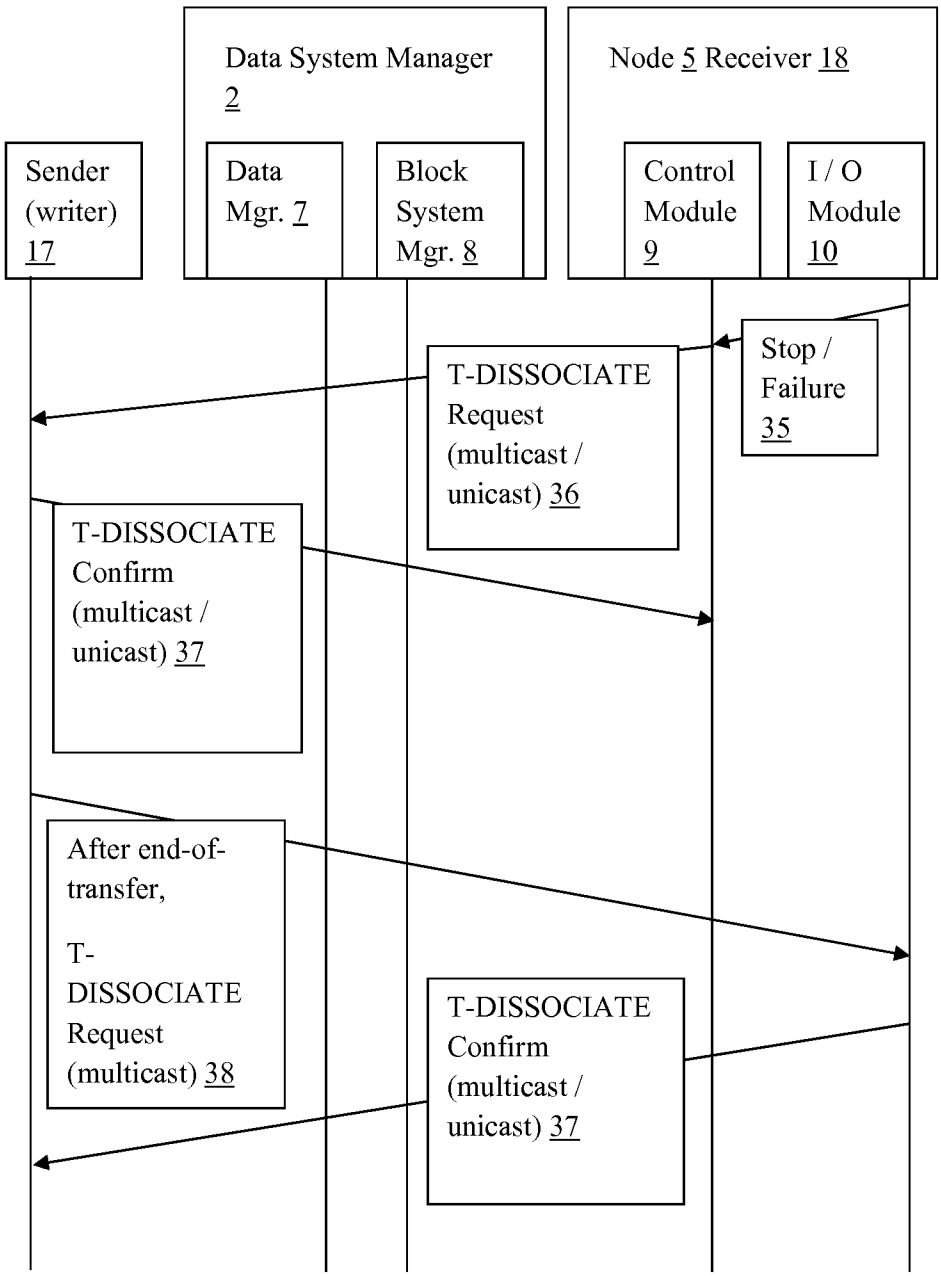
Data System Manager 2
Node 5 Receiver 18
Sender (writer) 17
Data Mgr. 7
Block System Mgr. 8
Control Module 9
I / O Module 10
Stop / Failure 35
T-DISSOCIATE Request (multicast / unicast) 36
T-DISSOCIATE Confirm (multicast / unicast) 37
After end-of-transfer,
T-DISSOCIATE Request (multicast) 38
T-DISSOCIATE Confirm (multicast / unicast) 37

FIG.10

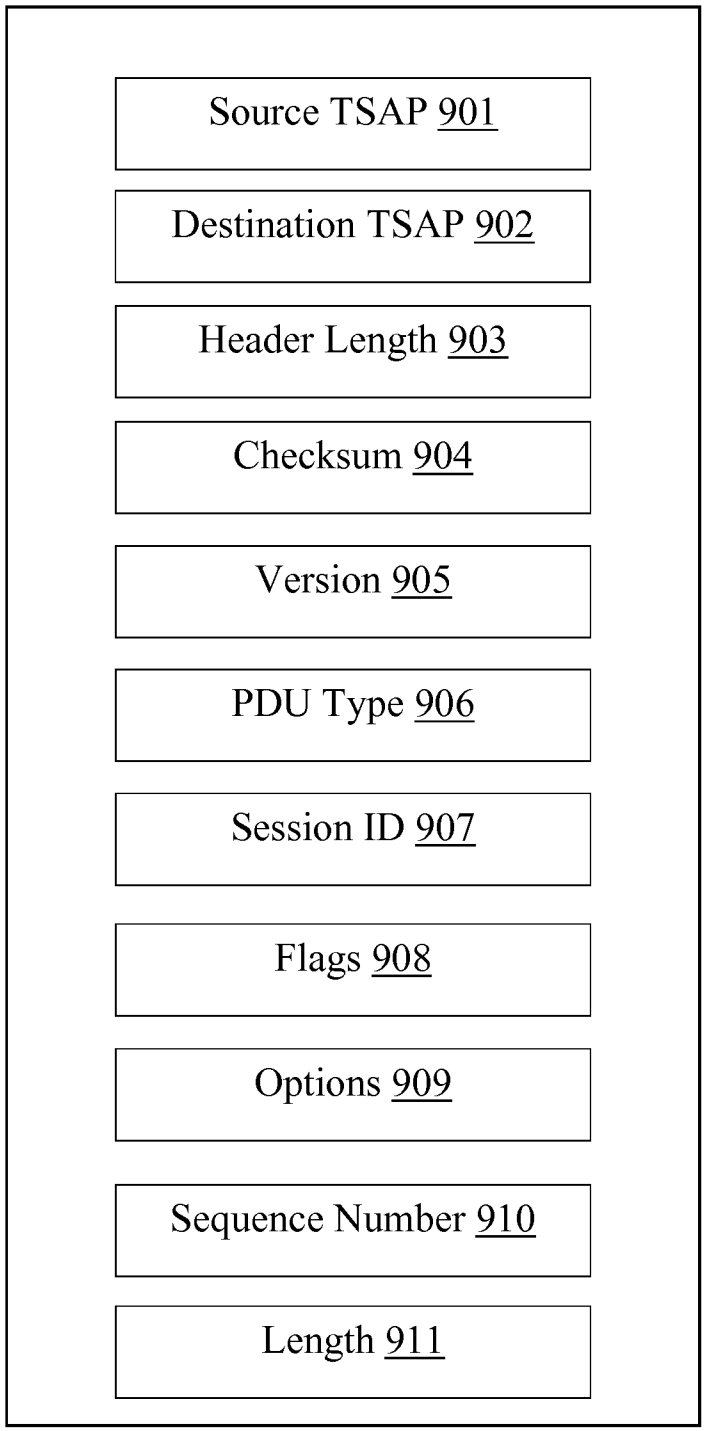
Source TSAP 901
Destination TSAP 902
Header Length 903
Checksum 904
Version 905
PDU Type 906
Session ID 907
Flags 908
Options 909
Sequence Number 910
Length 911

FIG.11

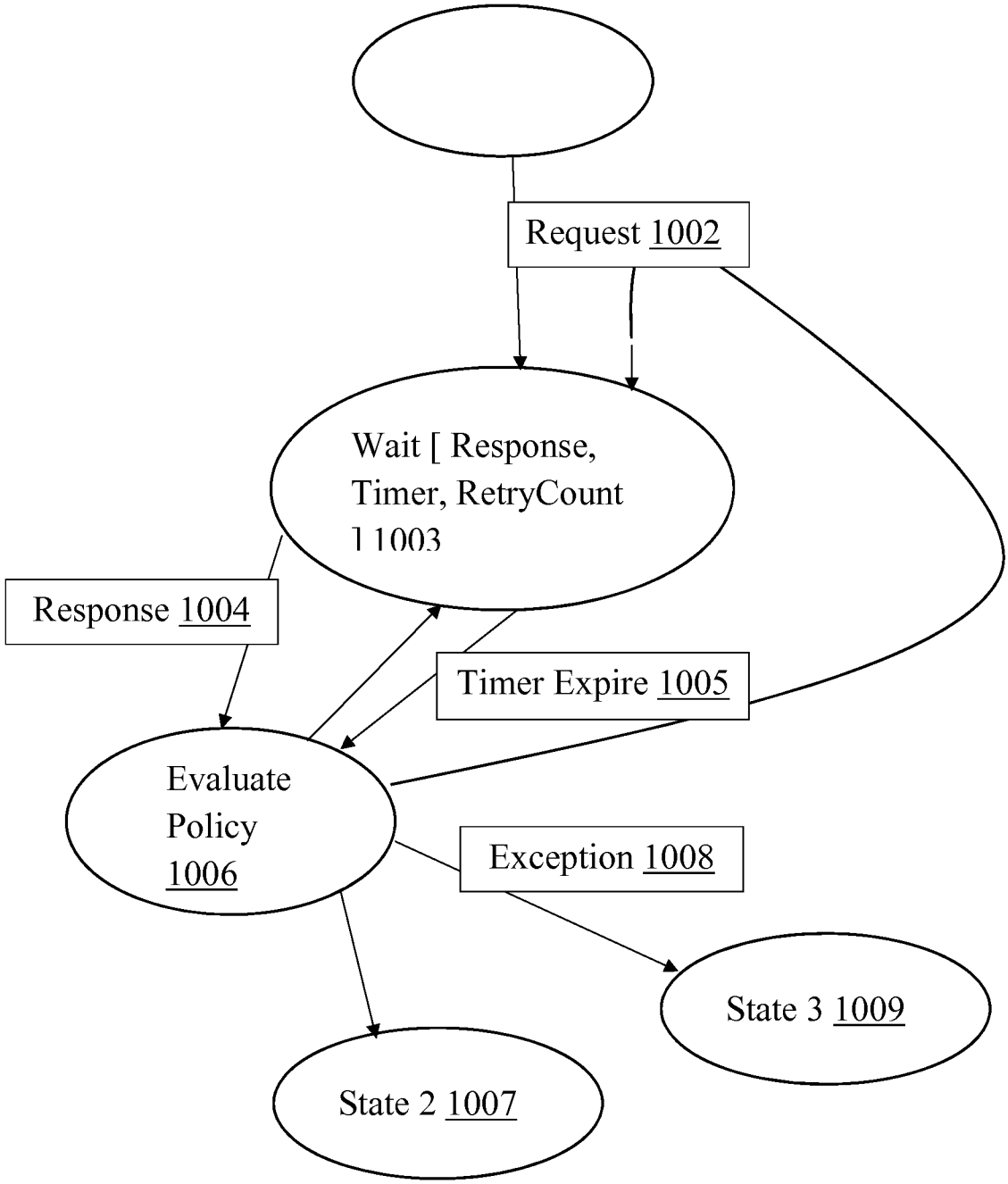
Request 1002
Wait [Response, Timer, RetryCount] 1003
Response 1004
Timer Expire 1005
Evaluate Policy 1006
Exception 1008
State 3 1009
State 2 1007

FIG.12

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/014219 11-05-2015

PCT/US2015/014219

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04L 29/06 (2015.01)
CPC - H04L 63/065 (2015.01)
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8) - H04L 12/28, 29/06 (2015.01)
USPC - 370/390, 392, 395.2, 395.21, 395.3, 398; 713/163

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
CPC - H04L 12/5693, 45/16, 63/065; H04Q 11/0478 2015.01) (keyword delimited)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

PatBase, Orbit, Google Patents, Google Scholar.
Search terms used: multicast, unicast, association, node, T-associate

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 1,139,602 A1 (CHEN et al.) 04 October 2001 (04.10.2001) entire document	11
--		---
Y		8, 12-28
Y	US 2010/0085970 A1 (BARKAN et al.) 08 April 2010 (08.04.2010) entire document	1-10, 14-35
Y	US 2007/0258466 A1 (KAKANI) 08 November 2007 (08.11.2007) entire document	2-10, 14-35
Y	US 6,317,415 B1 (DARNELL et al.) 13 November 2001 (13.11.2001) entire document	5, 12-13, 35
Y	US 2005/0138369 A1 (LEBOVITZ et al.) 23 June 2005 (23.06.2005) entire document	1-10, 12, 30
Y	US 2008/0123577 A1 (JAAKKOLA et al.) 29 May 2008 (29.05.2008) entire document	14-28, 34
Y	US 6,810,259 B1 (ZHANG) 26 October 2004 (26.10.2004) entire document	15-22

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
08 April 2015	11 MAY 2015

Name and mailing address of the ISA/US	Authorized officer:
Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774