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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 March 2013 (07.03.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/030207 A1

(51) **International Patent Classification**:
G06F 17/30 (2006.01)

(21) **International Application Number**:
PCT/EP2012/066725

(22) **International Filing Date**:
29 August 2012 (29.08.2012)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
13/224,404 2 September 2011 (02.09.2011) US

(71) **Applicant** *(for all designated States except US)*: **COMPUVERDE AB** [SE/SE]; Ö Vittusgatan 36, S-371 33 Karlskrona (SE).

(72) **Inventors; and**
(71) **Applicants** : **BERNBO, Stefan** [SE/SE]; Arklimästaregatan 46 A, S-371 36 Karlskrona (SE). **MELANDER, Christian** [SE/SE]; Selenvägen 3, S-370 30 Rödeby (SE). **PERSSON, Roger** [SE/SE]; Nyhemsvägen 1 C, S-371 43 Karlskrona (SE). **PETERSSON, Gustav** [SE/SE]; Kullavägen 30, S-370 43 Sturkö (SE).

(74) **Agent**: **HJALMARSSON, Magnus**; Awapatent AB, P.O. Box 99, S-351 04 Växjö (SE).

(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, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ,

[Continued on next page]

(54) **Title**: A METHOD AND DEVICE FOR WRITING DATA TO A DATA STORAGE SYSTEM COMPRISING A PLURALITY OF DATA STORAGE NODES

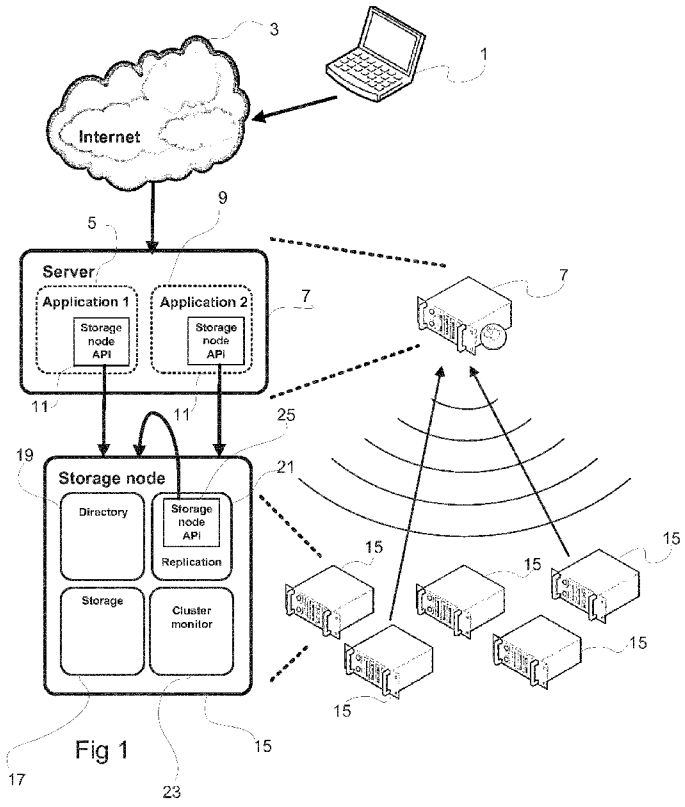

Fig 1

(57) **Abstract**: There is disclosed a method for writing data in a data storage system comprising a plurality of data storage nodes, the method being employed in a server running an application which accesses data in the data storage system, and comprising: sending a multicast storage query to a plurality of said storage nodes; receiving a plurality of responses from a subset of said storage nodes, said responses including a storage node property; selecting at least two storage nodes in the subset for storing said data, based on said responses, wherein the selecting is based on a data property of the data to be stored and a storage node property.

WO 2013/030207 A1

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, ML, MR, NE, SN, TD, TG).

Published:

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

A METHOD AND DEVICE FOR WRITING DATA TO A DATA STORAGE SYSTEM COMPRISING A PLURALITY OF DATA STORAGE NODES

TECHNICAL FIELD

[0001] The present disclosure relates a method and a device for writing data in a data storage system comprising a plurality of data storage nodes, the method being employed in a server in the data storage system.

BACKGROUND

[0002] Such a method is disclosed in, for example, US Patent Publication No. 2005/0246393 A1. This method is disclosed for a system that uses a plurality of storage centres at geographically disparate locations. Distributed object storage managers are included to maintain information regarding stored data.

[0003] One problem associated with such a system is how to accomplish simple and yet robust and reliable writing of data.

SUMMARY OF THE INVENTION

[0004] There is therefore disclosed a method for writing data in a data storage system including a plurality of data storage nodes. The method may be employed in a server running an application which accesses data in the data storage system. The method may include sending a multicast storage query to a plurality of said storage nodes. Responses may be received from a subset of said storage nodes, and the responses may include a storage node property. The server may select at least two storage nodes in the subset for storing said data, and the selection may be

based on said responses. The selection may be based on a data property of the data to be stored and/or the received storage node property.

[0005] A method for writing data to a data storage system may be accomplished via a server running an application which accesses data in the data storage system. The method may include sending a multicast storage query to a plurality of storage nodes, and receiving a plurality of responses from a subset of said storage nodes. The responses may include storage node information relating to each of the responding storage nodes. The server may select at least two storage nodes in the subset, for example based on said responses. The selection may include determining, based on an algorithm, for each storage node in the subset, a probability factor which may be based on its storage node information. The server may randomly select said at least two storage nodes. The probability of a storage node being selected may depend on its probability factor. The method may further involve sending data and a data identifier, corresponding to the data, to the selected storage nodes.

[0006] In an example embodiment, data may be stored in a content sensitive way. For example, a storage node may be selected that has basic characteristics or properties that match or are compatible with the attributes implied by the data.

[0007] Example node properties may include the type of disk, response time, redundancy configuration, reliability, node environment, territory, and/or energy consumption.

[0008] The data property may imply or be associated with a number of allowed storage node properties. For example, a certain reported data property may be used by the server and/or a storage node to select another storage node based on a corresponding storage node property. As an example, if the data property is that the data may need to be accessed quickly, the corresponding storage node property may be response time.

[0009] The data property may be defined by the type of user that is linked to the data. A premium user may require a more reliable storage node.

[0010] The data property may also defined by the type of data. For instance data that need be accessed quickly may imply the need of a quick storage node.

[0011] The disclosure further relates to a method for writing data in a data storage system including a plurality of data storage nodes. The method may be employed in a server running an application which accesses data in the data storage system. The method may include selecting a set of allowed storage node properties based on a data property of the data to be stored. The allowed storage node properties may be properties that may be possessed by a storage node and properties which the server desires be present in the storage node which will store the particular data. The server may send a multicast storage query to a plurality of said storage nodes, and the

storage query may include the allowed storage node properties. The server may receive a plurality of responses from a subset of said storage nodes which comply with requirements implied by the allowed storage node properties. The server may select at least two storage nodes in the subset for storing said data, for example based on said responses.

[0012] The disclosure further relates to a device for writing data in a data storage system. The data storage system may include a plurality of data storage nodes. The device for writing data may be included in a server running an application which accesses data in the data storage system. The device may send a multicast storage query to a plurality of said storage nodes. The device may receive a plurality of responses from a subset of said storage nodes. The responses may include a storage node property. The device may select at least two storage nodes in the subset for storing said data, for example based on said responses. The selection may be based on a data property of the data to be stored and/or a storage node property.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] Fig 1 illustrates a distributed data storage system.

[0014] Figs 2A –2C, and fig 3 illustrate a data reading process.

[0015] Figs 4A –4C, and fig 5a illustrate a data writing process, and figs 5b and 5c illustrate modifications of this process to obtain content sensitive writing.

[0016] Fig 6 illustrates schematically a situation where a number of files are stored among a number of data storage nodes.

[0017] Fig 7 illustrates the transmission of heartbeat signals.

[0018] Fig 8 is an overview of a data maintenance process.

DETAILED DESCRIPTION

[0019] The present disclosure is related to a distributed data storage system including a plurality of storage nodes. The structure of the system and the context in which it is used is outlined in Fig 1.

[0020] A user computer 1 may access, for example via the Internet 3, an application 5 running on a server 7. The user context, as illustrated here, may be a regular client-server configuration. However, it should be noted that the data storage system to be disclosed may be useful also in other configurations.

[0021] In the illustrated case, two applications 5, 9 run on the server 7. Of course however, this number of applications may be different. Each application may have an API (Application Programming Interface) 11 which provides an interface in relation to the distributed data storage system 13 and may support requests, typically write and read requests, from the applications running on the server. From an application's point of view, reading or writing

information from/to the data storage system 13 may be transparent and may be similar to using any other type of storage solution, for instance a file server or a hard drive.

[0022] Each API 11 may communicate with storage nodes 15 in the data storage system 13, and the storage nodes may communicate with each other. These communications may be based on TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

[0023] It should be noted that different APIs 11 on the same server 7 may access different sets of storage nodes 15. It should further be noted that there may exist more than one server 7 which accesses each storage node 15. This, however does not to any greater extent affect the way in which the storage nodes operate, as will be described later.

[0024] The components of the distributed data storage system may be the storage nodes 15 and the APIs 11, in the server 7 which access the storage nodes 15. The present disclosure therefore relates to methods carried out in the server 7 and in the storage nodes 15. Those methods will primarily be embodied as software/hardware combination implementations which may be implemented on the server and the storage nodes, respectively, and may together determining for the operation and the properties of the overall distributed data storage system.

[0025] The storage node 15 may typically be embodied by a file server which is provided with a number of functional blocks. The storage node may thus include a tangible storage medium 17, which typically comprises of a number of hard drives, optionally configured as a RAID (Redundant Array of Independent Disk) system. Other types of storage media are however conceivable as well. For instance, solid state drives, SSDs, or even random access memory, RAM, units may be used.

[0026] The storage node 15 may further include a directory 19, which may include lists of data entity/storage node relations as a host list, as will be discussed later.

[0027] In addition to the host list, each storage node may contain a node list including the IP addresses of all or some storage nodes in its set or group of storage nodes. The number of storage nodes in a group may vary from a few to hundreds of storage nodes. The node list may further have a version number.

[0028] Additionally, the storage node 15 may include a replication block 21 and a cluster monitor block 23. The replication block 21 may include a storage node API 25, and may be configured to execute functions for identifying the need for and carrying out a replication process, as will be described in detail later. The storage node API 25 of the replication block 21 may contain code that to a great extent corresponds to the code of the server's 7 storage node API 11, as the replication process may include actions that correspond to a great extent to the actions carried out by the server 7 during reading and writing operations to be described. For

instance, the writing operation carried out during replication corresponds to a great extent to the writing operation carried out by the server 7. The cluster monitor block 23 may be configured to carry out monitoring of other storage nodes in the data storage system 13, as will be described in more detail later.

[0029] The storage nodes 15 of the distributed data storage system may be considered to exist in the same hierarchical level. For example, there may be no appointed master storage node that is responsible for maintaining a directory of stored data entities and monitoring data consistency, etc. Instead, all storage nodes 15 may be considered equal from an hierarchical point of view, even though they may have different qualities as will be discussed in greater detail later. All storage nodes may, at times, carry out data management operations vis-à-vis other storage nodes in the system. This equality may help ensure that the system is robust. In case of a storage node malfunction other nodes in the system may provide services on behalf of the malfunctioning node (*e.g.*, make copies of files, provide access to files, answer requests, etc.) and ensure reliable data storage.

[0030] The operation of the system may be described in relation to the reading of data in the storage system, writing of data in the storage system, and/or data maintenance. Even though these methods work very well together, it should be noted that they may in principle also be carried out independently of each other. That is, for instance the data reading method may provide excellent properties even if the data writing method of the present disclosure is not used, and vice versa.

[0031] The reading method may be described with reference to figs 2A-2C and 3, the latter being a flowchart illustrating an example method.

[0032] The reading, as well as other functions in the system, may utilise multicast communication to communicate simultaneously with a plurality of storage nodes. By a multicast or IP multicast, it is meant a point-to-multipoint communication which may be accomplished by sending a message to an IP address which is reserved for multicast applications.

[0033] For example, a message, typically a request, may be sent to such an IP address (e.g. 244.0.0.1), and a number of recipient servers may be registered as subscribers to that IP address. Each of the recipient servers may have its own IP address. When a switch in the network receives the message directed to 244.0.0.1, the switch may forward the message to the IP addresses of each server registered as a subscriber.

[0034] In principle, a single server may be registered as a subscriber to a multicast address, in which case a point-to-point, communication is achieved. However, in the context of

this disclosure, such a communication may nevertheless be considered a multicast communication since a multicast scheme is employed.

[0035] Unicast communication may also be employed. Unicast communication may refer to a communication with a single recipient.

[0036] With reference to fig 2A and fig 3, the method for retrieving data from a data storage system may include the sending 31 of a multicast query to a plurality of storage nodes 15. For purposes of illustration, there are five storage nodes each having an IP (Internet Protocol) address 192.168.1.1, 192.168.1.2, etc. The number of storage nodes is, needless to say, just an example. The query contains a data identifier "2B9B4A97-76E5-499E-A21A6D7932DD7927", which may for instance be a Universally Unique Identifier (UUID).

[0037] The storage nodes may scan themselves for data corresponding to the identifier. If such data is found, a storage node may send a response, which is received 33 by the server 7, cf. fig 2B. As illustrated, the response may optionally contain further information in addition to an indication that the storage node has a copy of the relevant data. Specifically, the response may contain information from the storage node directory about other storage nodes containing the data, information regarding which version of the data is contained in the storage node, and/or information regarding which load the storage node at present is exposed to.

[0038] Based on the responses, the server may select 35 one or more storage nodes from which data is to be retrieved, and may send 37 a unicast request for data to that/those storage nodes, cf. fig 2C.

[0039] In response to the request for data, the storage node/nodes may send the relevant data by unicast to the server which receives 39 the data. For purposes of illustration, a single storage node may be selected. While this is sufficient, it is possible to select more than one storage node in order to receive two sets of data, for example to perform a consistency check. If the transfer of data fails, the server may select another storage node for retrieval.

[0040] The selection of storage nodes may be based on an algorithm that takes several factors into account in order to achieve a good overall system performance. For example, the storage node having the latest data version and the lowest load may be selected although other concepts are fully conceivable.

[0041] Optionally, the operation may be concluded by server sending a list to all storage nodes involved, indicating which nodes contains the data and with which version. Based on this information, the storage nodes may themselves maintain the data properly by the replication process to be described.

[0042] Figs 4A –4C, and figs 5a-c illustrate an example data writing process for the distributed data storage system.

[0043] With reference to fig 4A and fig 5a the method may include a server sending 41 a multicast storage query to a plurality of storage nodes. The storage query may include a data identifier and a query as to whether the receiving storage nodes can store a file. Optionally, if the file identity is included in the query, the storage nodes may check with their internal directories whether they already have a file with this name, and may notify the server 7 in the unlikely event that this is the case, such that the server may rename the file.

[0044] In any case, at least a subset of the storage nodes may provide responses by unicast transmission to the server 7. For example, storage nodes having a predetermined minimum free disk space may answer the query. The server 7 may receive 43 the responses which may include geographic data relating to the geographic position of each server. For instance, as indicated in fig 4B, the geographic data may include the latitude, the longitude and the altitude of each server. Other types of geographic data may however also be conceivable, such as a ZIP code or the like.

[0045] In addition to the geographic data, further information may be provided that serves as an input to a storage node selection process. In the illustrated example, the amount of free space in each storage node is provided and/or an indication of the storage node's system age and/or an indication of the load that the storage node currently experiences. The responses may be stored or cached for future use.

[0046] Based on the received responses, the server may select 45 at least two, in another example embodiment three, storage nodes in the subset for storing the data. The selection of storage nodes may carried out by means of an algorithm that take different data into account. The selection may be carried out in order to achieve some kind of geographical diversity. In an example, file servers are selected such that file servers in the same rack are not selected as storage nodes. Typically, a great geographical diversity may be achieved, even selecting storage nodes on different continents. In addition to the geographical diversity, other parameters may be included in the selection algorithm. For example, geographical diversity may be a primary criteria. In this example, as long as a minimum geographic diversity is achieved, e.g. free space, system age and current load may also be taken into account. Other criteria may serve as the primary criteria. It is advantageous to have a randomized feature in the selection process as will be discussed below.

[0047] The selection may include calculating, based on each node's storage node information (system age, system load, etc.) a probability factor which may correspond to a

storage node aptitude score. A younger system for instance, which is less likely to malfunction, may have a higher calculated score. The probability factor may thus be calculated as a scalar product of two vectors. For example, one vector may contain the storage node information parameters (or as applicable their inverses), and the other vector may contain corresponding weighting parameters.

[0048] The selection may then comprise semi-randomly selecting storage nodes, where the probability of a specific storage node being selected may depend on its probability factor. Typically, if a first server has a twice as high probability factor as a second server, the first server may have a twice as high probability of being selected.

[0049] The selection process for a file to be stored may be carried out based on responses received as the result of a multicast query carried out for that file. However, it would also be possible to instead use responses recently received as the result of a multicast query issued in relation to the storing of another file. As a further alternative, the server can regularly issue general multicast queries "what is your status" to the storage nodes, and the selection may be based on the responses then received. Thus, it may not be necessary to carry out a multicast query for every single file to be stored.

[0050] When the storage nodes have been selected, the data to be stored and a corresponding data identifier may be sent to each selected node, for example using a unicast transmission.

[0051] Optionally, the operation may be concluded by each storage node, for example after successful completion of the writing operation, sending an acknowledgement to the server. The server then may send a list to all storage nodes involved indicating which nodes have successfully written the data and which have not. Based on this information, the storage nodes may themselves maintain the data properly by the replication process to be described. For instance if one storage node's writing failed, there one or more files may be replicated to one more storage node in order to achieve the desired number of storing storage nodes for that file.

[0052] It may be that the servers are more or less equal in terms of basic characteristics, such as response time and reliability. Further, it may be that all data stored is considered equal from the storage system's point of view (*e.g.*, no data is of greater importance or of a higher priority than other data in the system). In another example, a differentiation in terms of services is provided, both vis-à-vis different types of data and different types of users. In other words, some data may be treated differently than others and/or some users/storage nodes may be treated differently than others.

[0053] For example, storage nodes may differ in terms of load, free disk space, system age, etc., which relate to how the storage nodes are and have been used. Further, the differences may relate to the type of storage node and its inherent capabilities.

[0054] For instance, as initially mentioned, a storage node may include different types of storage media, such a hard drives, solid state drives (SSDs), and random access memory (RAM), devices and/or the like. Reading from a hard drive may involve mechanical movement of a reader head and may therefore be slower that reading from a comparatively more expensive SSD, which may not include moveable parts. A RAM storage may be even faster, but may be considered less reliable, since a power blackout may clear the data. Also, different types of hard drives may have different reading times. In general, a storage node or an individual disk in a storage node may provide a certain response time rate, which may be expressed in different ways, e.g. average response time = 30 ms, or type=SSD, which may be used in the selection/-writing process as will be discussed.

[0055] Further, the quality and configuration of the storage media, and the premises in which it is installed, may affect the reliability of the storage, even though a replication process providing redundancy may be provided, as will be discussed later. For instance, each disk may have a probability of failure, which may be expressed as an annualized failure rate (AFR). Local disk configurations may also imply different level. For instance a RAID 1 configuration may increase the reliability greatly. Additionally, the environment in which the storage medium is installed may influences reliability. For instance, room cooling and humidity, uninterrupted power supplies, UPS, and 24 hours available surveillance maintenance staff are factors that may affect the overall reliability. Thus, a node or an individual disk in a node may provide a certain level of reliability that may be used in the selection/writing process.

[0056] Other factors that may be considered in the writing process may be energy consumption of the disk or storage node, territory/legislation of the storage node, etc.

[0057] In summary there may be defined a number of node properties including but not limited to: type of disk (e.g. hard drive or SSD), response time, redundancy configuration (e.g. RAID 1), reliability (e.g. AFR), environment (e.g. cooled premises with UPS), energy consumption, and/or the like.

[0058] These different properties of different storage nodes or disks may meet different requirements for different files or users.

[0059] For example, a word processor file may be stored on a comparatively slow disk, as a delay of half a second when opening a word processor file does not significantly obstruct the

user. Real time data on the other hand may require fast access. The same applies for data that may be accessed in a chain of steps, and where each link in this chain adds to the overall delay.

[0060] The level of reliability may also be a factor that is considered. Some data may be so critical that it should be stored on the best possible nodes or node disks, in terms of reliability. Other data may be easily restored if lost and yet other data may be simply unimportant, and such data may on the other hand be stored on any storage node.

[0061] Another option, to be used alone or in combination with high reliable storage nodes, for very important data is to increase the number of used storage nodes, e.g. three or four copies may be stored.

[0062] Another issue with data storage is energy consumption. Some environmental requirements imposed on a service may require that its data is stored only on disks with very low energy consumption, while response times are less important, even allowing a hard drive to go into a standby mode when not used.

[0063] Another example requirement may relate to legislation or policies. For instance, it may be the case that classified data may be required to be stored in a specific territory.

[0064] Another possibility is to obtain price differentiation, e.g. that users that purchase premium subscriptions may have their data stored on particularly fast and/or reliable disks.

[0065] As such, there are a number of different factors that may be met by matching files or users with specific sets of storage nodes in the storage cluster. This may be carried out by adding features to the writing method outlined above.

[0066] A method for implementing such matching may be outlined in two versions with reference to fig 5b and 5c.

[0067] In fig 5b, a storage query may be sent out by the server 7 as previously described (cf. 41, fig 5a). The responses may be issued by the storage nodes, 15, 15', 15'' which may include information about storage node characteristics. For example, the information about the storage node characteristics may include at least one of a response time parameter, a reliability parameter, a territorial parameter and/or an energy consumption parameter. When carrying out the previously described selection (cf. 45, fig 5a) the server may take those parameters into account, such that suitable servers are selected in view of file type, user priority, or other requirements.

[0068] In an example method, illustrated in fig 5c, the storage query, which may be sent out by the server 7, may include a storage node characteristics requirement, which may be based on one or more of the previously mentioned parameters. Then responses may be issued by the storage nodes, 15, 15'' which fulfil the storage node characteristics requirement. When carrying

out the previously described selection (cf. 45, fig 5a) the server may assume storage nodes that responded fulfil the node parameters, since the query provided that storage nodes respond on condition that they fulfil the requirements in question.

[0069] While this procedure may avoid irrelevant traffic, e.g. unicast transmission by storage nodes which do not fulfil the requirements, the former method as illustrated in fig 5b may allow greater flexibility, for example storage on sub-standard storage nodes in the event that there is an insufficient number of available storage nodes that fulfil the characteristics requirement.

[0070] The data writing method in itself may allow an API in a server 7 to store data in a very robust way, e.g. as excellent geographic diversity may be provided.

[0071] In addition to the writing and reading operations, the API in the server 7 may carry out operations that delete files and update files. These processes may be described in connection with the data maintenance process below.

[0072] The data maintenance process is may ensure that a reasonable number of non-malfunctioning storage nodes each store the latest version of each file. Additionally, it may ensure that no deleted files are stored at any storage node. The maintenance may be carried out by the storage nodes themselves. For example, the system may lack a dedicated "master" that takes responsibility for the maintenance of the data storage. By allowing any individual storage node to act as a master for a particular piece of data for a limited amount of time, improved robustness and reliability may be achieved, as the "master" may otherwise be a weak spot in the system.

[0073] Fig 6 illustrates schematically an example where a number of files are stored among a number of data storage nodes. For purposes of illustration, twelve nodes, having IP addresses consecutively numbered from 192.168.1.1 to 192.168.1.12, are depicted. Needless to say however, the IP address numbers need not be in the same range at all and there may be more or less than twelve nodes. The nodes are placed in a circular order to simplify the description, i.e. the nodes need not have any particular order. Each node may store one or two files identified, for the purpose of simplicity, by the letters A-F. However, each node may store many more files, and the files A-F are depicted for purposes of illustration.

[0074] With reference to fig 8, the method for maintaining data may include detecting 51 conditions in the data storage system that imply the need for replication of data between the nodes in the data storage system. A replication process 53 may be initiated upon detection 51 of the conditions. The result of the detection process 51 may be a list 55 of files for which the need

for replication has been identified. The list may further include data regarding the priority of the different needs for replication. Based on this list the replication process 53 is carried out.

[0075] The robustness of the distributed storage may depend on the number of storage nodes that maintain copies or instances of a specified file. To ensure reliability and robustness, a reasonable number of copies of each correct versions of a file be stored in the system. In the illustrated case, three copies of each file may be stored. However, should for instance the storage node with the address 192.168.1.5 fail, the desired number of stored copies for files "B" and "C" may be two, rather than the desired three.

[0076] One event that results in a need for replication may therefore be the mal-functioning of a storage node in the system.

[0077] Each storage node in the system may monitor the status of other storage nodes in the system. This may be achieved by letting each storage node emit a so-called heartbeat signal at regular intervals, as illustrated in fig 7. In the illustrated case, the storage node with address 192.168.1.7 may emit a multicast signal 57 to the other storage nodes in the system, which ,ay indicate that it is working correctly. This signal may be received by all other functioning storage nodes in the system or a subset of nodes carrying out heartbeat monitoring 59 (cf. fig 8). For purposes of illustration, the storage node with address 192.168.1.5 may be malfunctioning and may not emit any heartbeat signal. Therefore, the other storage nodes may notice that no heartbeat signal has been emitted by this node for a period of time, which may indicate that the storage node in question is down.

[0078] The heartbeat signal may include the storage node's address, and may also include its node list version number. Another storage node, which may listen to the heartbeat signal, ,ay discover based on the heartbeat signal that the transmitting storage node has a later version node list than it is currently storing. The listening storage node may then request that the transmitting storage node transfer its node list. In an example, the addition and removal of storage nodes into the system may be achieved by adding or removing a storage node and sending a new node list version to one single storage node. This node list may then be recursively spread to all other storage nodes in the system.

[0079] Again with reference to fig 8, each storage node may search 61 its internal directory for files that are stored by the malfunctioning storage node. Storage nodes which store files "B" and/or "C" may determine that a storage node that stores file "B" and/or "C" is mal-functioning storage node and may therefore add the corresponding file on their lists 55.

[0080] The detection process may however also reveal other conditions that imply the need for replicating a file. Typically such conditions may be inconsistencies, e.g. that one or

more storage nodes may have an obsolete version of the file. A delete operation may also imply a replication process as this process may carry out the actual physical deletion of the file. The server's delete operation indicate that the storage nodes set a deletion flag for the file in question. Each node may therefore monitor reading and writing operations carried out in the data storage system. Information provided by the server 7 at the conclusion of reading and writing operations, respectively, may indicate that one storage node contains an obsolete version of a file (e.g. in the case of a reading operation) or that a storage node did not successfully carry out a writing operation. In both cases the action may indicate that the replication process should occur such that the overall objects of the maintenance process are fulfilled.

[0081] In addition to the reading and writing operations 63, 65, additional processes may provide indications that a replication process may be initiated. For example, the deleting 67 and updating 69 processes may trigger the replication process.

[0082] The deleting process is initiated by the server 7 (cf. fig 1). Similar to the reading process, the server may send a query by multicasting to all storage nodes (or a subset thereof), in order to find out which storage nodes have data with a specific data identifier. The storage nodes may scan their systems for data with the relevant identifier and may respond by a unicast transmission if they have the data in question. The response may include a list, from the storage node directory, of other storage nodes containing the data. The server 7 may then send a unicast request, for example to the storage nodes that store the file, that the file be deleted. Each storage node may set a flag relating to the file and indicating that it should be deleted. The file may then be added to the replication list, and an acknowledgement is sent to the server. The replication process then physically deletes the file as will be described.

[0083] The updating process may include a search function, similar to the one of the deleting process, and a writing function, which may be similar to the one carried out in the writing process. The server may send a query, for example by multicasting to all storage nodes or a subset thereof, in order to find out which storage nodes include data with a specific data identifier. The storage nodes may scan themselves for data with the relevant identifier, and may respond by a unicast transmission if they have the data in question. The response may include a list, from the storage node directory, of other storage nodes containing the data. The server 7 then may send a unicast request, requesting that the storage nodes update the data. The request may contain the updated data. The storage nodes updating the data may send an acknowledgement to the server, which may respond by sending a unicast transmission containing a list with the storage nodes that successfully updated the data. The response may also include

storage nodes which did not successfully update the data. Again, this list may be used by the maintenance process.

[0084] Again with reference to fig 8 the read 63, write 65, delete 67, and/or update 69 operations may indicate that a need for replication exists. The heartbeat monitoring 59 may also indicated that replication should be performed. The overall detection process 51 may generate data regarding which files need be replicated. For instance, a reading or updating operation may reveal that a specific storage node contains an obsolete version of a file. A deletion process may set a deletion flag for a specific file. The heartbeat monitoring may reveal that a number of files, stored on a malfunctioning storage node, should be replicated to a new storage node.

[0085] Each storage node may monitor the need for replication for all the files it stores and may maintain a replication list 55. The replication list 55 may thus contain a number of files that should be replicated. The files may be ordered in correspondence with the priority for each replication. For example, there may be three different priority levels. In an example, the highest level may be reserved for files which the storage node holds the last online copy. Such a file may be quickly replicated to other storage nodes such that a reasonable level of redundancy may be achieved. A medium level of priority may relate to files where the versions are inconsistent among the storage nodes. A lower level of priority may relate to files which are stored on a storage node that is malfunctioning.

[0086] The storage node may replicate the files on the replication list 55 in accordance with their level of priority. The replication process may be described for a storage node which is here called the operating storage node, although all storage nodes may operate in this way.

[0087] The replication part 53 of the maintaining process may start with the operating storage node attempting 71 to become the master for the file it intends to replicate. The operating storage nodes may send a unicast request to become master to other storage nodes that are known store the file in question. The directory 19 (cf. fig 1) may provide a host list which may include information regarding which storage nodes to ask. In the event, for example in case of a colliding request, that one of the storage nodes does not respond affirmatively, the file may be moved back to the list for the time being, and an attempt may be made with the next file on the list. Otherwise the operating storage node may be considered to be the master of this file and the other storage nodes may set a flag indicating that the operating storage node is master for the file in question.

[0088] The operating storage node may find 73 all copies (or a subset thereof) of the file in question in the distributed storage system. For example, the operating storage node may send a multicast query to all storage nodes (or a subset thereof), requesting the identification of storage node that maintain copies of the file. The storage nodes that maintain copies the file may submit

responses to the query, and the responses may contain the version of the file they keep as well as their host lists, e.g. the list of storage nodes containing the relevant file that is kept in the directory of each storage node. These host lists sent to the operating node may be merged 75 by the operating storage node, such that a master host list may be formed corresponding to the union of all retrieved host lists. If additional storage nodes are found, which were not asked when the operating storage node attempted to become master, the request may also be sent to the additional storage nodes. The master host list may contain information regarding which versions of the file the different storage nodes keep and may illustrate the status of the file within the entire storage system.

[0089] Should the operating storage node not have the latest version of the file in question, this file may then be retrieved 77 from one of the storage nodes that do have the latest version.

[0090] The operating storage node may decide 79 whether the host list should be changed, for example if additional storage nodes should be added. If so, the operating storage node may carry out a process very similar to the writing process as carried out by the server and as described in connection with figs 4A-4C, and 5. The result of this process may be that the file is written to a new storage node.

[0091] In case of version inconsistencies, the operating storage node may update 81 copies of the file that are stored on other storage nodes, such that all files stored have the correct version.

[0092] Superfluous copies of the stored file may be deleted 83. If the replication process is initiated by a delete operation, the process may begin at this step. For example, once the storage nodes that maintain copies of the file to be deleted have accepted the deletion of the file, the operating storage node may request, for example via using unicast, all storage nodes to physically delete the file in question. The storage nodes may acknowledge that the file is deleted.

[0093] Further the status, e.g. the master host list of the file may be updated. It is then optionally possible to repeat 73-83 to ensure that a need for replication is no longer present. This repetition may result in a consistent master host list that need not be updated in step 85.

[0094] The replication process for that file may be concluded, and the operating storage node may release 87 the status as master of the file, for example by sending a corresponding message to all other storage nodes on the host list.

[0095] This system where each storage node may take responsibility for maintaining all the files it stores throughout the set of storage nodes may provide a self-repairing (e.g., in case of a storage node malfunction) self-cleaning (e.g., in case of file inconsistencies or files to be

deleted) system with excellent reliability. The system may be scalable and may store files for a great number of different applications simultaneously.

[0096] The invention is not restricted to the specific disclosed examples and may be varied and altered in different ways within the scope of the appended claims.

What Is Claimed:

1. A method for writing data in a data storage system including a plurality of storage nodes, from a server which accesses data in the data storage system, the method comprising:

sending a multicast storage query to the plurality of storage nodes;

receiving a plurality of responses from a subset of said plurality of storage nodes, each of said responses including a storage node property; and

selecting at least two storage nodes in the subset for storing said data based on said responses, wherein the selecting is based on a data property of the data to be stored and the storage node property of the at least two storage nodes in the subset.

2. The method according to claim 1, wherein said node property includes at least one of: a type of disk, a response time, a redundancy configuration, a reliability, a node environment, a territory, or an energy consumption.

3. The method according to claim 1, wherein the server associates the received data property with at least one storage node property, and the selection of the at least two storage nodes in the subset is based on the association.

4. The method according to claim 1, wherein the data property corresponds to a type of user that created or edited the data.

5. The method according to claim 1, wherein the data property is a file type.

6. A method for writing data in a data storage system, the data storage system including a plurality of storage nodes, the method employed in a server that accesses data in the data storage system, the method comprising:

selecting a set of storage node properties based on a data property of the data to be stored;

sending a multicast storage query to the plurality of storage nodes, said storage query including an indication of the set of storage node properties;

receiving a plurality of responses from a subset of the plurality of storage nodes, wherein the subset storage nodes comply with requirements of the set of storage node properties; and

selecting at least two storage nodes in the subset for storing said data based on the responses.

7. The method according to claim 6, wherein said set of node properties include at least two of: a type of disk, a response time, a redundancy configuration, a reliability, a node environment, a territory, or an energy consumption.

8. The method according to claim 6, wherein the data property corresponds to a type of user that created or edited the data.

9. The method according to claim 6, wherein the data property is a file type.

10. A device, the device included in a server, the device configured to access and write data to a data storage system, the data storage system including a plurality of storage nodes, the device configured to:

send a multicast storage query to a plurality of storage nodes;

receive a plurality of responses from a subset of said plurality of storage nodes, each of said responses including a storage node property; and

select at least two storage nodes in the subset for storing said data based on said responses, wherein the selecting is based on a data property of the data to be stored and the storage node property of the at least two storage nodes in the subset.

11. The device according to claim 10, wherein said node property includes at least one of: a type of disk, a response time, a redundancy configuration, a reliability, a node environment, a territory, or an energy consumption.

12. The device according to claim 10, wherein the device is further configured to associate the received data property with at least one storage node property, and select the at least two storage nodes in the subset is based on the association.

13. The device according to claim 10, wherein the data property corresponds to a type of user that created or edited the data.

14. The device according to claim 10, wherein the data property is a file type.

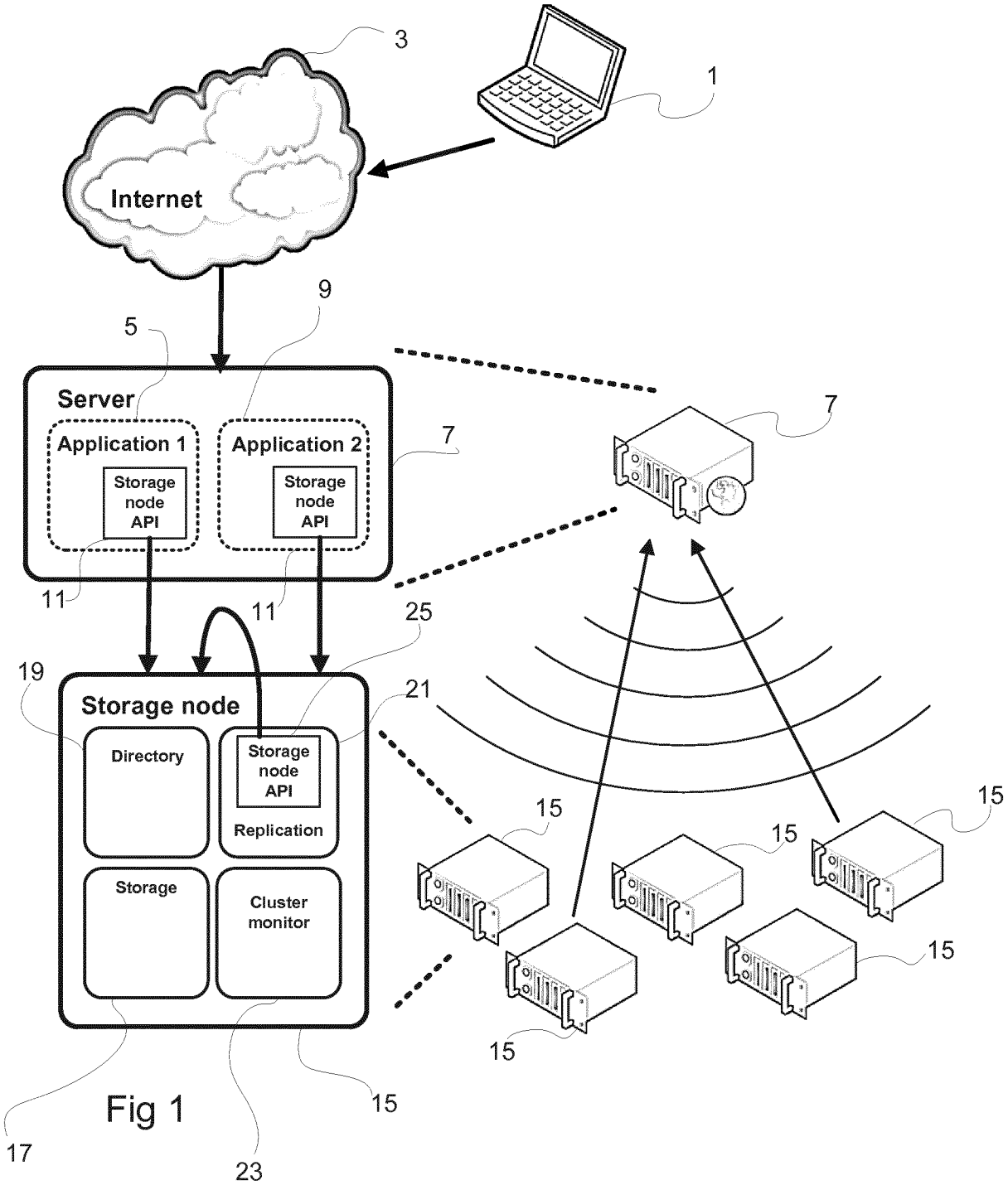
3
1
Internet
9
5
Server
Application 1
Application 2
Storage node API
Storage node API
7
7
11
11
25
19
21
Storage node
Directory
Storage node API
Replication
Storage
Cluster monitor
15
15
15
15
15
17
23
15

Fig 1

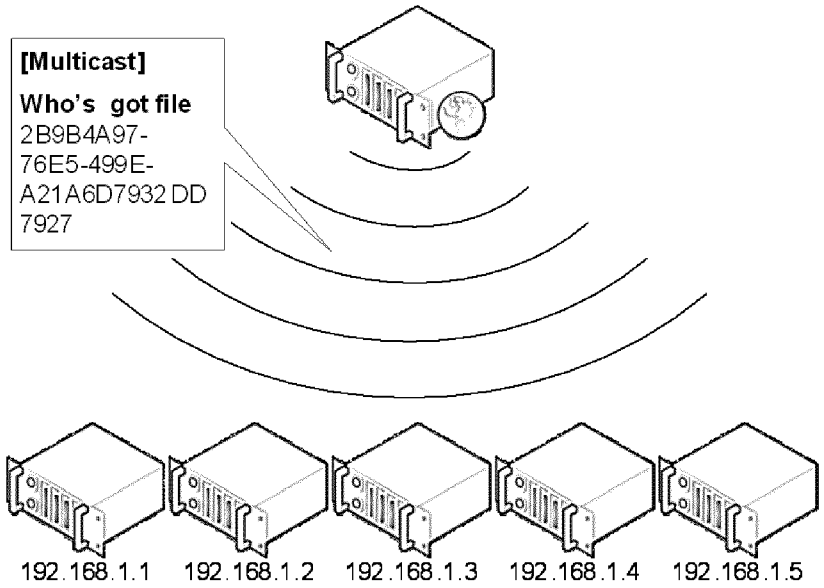

Fig 2A

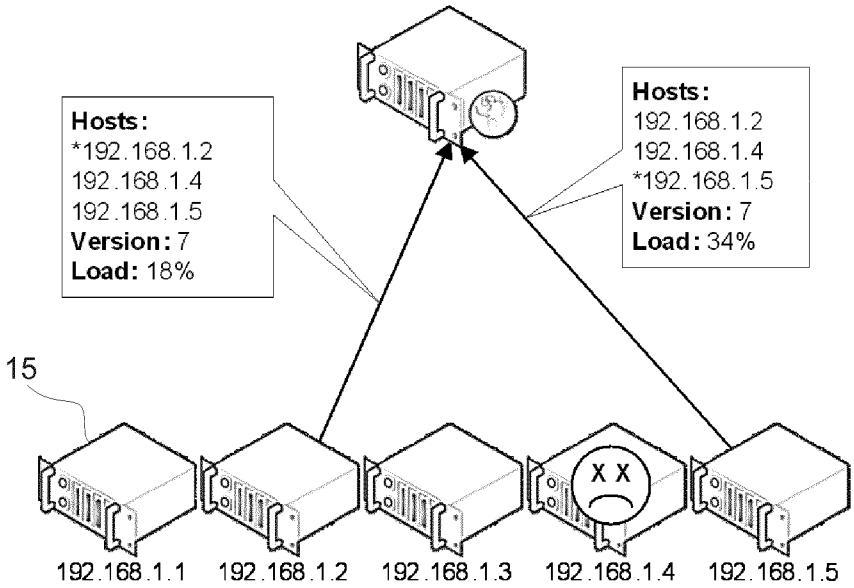

Fig 2B

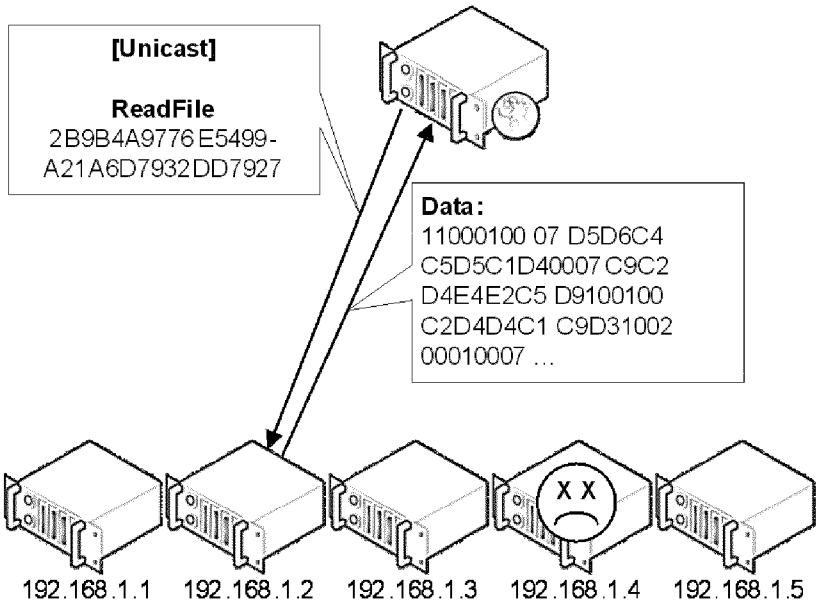

Fig 2C

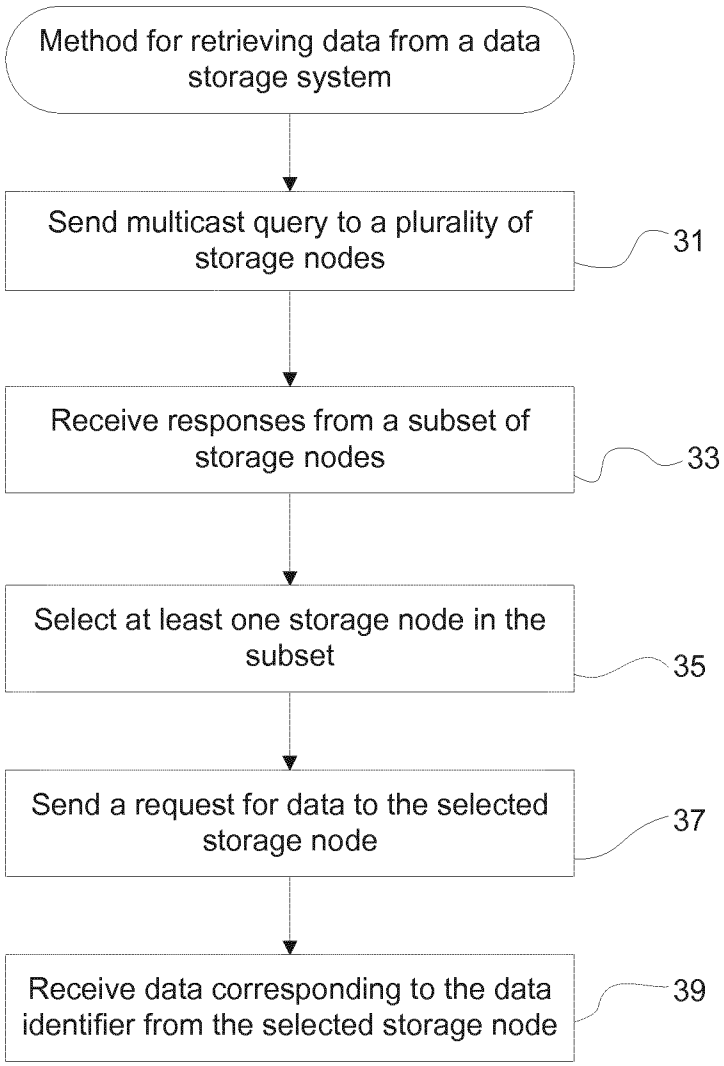
Method for retrieving data from a data storage system
Send multicast query to a plurality of storage nodes
31
Receive responses from a subset of storage nodes
33
Select at least one storage node in the subset
35
Send a request for data to the selected storage node
37
Receive data corresponding to the data identifier from the selected storage node
39

Fig 3

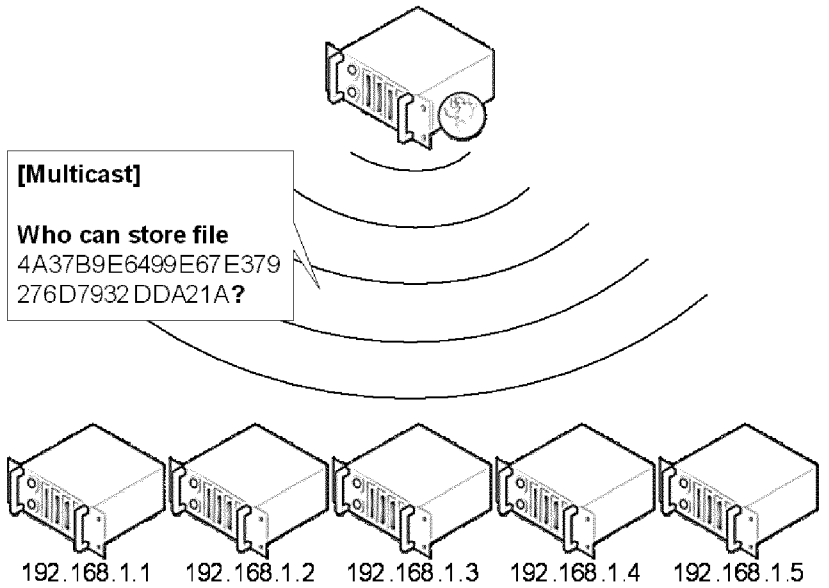

Fig 4A

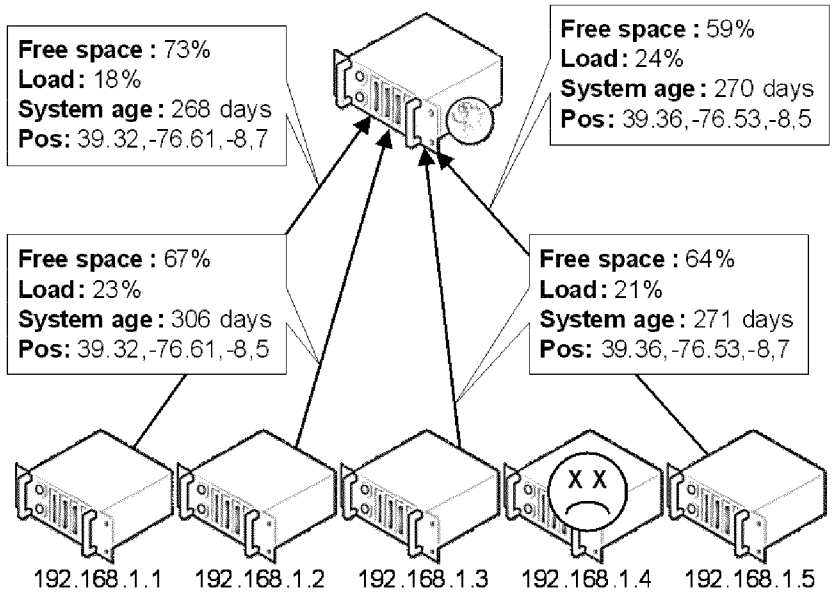

Fig 4B

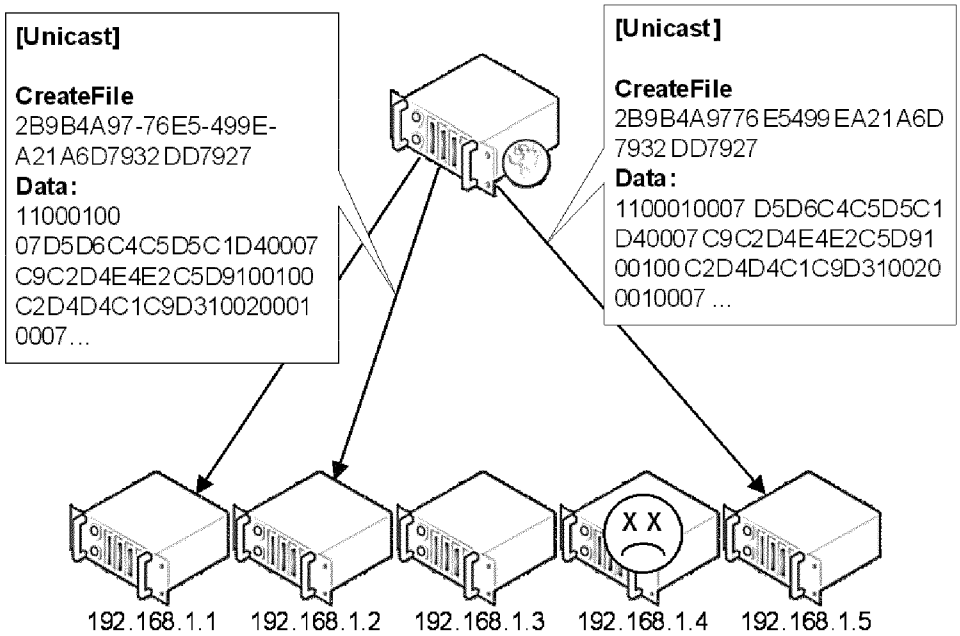

Fig 4C

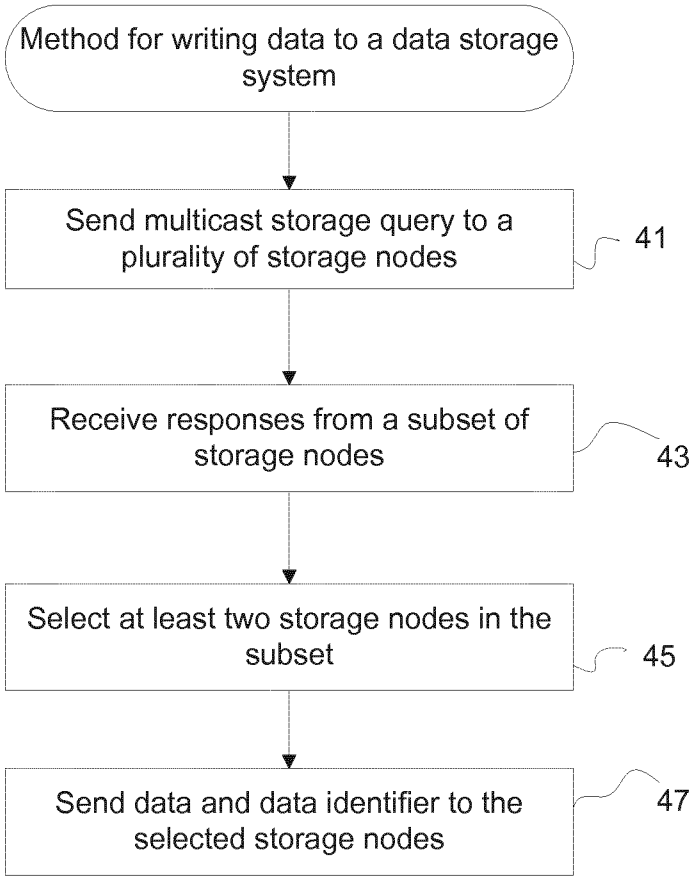

Fig 5a

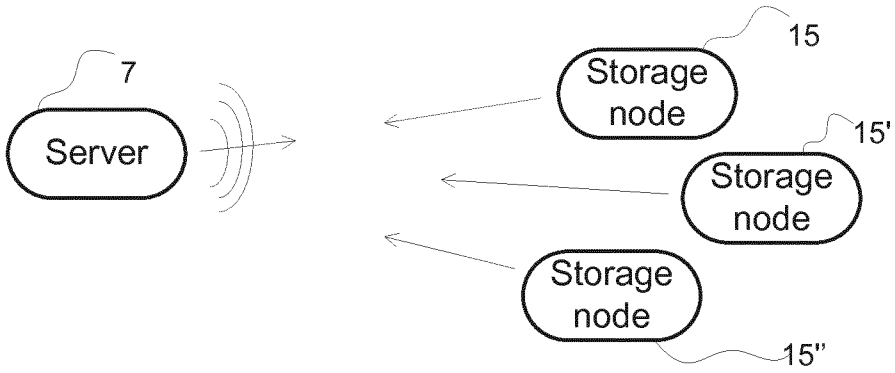

Fig 5b

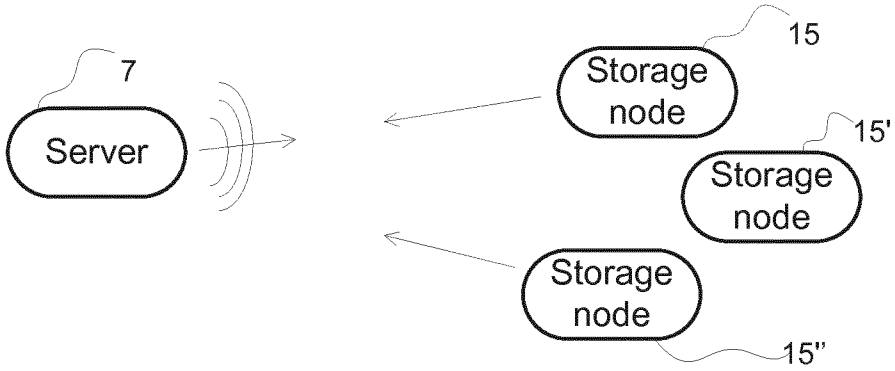

Fig 5c

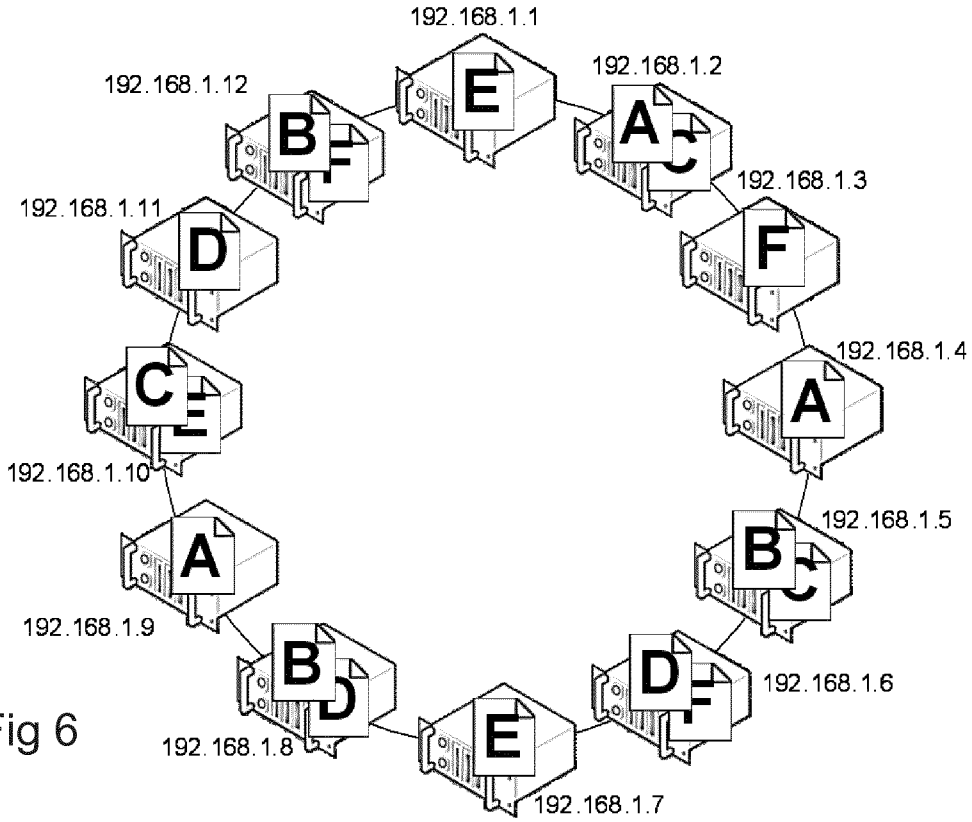
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
192.168.1.5
192.168.1.6
192.168.1.7
192.168.1.8
192.168.1.9
192.168.1.10
192.168.1.11
192.168.1.12

Fig 6

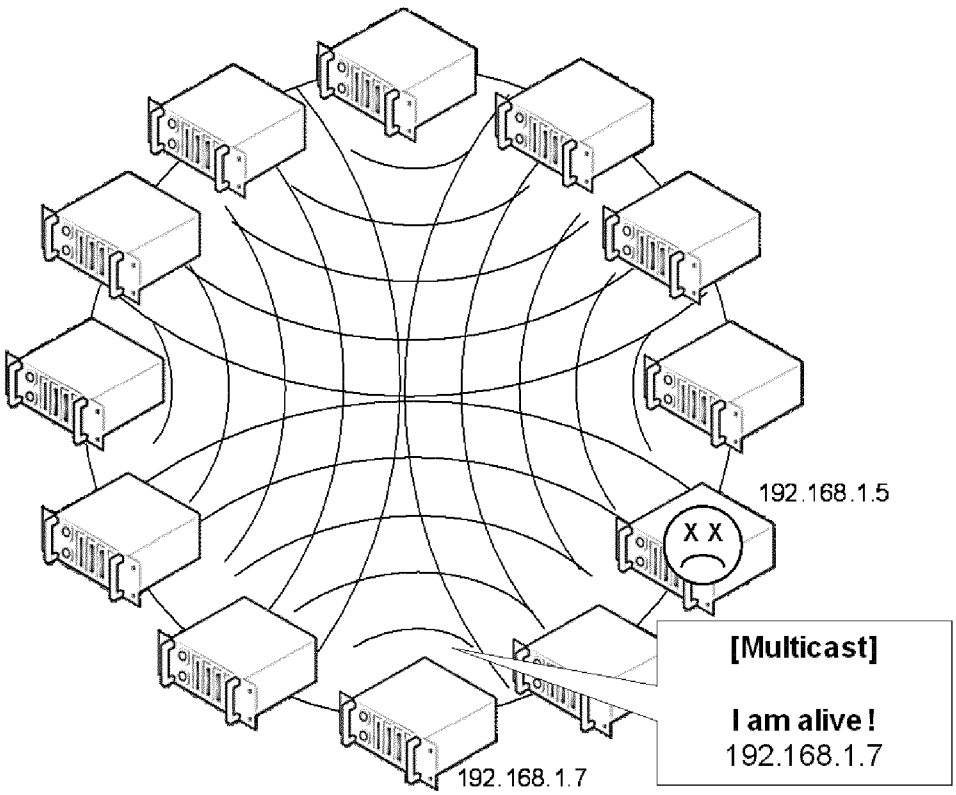
192.168.1.5
192.168.1.7
[Multicast]
I am alive!
192.168.1.7

Fig 7

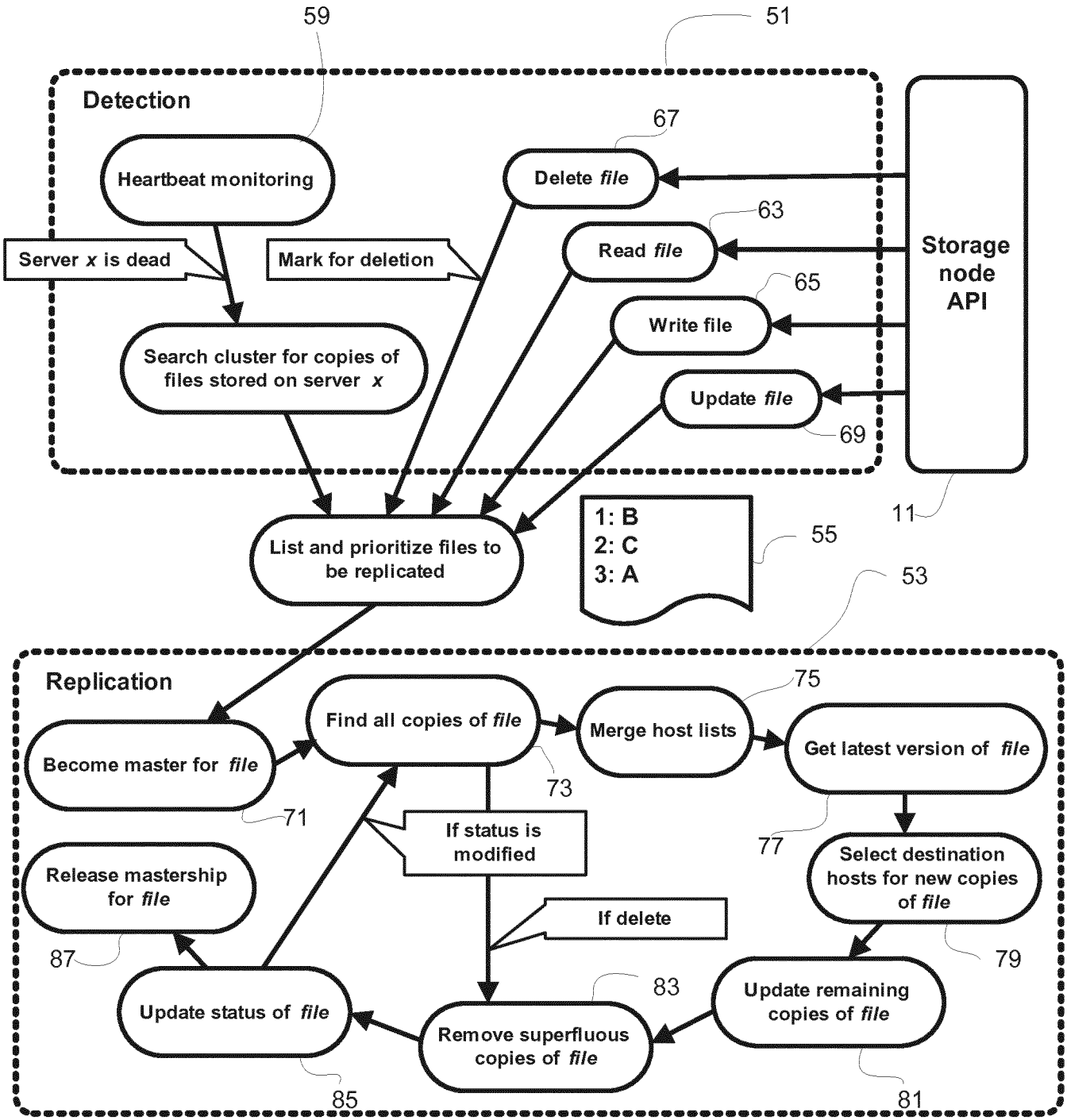
59
51
Detection
Heartbeat monitoring
Server x is dead
Search cluster for copies of files stored on server x
Mark for deletion
67
Delete file
63
Read file
65
Write file
Update file
69
Storage node API
11
List and prioritize files to be replicated
1: B
2: C
3: A
55
53
Replication
Become master for file
71
Find all copies of file
73
Merge host lists
75
Get latest version of file
77
Select destination hosts for new copies of file
79
If status is modified
If delete
Release mastership for file
87
Update status of file
85
Remove superfluous copies of file
83
Update remaining copies of file
81

Fig 8

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2012/066725

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F17/30
ADD.

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)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data, INSPEC, COMPENDEX

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2010/046393 A2 (ILT PRODUCTIONS AB [SE]; MELANDER CHRISTIAN [SE]; BERNBO STEFAN [SE];) 29 April 2010 (2010-04-29)	1-3,6,7, 10-12
Y	abstract page 1, line 1 - page 1, line 6 page 1, line 15 - page 3, line 15 page 3, line 28 - page 4, line 31 page 7, line 23 - page 9, line 6	4,5,8,9, 13,14
Y	US 2001/034812 A1 (IGNATIUS PAUL [US] ET AL) 25 October 2001 (2001-10-25) abstract paragraph [0008] paragraph [0010] paragraph [0034] - paragraph [0040] paragraph [0043]	4,5,8,9, 13,14

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* 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
25 October 2012	05/11/2012

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Boyadzhiev, Yavor

1

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2012/066725

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2005/283649 A1 (TURNER BRYAN C [US] ET AL) 22 December 2005 (2005-12-22) abstract paragraph [0013] paragraph [0018] - paragraph [0019] paragraph [0034] - paragraph [0035] paragraph [0046] - paragraph [0048] paragraph [0059] -----	1-14

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2012/066725

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2010046393 A2	29-04-2010	AU 2009306386 A1	29-04-2010
		CA 2741477 A1	29-04-2010
		CN 102301367 A	28-12-2011
		EA 201100545 A1	30-01-2012
		EP 2342663 A2	13-07-2011
		JP 2012506582 A	15-03-2012
		KR 20110086114 A	27-07-2011
		SE 0802277 A1	25-04-2010
		US 2011295807 A1	01-12-2011
		WO 2010046393 A2	29-04-2010
US 2001034812 A1	25-10-2001	AT 475929 T	15-08-2010
		EP 1384135 A2	28-01-2004
		HK 1062730 A1	22-10-2010
		US 2001034812 A1	25-10-2001
		WO 0155856 A2	02-08-2001
US 2005283649 A1	22-12-2005	EP 1769354 A2	04-04-2007
		US 2005283649 A1	22-12-2005
		WO 2005121962 A2	22-12-2005