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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 September 2012 (27.09.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/126794 A1

(51) International Patent Classification:
G06Q 10/00 (2012.01) *H04L 29/06* (2006.01)

(21) International Application Number:
PCT/EP2012/054537

(22) International Filing Date:
15 March 2012 (15.03.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11305319.3 21 March 2011 (21.03.2011) EP

(71) Applicant *(for all designated States except US)*: **THOMSON LICENSING** [FR/FR]; 1-5 rue Jeanne d'Arc, F-92130 Issy-les-Moulineaux (FR).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **LE MERRER, Erwan** [FR/FR]; c/o Technicolor R&D France, 1, avenue de Belle Fontaine, CS17616, F-35576 Cesson-Sévigné (FR). **NEUMANN, Christoph** [DE/FR]; c/o Technicolor R&D France, 1, avenue de Belle Fontaine, CS17616, F-35576 Cesson-Sévigné (FR). **ONNO, Stéphane** [FR/FR]; c/o Technicolor R&D France, 1, avenue de Belle Fontaine, CS17616, F-35576 Cesson-Sévigné (FR). **HEEN, Olivier** [FR/FR]; c/o Technicolor R&D France, 1, avenue de Belle Fontaine, CS17616, F-35576 Cesson-Sévigné (FR).

(74) Agents: **LORETTE, Anne** et al.; Thomson Licensing, 1-5 rue Jeanne d'Arc, F-92130 Issy-Les-Moulineaux (FR).

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

Declarations under Rule 4.17:

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

Published:

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

(54) **Title**: DECENTRALIZED METHOD FOR MANAGING GROUPS OF DATA AND GROUP MEMBERS

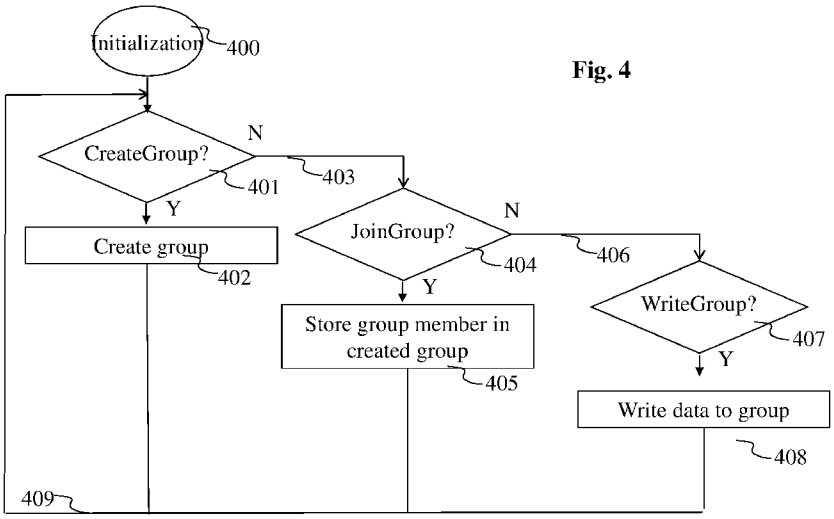

Fig. 4

(57) **Abstract**: We propose a completely distributed approach for group management requiring no central authority, possibly based on Distributed Hash Tables. As no enrollment to a central authority is required, the system can be leveraged by various applications that require group management.

WO 2012/126794 A1

Decentralized method for managing groups of data and group members.

1. Field of invention.

The present invention generally relates to digital data communication and access, and more particularly, to a method and device for exchanging data in networks between a plurality of groups of communicating entities.

2. Technical background.

Data sharing has become a sensitive topic in today's connected world. Social networks, define relationships between users of a social network application, and store (relations to) data that users are willing to share (pictures, videos, etc.) by means of these applications. In these networks, to capture interactions and access rights between users and data, the notion of "group" allows an improved use of the applications. For instance, the Facebook social network application partly relies on groups in order to restrain access to group data to group members. This notion comes in addition with the traditional restriction of facebook visibility: visible to friends, visible to friends of friends, world visible. Oneswarm, another example of social network application, applies a data sharing P2P (peer-to-peer) network (using social acquaintances) comprising "community servers" that maintain user groups. Persona, an online social network, is built to handle fine grained user privacy on user files; this is achieved through cryptography and attribute-based encryption, to capture accurately group relations. Prometheus, a social data management system, comprises a single group per participating user, which group assembles in essence simply friends of that user for securely storing the user files. The notion of a group abstraction level captures the social aspect that is at the core of the success of multiple services available on the Internet. Groups are used by the management mechanism of the social network application to check, refuse or grant access right for each interaction between two social network users, by controlling proactively who should be able to join or not a particular group.

But a wide adoption of social network applications is prohibited because there is no generic solution for access control based on groups of users. Rather, throughout the plurality of different social network applications,

the notion of group structure is used and implemented differently which forbids reusability of data in the plurality of applications.

In addition, social network applications such as Facebook or OneSwarm make use of central servers to handle the task of managing groups, or servers that are operated and controlled by a same authority. But with today's extension of high-speed digital data communication to nearly every home and the proliferation of electronic devices in these homes, a distributed approach is better suited to scalability and allows to embed such a discussed generic mechanism directly into the social network application program on those devices. This avoids the need for expensive server farms, removes the single point of failure and the unpleasant feeling of having "big brother" watching over user data and connections between users and data, and allows access control to occur close to the application itself. Even if, with current social network applications, user data is stored in a decentralized manner on user devices around the world, the system logic is still centralized, as access to a centralized server (e.g. the Facebook website) is necessary for discovering where to fetch the data.

There is thus a need for a clear definition of a group structure together with a need for a generic mechanism to manage groups at a large scale, in order to form a fundamental building block which improves current social-based applications and enables new ones, as opposed to the majority of current social network applications that simply provide a communication channel for interacting with friends. There is thus a need for rethinking the social network application in a distributed manner.

3. Summary of the invention.

The present invention aims at alleviating some inconveniences of prior art.

Notably, the invention proposes a method for managing groups of data and group members, the method being implemented by communicating devices connected in a communication network, and the method uses a cryptographical hashing function to determine a first identifier indicating one or more of the communicating devices for storing of data of at least one of

the group of data on one of the communicating devices, the method comprising a step of: upon reception of a request to create a group by a group creator, the request comprising the group identifier and a second identifier representative of the group creator, storing of data of the group on the communication device that is determined according to the function; and a step of: upon reception of a request from the group creator to insert a member in a group created by the group creator, the creator being identified by a the second identifier comprised in the request, the request further comprising a group member identifier, storing of the group member in the created group; and a step of: upon reception of a request to access data of a group to access, the request comprising a group identifier of the group to access and comprising an identifier identifying a requester of the request to access data, giving access to the data of the group to access if the identifier identifying the requester is stored in the group to access.

According to a particular embodiment of the method of the invention, the cryptographical hashing function implemented through a Distributed Hash Table.

According to a particular embodiment of the method of the invention,, the group member identifier is a group identifier.

According to a particular embodiment of the method of the invention, the request to create a group, the request to insert a member in the group and the request to access data of a group to access are being made accessible via an application programming interface.

The present invention also comprises a communicating device for managing groups of data and group members, the device being connected with other communicating devices connected in a communication network, the communicating device using a cryptographical hashing function to determine a first identifier indicating at least one of the communicating devices for storing of data of at least one of the group of data on one of the communicating devices, the device comprising means for reception of a request to create a group by a group creator, the request comprising the group identifier and a second identifier representative of the group creator, storing of data of the group on the communication device that is determined

according to the function; and means for reception of a request from the group creator to insert a member in a group created by the group creator, the creator being identified by a the second identifier comprised in the request, the request further comprising a group member identifier, storing of the group member in the created group; and means for reception of a request to access data of a group to access, the request comprising a group identifier of the group to access and comprising an identifier identifying a requester of the request to access data, giving access to the data of the group to access if the identifier identifying the requester is stored in the group to access.

4. List of figures.

More advantages of the invention will appear through the description of particular, non-restricting embodiments of the invention. The embodiments will be described with reference to the following figures:

Figure 1 shows an UML diagram of a group data structure according to the invention.

Figure 2 shows communication between an application and multiple nodes connected in a network according to the invention.

Figure 3 shows an example implementation of a node or communicating device according to the invention.

Figure 4 shows a flow chart of the method of the invention.

5. Detailed description of the invention.

Figure 1 shows an example of a group structure according to the invention. Distribution of the social network application is obtained through distribution of access management and control through the group structure of the present invention. The diagram represents an UML (Unified Modeling Language) relation model between entities of the group class according to the invention. UML is a standardized general-purpose modeling language used in the technical field of object-oriented design. Among others, UML allows to design object-oriented architecture with the help of so-called class diagrams. The class diagrams represent relationships between the classes of the object-oriented architecture.

According to the invention, members, member data and access to those elements are handled using such a group data structure as depicted.

The invention comprises a generic API (Application Programming Interface) that is provided on top of the group data structure of the invention, making manipulation of the group data structure according to the invention suited for adoption by a plurality of different social network applications, and thereby allowing data exchange between them. This unified view of the invention of members, groups and data allows a separation of roles: administration rights can be easily refused, granted and extended per group, while access to data within a group can occur without the need for an administrator to be online. The invention allows a distributed implementation of this group data structure (and then the building of an access control based on groups), thus removing the “big brother” effect of a centralized operator, while increasing scalability of social network application systems using it; when the group data structure is implemented in a decentralized manner according to the invention, a social networking system using the invention is easily scalable for use by only several users (e.g. 1, 2, 10, 100) to use by many users (e.g. 1000, 10,000, 100,000 or more). Note that we make a difference between a social network application and a social network application system. The social network application uses the social network application system to execute the social network application. While social network application is typically a web application, a social network application system, in the scope of the invention, is a plurality of communicating entities organized in a network that have or have no storage and that are comprised with an application programming interface according to the invention.

Now returning to fig. 1, according to the UML notation used in the diagram of Figure 1,

- an hollow diamond shape on a class end means an aggregation relationship (‘has a’). Aggregation can occur when a class is a collection or container of other classes, but where the contained classes do not have a strong life cycle dependency on the container. I.e. if the container is destroyed, its contents are not.

- a filled diamond shape on a class end means a composition relationship (‘owns a’). Composition usually has a strong life cycle dependency between instances of the container class and instances of the contained class(es): If the container is destroyed, normally every instance that it contains is destroyed as well.

- a hollow triangle shape on a class end means a generalization relationship (‘inheritance’ or ‘is a’). The generalization relationship indicates

that a related class, the subclass, is considered to be a specialized form of a superclass, which is considered as a 'generalization' of the subclass.

The arrow used in diagram of Figure 1 indicates “usage”, i.e. 1 to many Social Apps (for social network application) 10 uses the Group class 12, which is indicated by arrow 100.

An optional notation at each line end indicates the multiplicity of instances of that entity (the number of objects that participate in the association):

0..1	No instances, or one instance (optional, may)
1	Exactly one instance
0..* or *	Zero or more instances
1..*	One or more instances (at least one)

A **group** 12 according to the invention is designed to be the central component that is used by social network applications 10 to provide access control to data or comprising information such as membership to a group, or a friend list. Being a member of a group 12 allows multiple actions, that are defined by the group 12 creator. Examples actions are: listing other members of a group 12, or sending a message to one particular group member or to all group 12 members. A group 12 is composed by a **list** 11 listing members 13 or groups 12, a member being a mere instance of a group, and a **whiteboard** 14 containing data like digital content or reference to such data. A group 12 list can contain **members** 13 (such as pseudonyms that member 13 chooses to be used within a given social network application 10) or other groups 12. A whiteboard 14 contains **objects** 15 (that are in practice the resources/files that a member 13 wants to share with some predefined group 12 of members 13), and various system information. Note that the design of figure 1 allows implementation of complex objects with this group notion: a member 13 is simply a group 12 with optional list 11 and whiteboard 14. Then, as an example, a folder containing pictures of a member’s 13 last holidays is a group 12, with pictures referenced in that group’s 12 whiteboard 14. The social network application 10, makes use of groups 12 generated by members 13, in order to provide access control in the social network application system.

Remark: prior-art social network applications merely use groups to register members within these groups. According to the invention, the group 12 is an abstraction for different kinds of elements. This allows for innovative applications like registering groups 12 in members 13, registering members 13 in members 13, registering members 13 in groups referencing digital content in their whiteboard, etc.

A **member** 13 can have several **roles** in the social network application system. Firstly, a member 13 creates as many identities as he wants, which we call “pseudos”. Secondly, a member 13 creates and administers one or more groups 12. Other identities and groups 12 may join the created group 12 and thus become members 13 of group 12. Note that we use the word ‘member’ as a generic term for any item in the list of a group or a member of a group. According to a particular embodiment of the invention, for each group 12 three roles exist: group 12 **creator**, group 12 **administrators** and group 12 **members**. The distinction is made according to an access right hierarchy. A group 12 creator has the right to create, destroy or inactivate his group 12. A group 12 administrator has the right to add or remove members to the group he administers. A group 12 member can access messages sent to the group 12 and can himself send messages to the group 12 he belongs to..

If the list 11 of group 12 members is accessible to group 12 members (decided by the group 12 creator), group 12 members can send messages to each other group 12 member individually. Note that a member 13 may have several of above roles (group creator, group administrator, group member (= simple member) for a given group 12.

According to a particular embodiment of the invention, the group member identifier is a group identifier. This allows to have less classes/entities in the group framework, and then eases development and improves protocol clarity.

Example member pseudos are human readable like “Franklin Family”, carrying some cryptographic semantic like SHA256(“Franklin Family”) (SHA256 is an example of a cryptographical hashing function), carrying time semantic like “Franklin Family-24-02-2011-18:05:42,23”, or carrying no semantic at all, like a long random number.

According to a particular implementation of the invention, the invention uses a so-called DHT (Distributed Hash Table). One of the advantages of using DHT is that it allows for load balancing, which makes it possible for the invention to be implemented on a set of devices with only moderate resources, such as Internet gateways. DHT is a class of a decentralized distributed system that provides a lookup service similar to a hash table; (*key*, *value*) pairs are stored in a DHT, and any participating node (device) can efficiently retrieve the value associated with a given key. Responsibility for maintaining the mapping from keys to values is distributed among the nodes (devices), in such a way that a change in the set of participants causes no disruption. This allows a DHT to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures. Such a DHT provides basic PUT and GET operations to respectively store and retrieve items in a distributed manner among participating nodes (devices). A DHT is opposed to a centralized lookup service which is vulnerable because it forms a single point of failure. The DHT uses a key-based routing for retrieving and storing data. In a DHT, the nodes (devices) collectively form the system without central coordination, and makes such a system relatively fault tolerant and scalable. The structure of a DHT can be decomposed into several main components. The DHT uses an abstract keyspace, such as a set of n-bit strings. A keyspace partitioning scheme splits ownership of this keyspace among the participating nodes. An overlay network then connects the nodes, allowing them to find the owner of any given key in the keyspace. Once these components are in place, a typical use of the DHT for storage and retrieval might proceed as follows. Suppose the keyspace is the set of n-bit strings. To store a file with given *filename* and *data* in the DHT, a SHA-1 hash of *filename* is generated, producing an n-bit key *k*, and a message *put*(*k*,*data*) is sent to any node participating in the DHT. The message is forwarded from node to node through the overlay network until it reaches the single node responsible for key *k* as specified by the keyspace partitioning. That node then stores the key and the data. Any other node can then retrieve the contents of the file by again hashing *filename* to produce *k* and asking any DHT node to find the data associated with *k* with a message *get*(*k*). The message will again be routed through the overlay to the node responsible for *k*, which will reply with the stored *data*.

According to a variant embodiment of the invention, additionally to storing data they are responsible for, nodes are perform basic security

checking: for instance, if a user claims the right to access the data of a given group, the node must enforce the protocol, i.e. check if the user is indeed a member of the group concerned by the query.

The set of participating nodes (devices) on which the invention is built could be in a datacenter, or linked thanks to the Internet (this covers the possibility that participating nodes belong to various authorities) or other types of data communication networks; users of the invention do not have to participate in this service, i.e. users of the invention do not have to run nodes (devices) of the DHT. They may do so, like Skype users. They may not do so, in which case they are like users of a social network that do not need to install a specific client in order to use services that the social network offers.

In a fully integrated service, the invention is implemented as part of the social network application, removing the need for an extra service provider.

Note that in the following description, "hash(UID_g)" stands for applying a hash function to the UID group, and UID_g, UID_l and UID_wb being keys generated at group creation, for example generated by means of an address mapping function, typically an injective function.

- The group 12 structure is stored at an address determined by the function "hash(UID_g)" on the DHT, for example using a PUT operation, one of the operations provided by a DHT, with a reference pointing to the List 11, and another one pointing to the Whiteboard 14
- The List 11 is stored on nodes holding address hash(UID_l) on the DHT
- The whiteboard 14 is stored on nodes holding at address hash(UID_wb) on the DHT

According to a variant embodiment of the invention, the data structure of figure 1 is not implemented on a single node, but rather distributed over multiple nodes, such as a node for list 11, and another node for whiteboard 14, and so on; this has several advantages, such as load balancing, making it possible for nodes implementing the invention to propose only moderate resources, but also has a security advantage, one single node only storing a small part of the data stored in the whole system.

Each of the previously discussed roles (group creator, group administrator, group member) in the social network application system

according to the invention owns an **inbox**. Inboxes for, members, administrators and creators are created in the DHT. They basically store events and request concerning member's current role. Each inbox is stored at hash(key), with key being an UID generated when creating such a role in our system.

The DHT system then attributes a physical node (device) to be the storage host of each of those structures, and enforces basic security checks. Simply knowing UID_g, which is advertised on websites (or simply send by an invite or by reference copy in an email), users know which DHT node to interact with, in order to access to functionalities provided by group management. Operations on the group structure figure 1 is provided as an API.

Such an API comprises functions as:

- Create a group
  Destination Address: Hash(UID_g)
  Parameters: UID_a, optional elements: symbolic_name, public cryptographic keys
  Return Value: ACK, NACK
  Initial sender: Any communicating entity

When a communicating entity (= group creator) creates a group (e.g. with symbolic name "Franklin Family"), the system generates a group identifier denoted UID_g for uniquely identifying this group.

The group structure is stored at hash(UID_g) in our group management system (on the DHT);

Then, group list and optionally whiteboard are created, each of them stored on nodes at hash(UID_l) and hash(UID_wb) respectively. The DHT automatically replicates the data on neighbor nodesfor reliability and availability purposes according to the particular policy of the DHT. The group structure on hash(UID_g) points to hash(UID_l), hash(UID_wb). A new UID, UID_a (for group administrator) is generated. The system also creates an inbox for the administrator at address hash(UID_a).

- Join a group
  Destination Address: Hash(UID_a)

Parameters: X
Return Value: ACK, NACK
Initial sender: Any communicating entity identified by X

As the UID of the group that a communicating entity X wants to join, UID_g, is public (or sent with an invite), a communicating entity forges a query to join this group indicating UID_g to the system. The system finds the node responsible for the group at hash(UID_g), using a GET operation. System then obtains the administrator's inbox address in the DHT (because indicated in the group structure), so the JOIN request is sent to his inbox. When the administrator comes online (i.e. launch application using our group management system), it checks all its group inboxes. Consequently the administrator gets the JOIN request from communicating entity X. The administrator may then update the list using the function "store member list", or reject request.

- Store member list (action performed by group administrator)
  Destination Address: Hash(UID_l)
  Parameters: List of identities of group members
  Return Value: ACK, NACK
  Initial sender: Administrator of group UID_g identified by UID_a

A group administrator can arbitrarily decide to add communicating entities to a group. This is done by storing on the node hosting the list structure at hash(UID_l) a list with the identities (such as X) of the new members .

- Retrieve members list of a group (= request to access data of a group)
  Destination Address: Hash(UID_l)
  Parameters: none
  Return Value: List of identities of group members
  Initial sender: Any communicating entity

A communicating entity that wants to access a list must send a request to the hosting node of the list at Hash(UID_l). The result is returned if the entity is allowed such an action (i.e. already a member and administrator/creating have allowed to list members)

- Write to a communicating entity identified by Y of the system (= second example of a request to access data of a group)

Destination Address: Hash(Y)
Parameters: message
Return Value: ACK, NACK
Initial sender: Any communicating entity

A communicating entity that wants to send a message to communicating entity identified by Y of the system, sends the message to the node at Hash(Y).

- Write to a group's whiteboard (= third example of a request to access data of a group)

Destination Address: Hash(UID_g)
Parameters: message
Return Value: ACK, NACK
Initial sender: Any communicating entity

A communicating entity that wants to share an object (picture, reference to a data) with members of group g, sends a write request to the hosting node of group g at Hash(UID_g). Hosting node should check if requesting node is in the group list (so he has the right to write in the group's whiteboard), in order to acknowledge or not the write operation.

We now present an example application, as an illustration which uses the mechanism for group management of the invention. Reference is made to Figure 1. In a social network application, members want to provide and restrict access to their data (pictures, videos, etc.) to people they know (friends, family, co-workers, etc.) or to identified categories of people (Action film fans, researchers in security, etc.). A system handling access control is then needed. Possibly, they want to keep data locally, that is, on their trusted devices.

In this sharing scenario, members of a given group are able to read and write to a group's 12 whiteboard 14, as enforced by the protocol (API) of

the invention. This whiteboard 14 basically contains references to data shared by group 12 members (a member 13 sharing a list of pictures on a given subject, actually shares the address of a group in the DHT, which itself is a list of all the pictures in that folder). References can then be added by members, simply by writing them on the whiteboard 14. A reference is in this context is the name of a resource, along with an URL providing for instance an IP address, a port and a unique identification string (e.g. http://121.14.12.16:9001/x2a58CzT). To retrieve a shared file, member of a group 12 then follows the URL to identify the host machine, and connects to it in order to download the resource, provided that the identification string is correct. This method forbids members 13 that are not members of this sharing group 12 to be aware of the location and unique string needed to get the data, as this information is readable only by group 12 members. When the access to a group 12 is removed to a given member's 13 pseudo, the group 12 administrator has to contact group 12 members that are sharing resources in that group 12, in order to ask their sharing application to regenerate unique strings for their shares (and then to update the whiteboard). This has the consequence that a removed member cannot access anymore to addresses it previously knew (when he was a member). Then ensure a property called "forward" and "backward" secrecy in the domain of security, as the revoked member will not be able to access to newly inserted references in the group's 12 whiteboard 14. According to a variant embodiment, being a member of a group may give access to a group secret key. A user willing to access to retrieve a file will be challenged by the node hosting that resource, in order to be sure that it knows the secret key and then is part of the group.

To sum up, sharing is then restricted to a list of members of a group 12. One of the advantages of the invention with regard to prior art is that the group structure is so designed that the integrity of a group 12 structure is maintained in a totally distributed fashion, thus obtaining load balancing and data security advantages, and members' 13 roles provide ease of use.

Similarly, it is possible to trivially build social applications like for instance instant messaging, mailing lists, download trackers (e.g. BitTorrent trackers) or Twitter, simply using the protocol (API) according to the invention.

Figure 2 illustrates an example application and inter-node communication for multiple nodes connected in a network according to the invention, illustrated by means of an example usage scenario. Nodes A (11), B (12), C (13) and D (14) are nodes according to the invention. The nodes A-D (11-14) are interconnected in a network (15). Arrows (100-104) indicate communication flows. Device M (10) is a smartphone with web access. Device A (11) provides an entry point to the network and providing a social network application via a web interface.

I) Device M (10), using the social networking application web interface provided by node A (11), issues a group creation request (100) (i.e. API call 'CreateGroup (UID_g)'). Upon reception of the request, the first thing for node A (11) is to help M to find out the IP address of the node that is responsible for storing the group with IDUID_g. According to the present scenario, A(11) does not know the responsible node, so it addresses a request (101) to node B (12) (the closest it knows from the target responsible node). However, node B is also not responsible for this group with this ID, so it forwards the request (102) to node C. Node C is responsible for this group ID and returns its IP address (103) to device M (10). For further communication with regard to this group, the device M (10) directly communicates (104) with node C (13). For example:

II) Using the social networking application provided by Node C (13) (if this node is no longer responsible for that group, a DHT request is triggered to find the new responsible node for h(UID_g)), device M (10) wishes to add a member that wants to join the group UID_g. To this end, the API call 'JoinGroup(UID_g)' is used.

III) Device M (10) wishes to list the members of group UID_g. the social networking application provided by Node C (13), device M (10) uses API call 'ListGroup(UID_g)'. The call is directly addressed to Node C (13) by the social networking application.

In the above scenario, no access rights are verified. According to a particular embodiment of the invention, the nodes also have an access security role to fulfill, in addition to their role as data storage. For example, Node C (13) verifies in I) if the user using device M (10) to send the ListGroup request has sufficient access rights to list the members of the group (by for instance being part herself of that group).

Figure 3 shows an example implementation of a node or communicating device according to the invention. The device 13 comprises the following components, interconnected by a digital data- and address bus 135:

- a processing unit 132 (or CPU for Central Processing Unit);
- a non-volatile memory NVM 130 ;
- a volatile memory VM 131 ;
- a clock unit 133, providing a reference clock signal for synchronization of operations between the components of the device 13 and for other timing purposes;
- a network interface 134, for interconnection of device 13 to other devices connected in network 15.

It is noted that the word "register" used in the description of memories 130 and 131 designates in each of the mentioned memories, a low-capacity memory zone capable of storing some binary data, as well as a high-capacity memory zone, capable of storing an executable program, or a whole data set.

Processing unit 132 can be implemented as a microprocessor, a custom chip, a dedicated (micro-) controller, and so on. Non-volatile memory NVM 130 can be implemented in any form of non-volatile memory, such as a hard disk, non-volatile random-access memory, EPROM (Erasable Programmable ROM), and so on. Non-volatile memory NVM 130 comprises notably a register 1301 that holds a program representing an executable program comprising the method according to the invention. When powered up, the processing unit 132 loads the instructions comprised in NVM register 1301, copies them to VM register 1311, and executes them.

The VM memory 131 comprises notably:

- a register 1311 comprising a copy of the program 'prog' of NVM register 1301 ;
- a register 1312 comprising, for DHT purposes:
  - IP/port addresses of neighbors in the DHT structure (leafset and routing table in Pastry for instance)
- A register 1213 comprising, for the group management mechanism (non exhaustive list, cf Figure 1):
  - List of groups hosted by current node
  - List of groups (i.e. of members also) of a particular group

- Whiteboard (i.e. arbitrary data) for listing items stored in a particular group; and
- Cryptographic material for enforcing group management.

- A further register 1314 comprises the data storage area for storing the data that it is responsible for.

A device such as device 13 is suited for implementing the method of the invention of automatic management of a collection of images, the device comprising

- means for reception (134) of a request to create a group by a group creator, the request comprising the group identifier and a second identifier representative of the group creator, storing of data of the group on the device (13) that is determined according to the function;
- means for reception (134) of a request from the group creator to insert a member in a group created by the group creator, the creator being identified by a the second identifier comprised in the request, the request further comprising a group member identifier, storing of the group member in the created group;
- means for reception (134) of a request to access data of a group to access, the request comprising a group identifier of the group to access and comprising an identifier identifying a requester of the request to access data, giving access to the data of the group to access if the identifier identifying the requester is stored in the group to access.

Other device architectures than illustrated by fig 3 are possible and compatible with the method of the invention. Notably, according to variant embodiments, the invention is implemented as a pure hardware implementation, for example in the form of a dedicated component (for example in an ASIC, FPGA or VLSI, respectively meaning Application Specific Integrated Circuit, Field-Programmable Gate Array and Very Large Scale Integration), or in the form of multiple electronic components integrated in a device or in the form of a mix of hardware and software components, for example a dedicated electronic card in a personal computer.

Figure 4 shows a flow chart of the method of the invention. In a first step 400, variables are initialized that are needed for execution of the method. In a first decisional step 401, it is determined if a received request is a request to create a group. If so, in a step 402 a group identifier UID_g is created for uniquely identifying the created group. The group structure is stored at hash(UID_g) and a replication copy is stored automatically by the DHT on neighbor nodes. The group structure on hash(UID_g) points to hash(UID_l), hash(UID_wb). A new UID, UID_a (for group administrator) is generated. The system also creates an inbox for the administrator at address hash(UID_a). Then, the method is repeated (409).

If the determination if the request is a create group request is negative, it is determined in a step 404 if the request concerns an insertion of a member in a created group. If so, in a step 405, the DHT finds the node responsible for the group at hash(UID_g), using a GET operation. The DHT then obtains the administrator's inbox address in the DHT (because indicated in the group structure), so the JOIN request is sent to his inbox. When the administrator comes online (i.e. launch application using our group management system), it checks all its group inboxes. Consequently the administrator gets the JOIN request from communicating entity X. The administrator may then update the list using the function "store member list", or reject request. Then, the method is repeated (409).

If the determination if the request is a create group request or a join group request is negative, it is determined in a step 407 if the request concerns a write group. If so, in a step 408, the hosting node of group g at hash(UID_g) checks if requesting node is in the group list (so he has the right to write data to the group), in order to acknowledge or not the write operation. If it is, the write operation is executed in step 408. Then, the method is repeated with step 409.

As the invention proposes a distributed implementation, it is particularly suited to be widely implemented in connected devices without the need to access a third party service for securing and providing privacy to users' interactions. This includes home environments devices, such as home gateways, that are core devices for service deployment. Doing so could enable privacy-preserving applications that leverage the social aspect of user targeted applications, as data resource sharing for instance. The fact that the invention provides a generic application interface allows for reusability,

meaning that different services may leverage one single installation of our system on a given set of devices, thus reducing deployment cost and complexity. As the group structure according to the invention allows precise distribution of roles, the resulting separation of those roles increase the usability of the group management service (many operations can occur in parallel, without the need for users to be online at all times).

CLAIMS

1. Method for managing groups of data and group members, said method being characterized in that it is implemented by communicating devices (13) connected in a communication network, and that the method uses a cryptographical hashing function to determine a first identifier indicating at least one of the communicating devices for storing of data of at least one of the group of data on one of the communicating devices, the method comprising the following steps:

- Upon reception of a request to create a group by a group creator, the request comprising the group identifier and a second identifier representative of the group creator, storing of data of the group on the communication device that is determined according to the function;
- Upon reception of a request from the group creator to insert a member in a group created by the group creator, the creator being identified by a the second identifier comprised in the request, the request further comprising a group member identifier, storing of the group member in the created group;
- Upon reception of a request to access data of a group to access, the request comprising a group identifier of the group to access and comprising an identifier identifying a requester of the request to access data, giving access to the data of the group to access if the identifier identifying the requester is stored in the group to access.

2. Method according to claim 1, characterized in that the cryptographical hashing function implemented through a Distributed Hash Table.

3. Method according to any of claims 1 or 2, characterized in that the group member identifier is a group identifier.

4. Method according to any of claims 1 to 3, characterized in that the request to create a group, the request to insert a member in the group and the request to access data of a group to access are being made accessible via an application programming interface.

5. Communicating device (13) for managing groups of data and group members, the device being characterized in that it is connected with other communicating devices connected in a communication network, the communicating device using a cryptographical hashing function to determine a first identifier indicating at least one of the communicating devices for storing of data of at least one of the group of data on one of the communicating devices, the method comprising the following steps:

- means for reception of a request to create a group by a group creator, the request comprising the group identifier and a second identifier representative of the group creator, storing of data of the group on the communication device that is determined according to the function;
- means for reception of a request from the group creator to insert a member in a group created by the group creator, the creator being identified by a the second identifier comprised in the request, the request further comprising a group member identifier, storing of the group member in the created group;
- means for reception of a request to access data of a group to access, the request comprising a group identifier of the group to access and comprising an identifier identifying a requester of the request to access data, giving access to the data of the group to access if the identifier identifying the requester is stored in the group to access.

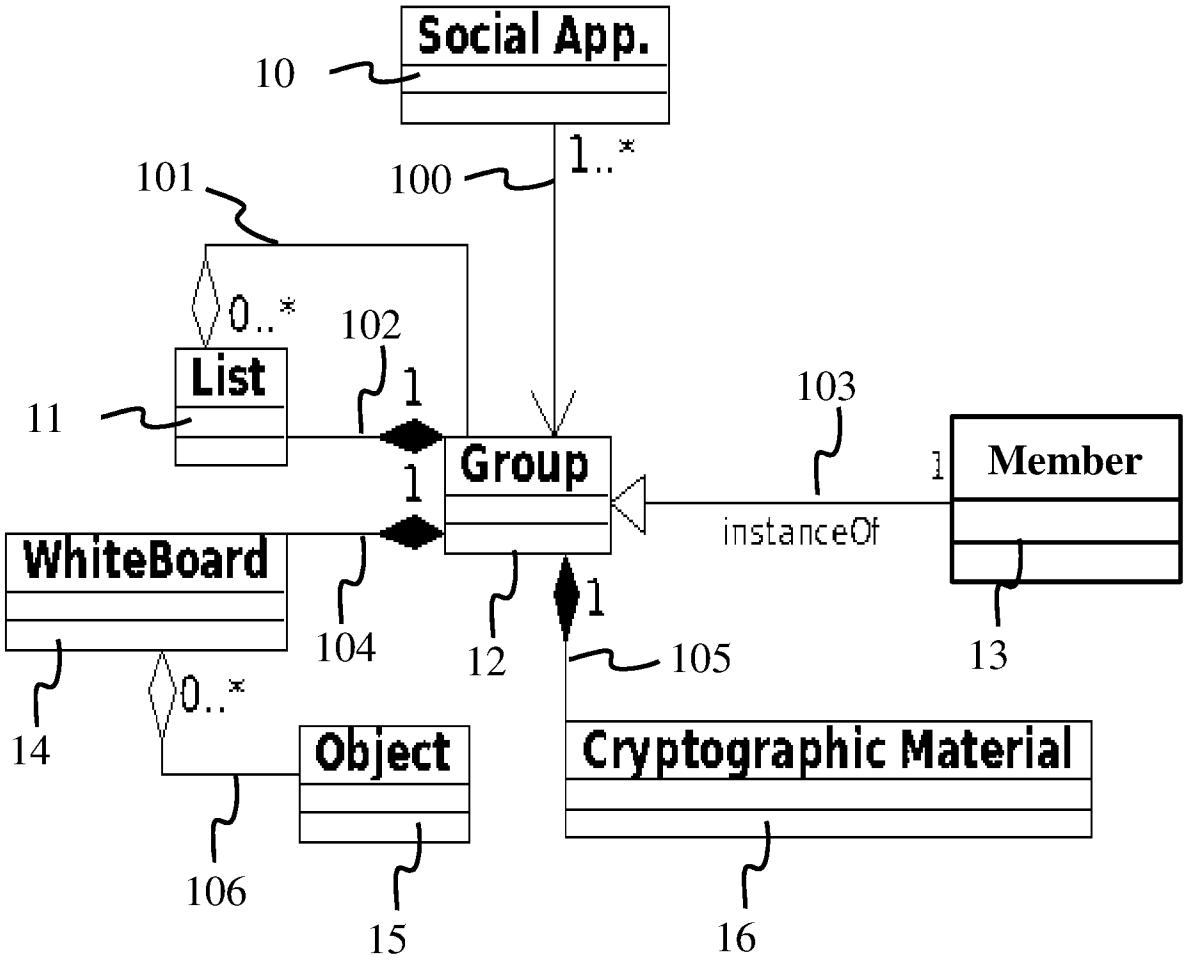
Social App.
10
1..*
100
101
0..*
102
List
11
1
1
Group
103
1
Member
instanceOf
WhiteBoard
104
12
1
105
13
0..*
14
Object
Cryptographic Material
106
15
16

Fig. 1

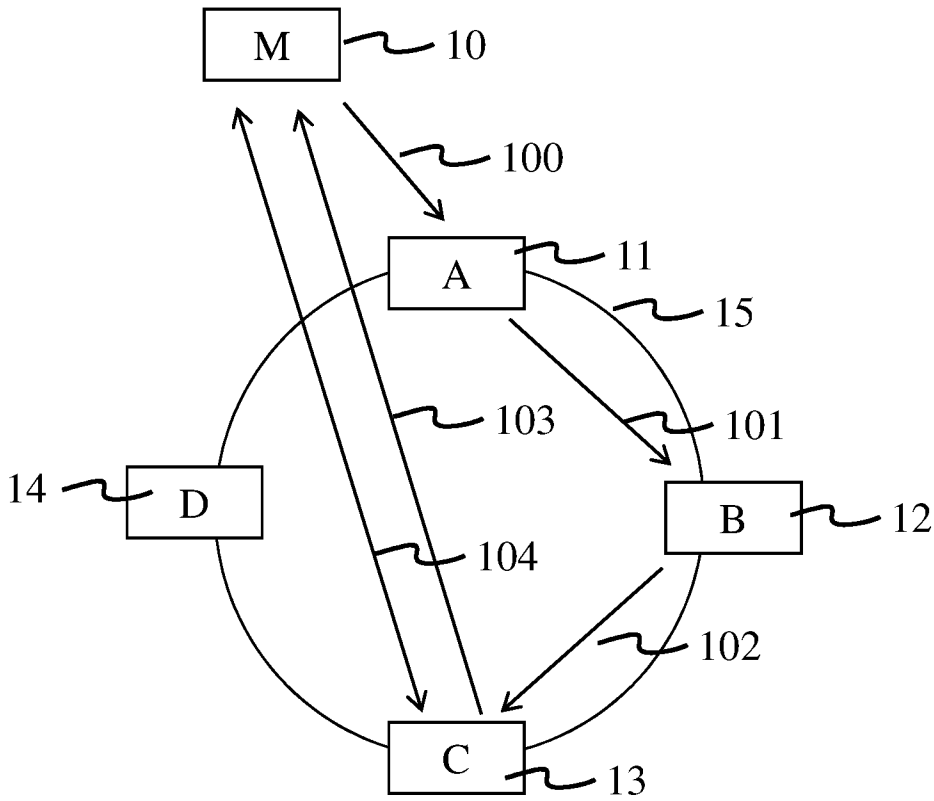

Fig. 2

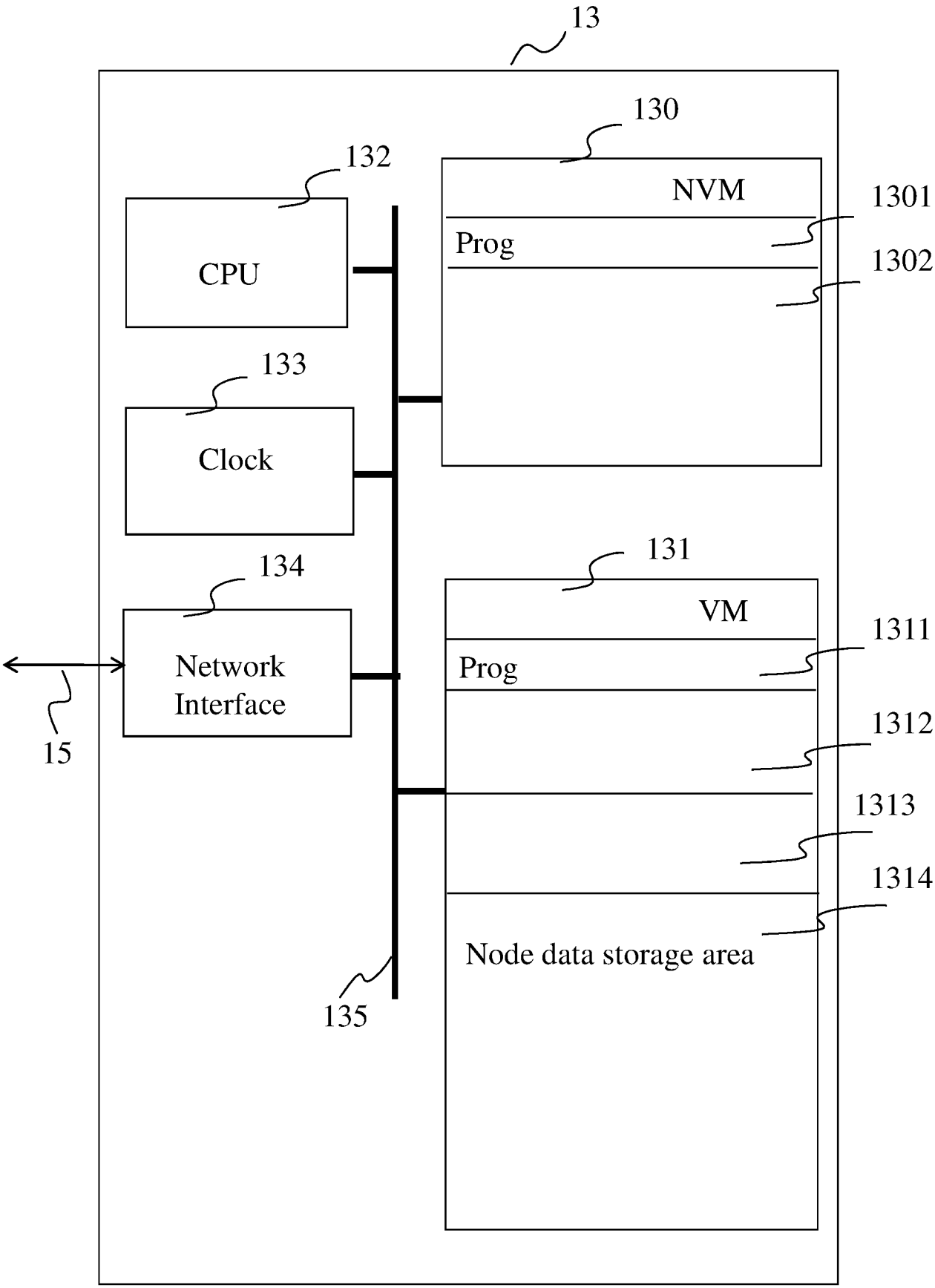
13
130
132
NVM
1301
Prog
1302
CPU
133
Clock
131
134
VM
1311
Network
Interface
Prog
1312
1313
1314
Node data storage area
15
135

Fig. 3

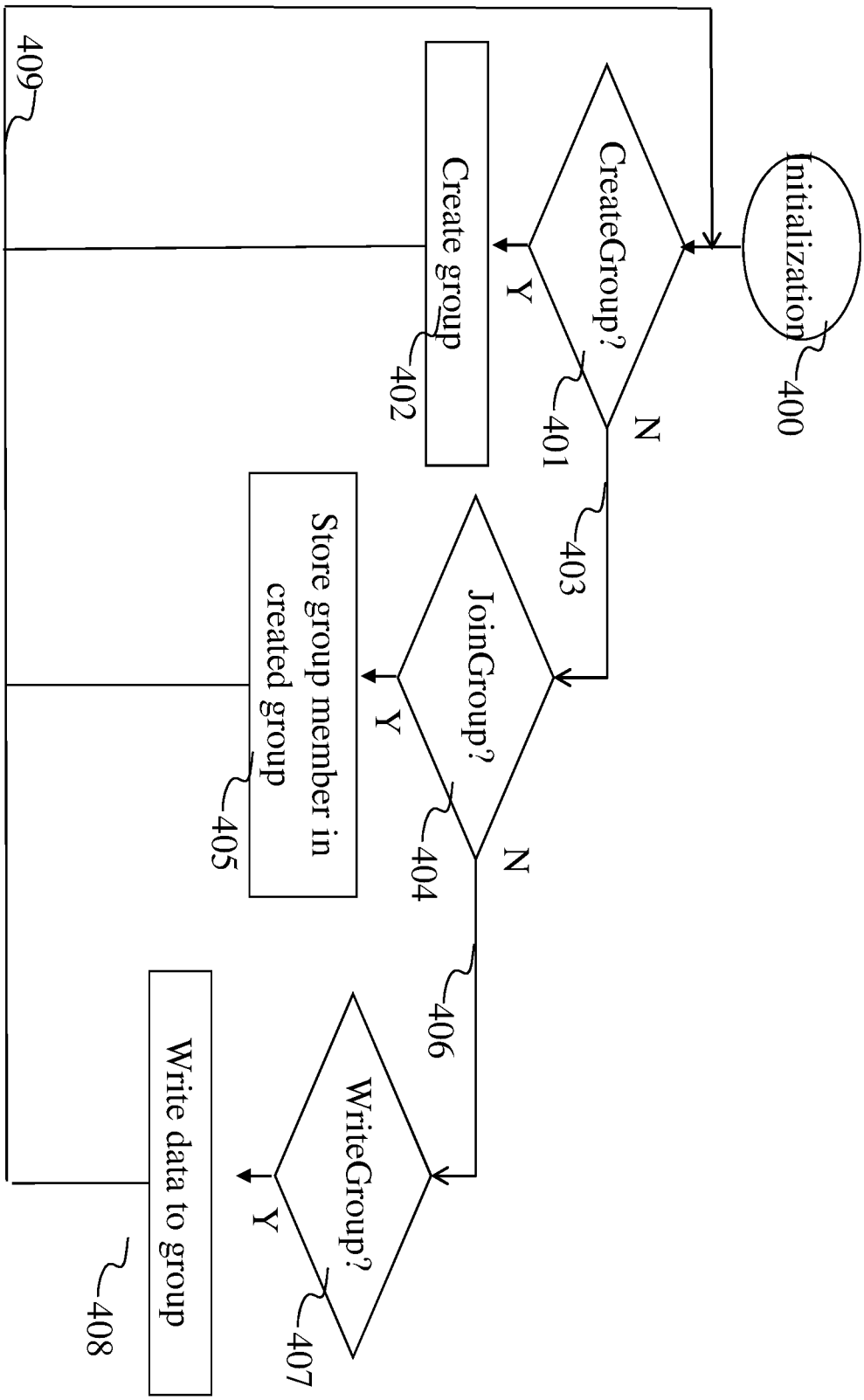
Initialization
400
CreateGroup?
401
Y
N
Create group
402
JoinGroup?
403
Y
N
Store group member in created group
405
WriteGroup?
404
406
Y
Write data to group
407
408
409

Fig. 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2012/054537

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06Q10/00 H04L29/06
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)
G06Q H04L

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	MIGUEL CASTRO ET AL: "Scribe: A Large-Scale and Decentralized Application-Level Multicast Infrastructure", IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE SERVICE CENTER, PISCATAWAY, US, vol. 20, no. 8, 1 October 2002 (2002-10-01), XP011065540, ISSN: 0733-8716 the whole document	1-5

-/--

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

[ ] 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
13 June 2012	27/06/2012

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	Neppel, Clara

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2012/054537

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	KUN-YI CHENG ET AL: "Peeraid: A resilient path-aware storage system for open clouds", 14 January 2009 (2009-01-14), SERVICE-ORIENTED COMPUTING AND APPLICATIONS (SOCA), 2009 IEEE INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, PAGE(S) 1 - 8, XP031625981, ISBN: 978-1-4244-5300-9 the whole document	1-5
X	FELBER P ET AL: "SPADS: Publisher Anonymization for DHT Storage", 25 August 2010 (2010-08-25), PEER-TO-PEER COMPUTING (P2P), 2010 IEEE TENTH INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, PAGE(S) 1 - 10, XP031752236, ISBN: 978-1-4244-7140-9 the whole document	1-5