International Journal of Internet and Distributed Systems, 2014, 2, 5-14
Published Online May 2014 in SciRes. http://www.scirp.org/journal/ijids
http://dx.doi.org/10.4236/ijids.2014.22002

Scientific Research

Reliable Content Distribution in P2P Networks Based on Peer Groups

Elias P. Duarte Jr., Ana Flavia B. Godoi

Department of Informatics, Federal University of Parana (UFPR), Curitiba, Brazil
Email: anaflavia@inf.ufpr.br, elias@inf.ufpr.br

Received 20 February 2014; revised 25 March 2014; accepted 2 April 2014

Abstract

Peer-to-Peer (P2P) networks are highly dynamic systems which are very popular for content distribution in the Internet. A single peer remains in the system for an unpredictable amount of time, and the rate in which peers enter and leave the system, *i.e.* the churn, is often high. A user that is obtaining content from a selected peer is frequently informed that particular peer is not available anymore, and is asked to select another peer, or will have another peer assigned, often without enough checks to confirm that the content provided by the new peer presents the same quality of the previous peer. In this work we present a strategy based on group communication for transparent and robust content access in P2P networks. Instead of accessing a single peer for obtaining the desired content, a user request is received and processed by a group of peers. This group of peers, called PCG (Peer Content Group) provides reliable content access in sense that even as members of the group crash or leave the system, users continue to receive the content if at least one group member remains fault-free. Each PCG member is capable of independently serving the request. A PCG is transparent to the user, as the group interface is identical to the interface provided by a single peer. A group member is elected to serve each request. A fault monitoring component allows the detection of member crashes. If the peer is serving request crashes, another group member is elected to continue providing the service. The PCG and a P2P file sharing applications were implemented in the JXTA platform. Evaluation results are presented showing the latency of group operations and system components.

Keywords

Content Distribution, Peer-to-Peer Networks, Group Communication

1. Introduction

P2P (peer-to-peer) networks and applications are very popular content distribution systems [1]. Typical P2P ap-

How to cite this paper: Duarte Jr., E.P. and Godoi, A.F.B. (2014) Reliable Content Distribution in P2P Networks Based on Peer Groups. *International Journal of Internet and Distributed Systems*, **2**, 5-14.
http://dx.doi.org/10.4236/ijids.2014.22002

plications go beyond content sharing, and include distributed computing and collaboration [2]. A peer is a process that can serve requests and, at the same time, can issue requests to other peers [3]-[5]. A P2P system is usually highly dynamic, in the sense that the set of peers of a system varies with time, as they continuously enter and leave the system. The rate in which the system composition varies is called the system *churn*. Because of their very nature, P2P applications present potentially good intrinsic properties such as availability, user and resource anonymity, and scalability. On the other hand, their dynamic nature creates challenges for effectively deploying traditional distributed abstractions, such as distributed agreement [6].

Because of their intrinsic redundancy, P2P systems can naturally lead to high availability. Nevertheless, most current systems do not offer continuous service and data in the presence of failures and reconfigurations. In this way, interruptions and abrupt variations of performance are frequent. In fact, in most systems it is common to have the client or even the end-user to be responsible for selecting the specific peer from which service or content is obtained. If the peer that is providing the content fails or leaves the system, the client peer not only stops receiving the content but also usually has to select another peer to continue receiving the content. In general, there is no verification of whether the quality of the new serving peer content has not changed.

In this paper we propose a new approach to build robust P2P systems which increases the availability of content provision, providing transparent recovery after serving peers fail or leave the system. The main goal of the proposed system is to place the responsibility of keeping the quality of service on the P2P infrastructure itself, being transparent to the end-user. The proposed approach is based on the fact that most content is naturally replicated in a P2P system. Group communication [7] is a useful approach that can be employed to support consistent replication in distributed systems. A group consists of a set of processes that share a common state and cooperate to execute tasks. From the point of view of the users, the group represents a single logical entity. The group communication service guarantees the availability and the consistency of the distributed application that relies on the group, even after faults occur. A key component of the group service is the module responsible for keeping a view of which members are in the group.

In this work we introduce a novel abstraction called Peer Content Groups (PCG) to deploy transparent content delivery despite of churn, based on a group service for P2P systems. All members of a PCG maintain the same content. A PCG provides reliable content access if at least one group member remains fault-free. The group interface is transparent, identical to the interface provided by a single peer. A group member is elected to serve each request. A fault detection component allows the detection of member crashes. If the peer that is serving a request crashes or leaves the system, another group member is elected to continue providing the service. Peer election and replacement are transparent to client peers. Both the PCG and a P2P file sharing application were implemented in the JXTA platform [8]. This application allows a file to be continuously and transparently transferred from the P2P system even when the serving peers crash or leave the system. Case studies are presented showing representative values for the latency of several system components. To the best of our knowledge this is the first strategy proposed to create and maintain transparent peer groups for continuous content delivery in P2P networks.

The rest of this paper is organized as follows. Section 2 presents the Peer Content Groups, describing the PCG architecture, functionality and algorithms. Section 3 describes the JXTA resources employed to implement the PCG and the file-sharing application. Section 4 presents related work. Finally, Section 5 concludes the paper.

2. Peer Content Groups

A *Peer Content Group* (PCG) consists of a set of peers all of which maintain a specific content, each can provide this content to other client peers, and maintain information on which peers are currently in group (group composition). We employ the terms peer, node and process interchangeably and with the same meaning in this work. The network topology can be represented by a complete graph, *i.e.* each peer can communicate directly with any other peer. We consider an asynchronous system, *i.e.* bounds on the time required for the execution of an action by a process or for a channel to deliver a message are unknown [6]. Peers fail by crashing, *i.e.* a faulty process ceases all activities. However a peer can also simply leave the system, and in this case the effect on the group is the same as if it had crashed. A peer that fails can later recover, but it does not maintain any state information about its previous computations. In the same way, a process that left the system explicitly, can later rejoin the system again. Peers communicate through reliable channels: if a process p sends a message m to process q and assuming that process q does not fail, then q receives m.

The PCG service allows member peers to communicate among themselves and also provides the group interface for peers outside the group to interact with the group. The group membership component is responsible for detecting faulty members, and for keeping an up-to-date list of peers that are fault-free, called the group view [7] [9] [10]. Furthermore there are also mechanisms for members to join and to leave a PCG. The next subsection presents the protocols and functionalities of this component, shown as Algorithm 1.

Algorithm 1: PCG Composition Management.

```
/* Variables: */
ListOfPeers agreementGroupView <- empty;
ListOfPeers localGroupView <- empty;
List of ListOfPeers lViews <- empty;
Join(GroupId):
/* peer i executes this function to join group GroupId */
search for group GroupId;
if group is found
then send join message to the members
else create group with GroupId;
ReceiveMessageFromGroup(msg):
/* peer i receives a message from peer j */
Case msg is:
    a join message:
        send reply with localGroupView to peer j;
        insert peer j in the localGroupView;
    a reply to the first join message:
        insert peer j in the localGroupView;
        store peer j's group view in lViews;
    a heartbeat message from peer j :
        insert peer j in localGroupView;
        store peer j's group view in lViews;
UpdateView():
/* Executed by peer i at each heartbeat interval */
send heartbeat message with localGroupView to group members;
if peer i has not received a heartbeat from peer j;
then remove peer j from localGroupView
agreementGroupView = peers in all views of lViews;
```

2.1. Joining/Creating a Peer Content Group

The procedure executed by a peer to join a PCG starts when the peer sends an initial request to become a group member. The request carries the PCG identifier, which must be related to the content provided (id's are detailed later in the context of the implementation). If no reply is received, the peer creates the PCG, as it concludes the group does not exist. Otherwise, the peer updates its local view with the identifiers of all peers from which it received replies to the original request. We assume every member correctly sends a reply to a join request. Furthermore every peer *i* that received the join request from peer *j*, inserts the peer *j*'s identifier in its local view and sends an acknowledgement to peer *j*.

2.2. Updating the Peer Content Group View

The group management service allows peers to dynamically maintain the PCG view. Every time a message is received by a given peer, its local PCG view is updated with information about the peer that sent the message. All messages exchanged among peers, except the initial join message, carry the local view of the peer that sent the message. All peers of the PCG keep the views of the other members of the group. These views are used to construct an agreement group view, which is described below. An agreement group view consists of a list of all fault-free members of the PCG, ordered lexicographically.

A *heartbeat* is sent by every peer to other PCG members periodically, at a *heartbeat interval*. Heartbeats are used by a peer as a means to inform all group members it is alive. Heartbeats also carry the group view. A peer also sends its local group view to all group members every time its local view changes by the detection of an event, which is either a new peer that entered the group or one that has left or crashed.

2.3. Local Group View & Agreement View

Each peer keeps a local group view, which is a set of peer identifiers. As the system is asynchronous and dynamic, and peers join and leave the PCG continuously, each peer may have a local group view different from the views kept by other members. Nevertheless the PCG as a whole can offer dependable services if members share a unique agreement view. A peer builds the agreement view from the local views received from other peers in the group, as follows. Peer *j* is in the group view of peer *i* if peer *j* is in *all* local views maintained and received by peer *i*. In other words, only peers that are not considered to be faulty by all members are kept in the agreement view. Thus the agreement view is an intersection of all local views. It must be noted that, as the system is highly dynamic, the agreement view presents weak guarantees of consistency.

Each peer periodically checks whether it received heartbeats from the peers that are in its local view. In case peer *i* does not receive heartbeats for long enough from some peer *j*, peer *i* removes this peer *j* from its local view. As the system is asynchronous, a slow peer can be considered to be faulty by some member, and may be eventually removed from the agreement view. When the slow peer receives a local view in which it is not listed, it executes the initialization procedure, *i.e.* it leaves and joins the group again.

Observe that the protocol presented above allows the existence of several distinct agreement views at some point, again because the system is highly dynamic. However, if for a large enough interval of time the system remains stable (without members joining/leaving the group) and if there are no undue suspects, the proposed protocol allows that the several local group views to converge to a unique agreement view. We assume that the systems goes through stable periods frequently enough. During these periods the agreement view is unique and lists the actual fault-free members in the group.

An API (Application Programmer Interface) is also defined to the group management service, which allows applications to be constructed with the proposed system. The interface allows messages to be sent and received to/from an individual peer or to the group as a whole, and also provides means to obtain PCG composition information.

2.4. Peer Election

A distributed leader election protocol is executed to select the peer that will serve a request. The election is executed after a client peer sends a request to the PCG. This request is received by all members but only one member replies. All members maintain a list with all pending requests. This list also contains information about which peer is serving each request. When a new request arrives, PCG members must decide which peer will serve the request, and inserts the information in the list.

A server peer is elected based on the agreement group view and on the number of the requests received. Each request received is assigned an identifier, which is the same at all peers. The request identifier is the number of requests received by the file group plus one. The id of the last request received is exchanged by all PCG members in the heartbeat message. Each peer computes locally the remainder of the division of the request id by the amount of peers in the agreement view of the file group. The result identifies the peer of the agreement view elected to serve the request. When a peer decides it is a server of some request, this peer starts to send of the file to the client peer.

In case this server peer becomes faulty or leaves the system, the group elects a new server to replace the previous one. Each peer of the file group determines the new server, based on the agreement view and the request id. The new server starts to the send the file from the point the download had stopped. After a request has been served, all group members remove the request id from their lists of pending requests.

3. A JXTA-Based File Sharing Application

The proposed Peer Content Group service was implemented in the JXTA platform [8] version 2.5, using Java language. Besides the PCG service itself, an example file sharing application was implemented, called *JXTA Content Group*.

3.1. JXTA Implementation

Figure 1 shows how the application and group service fit within the JXTA architecture. The JXTA network overlay, as well as several JXTA components and services were used to implement and run both the group service and the application. The JXTA core provides essential services, including basic operations such as the creation of peers with unique identifiers. The JXTA service layer provides non-essential but nevertheless important services, for example: peer authentication and content indexing. Another key JXTA component is the *pipe*, which corresponds to a communication channel between peers. The native JXTA peer groups were also used, as well as JXTA advertisements, employed to publish and find resources in the JXTA network.

Although JXTA does have a "peer group" primitive, which was employed in the implementation of the proposed PCG, it is important to make clear that these groups are different from each other. Every JXTA peer takes part of some JXTA peer group. When a peer is first created and starts running, it is already part of the main JXTA *World Peer Group*. From then on a peer can create new groups, or join other existing JXTA groups. The proposed PCG or its JXTA implementation *JXTA Content Group* is actually an application group, which peers can join and leave, depending on the content they have available and are willing to share. The PCG structure is described in the next subsection.

Peers communicate through communication channels called pipes. In order to create a pipe, a peer needs information about the identification of the pipe of the other peer with which communication is desired. This information is obtained from JXTA advertisements. Each resource of the JXTA P2P network is described by an advertisement. In order to share a resource, a peer that has that resource needs to create an advertisement for that resource and publish this advertisement on the JXTA network. Other peers can then search and locate the resource.

The publication and search of advertisements in the JXTA network are executed with the help of super-peers, called rendezvous. A rendezvous keeps indexes that allow the location of resources across the network. When a peer publishes an advertisement, the rendezvous keeps information about both the advertisement and the identification of the advertised peer. Each peer needs to be connected to at least one rendezvous. To start a search, one peer sends a request to the rendezvous to which it is connected and waits for responses. The rendezvous checks local indexes and may propagate the search request to its neighbours. If the resource is found, the rendezvous sends a reply, with information about the requested resource.

All searches are deployed in the context of the JXTA groups. This means that if a peer executes a search inside a JXTA group to which the peer belongs, only resources published in this JXTA group will be returned.

3.2. A PCG File Sharing Application

In this section we describe an example application based on the proposed PCG. The application allows peers that keep a given file to form a PCG. Not only can a client can download the file reliably and transparently from the group, but also the group interface is identical to a single peer interface, thus the group is transparent to the client. Changes in the group composition are also transparent to clients, service is automatically recovered by the group if peers join/leave the group or fail/recover. The group is capable of recovering from service interruptions, providing continuous download, even if only one peer is available.

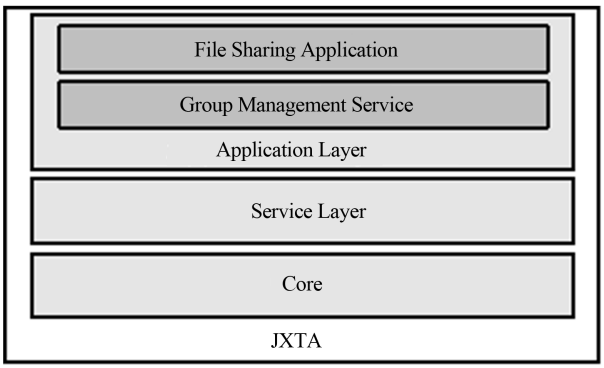

Figure 1. PCG composition management.

3.2.1. JXTA Content Groups in the JXTA Group Hierarchy

Peers are grouped hierarchically in a JXTA network. The main JXTA group is the *World Peer Group*, as shown in **Figure 2**. All other JXTA groups are created from this one. When a peer initiates the execution of JXTA protocols, it automatically becomes part of the main group. If the peer wants to access the content sharing application, it issues a search for the application group and then joins this group. At the lower layer, file groups are defined within the application group, three examples of which are shown in **Figure 2**: groups A, B and C. A file group is composed of peers that maintain a file they are willing to share. Each file group is responsible for keeping and making just one file available to other peers, peers can take part of as many file groups as they wish to, according to the number of files they are willing do share. The application groups are also JXTA groups. All members of these groups execute the content sharing application.

Group management procedures are executed in order to maintain the file group. Maintenance tasks include continuously monitoring and updating the group composition, taking account of peers that join and leave the group, either intentionally or because of faults. Another task is the election of peers that will serve client requests. Request management is also another group task. After a peer initializes, it executes a search for the application group, hierarchically below the JXTA main group. In case the peer does not find the group, it creates one. Otherwise the peer joins the group, becoming a member of the application group. The next steps depend on whether the peer wishes to share files (server peer) or to download files (client peer). Both cases are described in the subsections below. If a client peer wishes to obtain a file, this peer searches for the corresponding file group and requests the file. If a server peer wishes to share a file, it searches for the corresponding group and joins that group, or creates a new one in case the group does not exist.

The identifier of a file group is the *hash* of the shared file. The *id* of each group is the hash. A server peer that wishes to share a file computes the hash and searches for a group with *id* equal to that hash. A client peer searches for file groups using the group description in the corresponding JXTA advertisement. The PCG description contains the file name-not the hash. In this way, a client can find more than one file group with the same file name, these groups keep different versions of the file, the hashes are thus different.

3.2.2. Server Peer

The server peers have files to share with other peers. After they connect to the application group, the server peers join the file group of the corresponding file they desire to offer. If the group does not exist, the first peer creates and publishes the group advertisement.

The peers that are members of file groups execute the group management service which includes the group management tasks and the leader election protocol that designates a peer to serve each download request received by the group. For each file group of which the peer is a member, it executes one instance of the group management service.

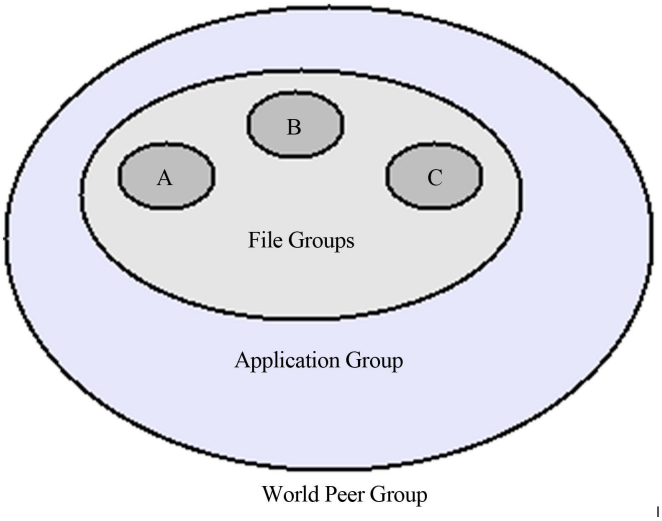

Figure 2. The JXTA group hierarchy.

The file groups provide the interface through which other peers can get the file. Client peers use this interface to reply to download requests. When a group receives a request, a member is elected to send the file to the client. After the election, the peer designated to serve the request, *i.e.* the server peer, starts the transmission of the file to the client. In case this server becomes faulty or leaves the group before the transmission is complete, the other members elect a replacement after the group composition is updated. After the transmission is finished, the group members receive a confirmation message sent by the client, and remove the request id from the pending requests list.

3.2.3. Client Peer

Client peers connect to the application group to search for available file groups. These peers are only in the application group, and use the download functionality of content groups. For each file that a client desires, the client must search for a peer or group from which the file can be obtained. The group interface is identical to a single peer interface, thus a search may return both. If the client finds the file group, a request is sent to the group. The request has the advertisement of the client's communication channel, which the server has to use to send the file to the client. After sending the request, the client creates a communication channel and waits to receive the file from the group. After the file is completely received, the client closes the channel and sends a confirmation message to the group.

3.3. Evaluation

This section shows experimental results obtained from the execution of a case study described below. The environment consisted of five hosts based on Pentium 2.8 GHz processors, running Ubuntu Linux, connected on a 100 Mbps Ethernet. Eight server peers were executed on four hosts, *i.e.* each host executed two server instances. The fifth host executed both the rendezvous peer and the client. Client peers were employed to allow the evaluation of the leader election module.

A single file was used in the measurements, *i.e.* the eight server peers all belong to the same PCG. The clients requested downloads of this file. The group management module was configured so that peers would send heartbeat/local view messages periodically in a 10 seconds heartbeat interval. Experiments were repeated several times and results presented below are representative of the set of results obtained.

In order to make the execution of more than one peer on a single host possible, each peer used a different communication port. Furthermore each peer must be executed from a different directory. In order to simulate a realistic P2P scenario, a random artificial delay was inserted as each message was transmitted. Without this delay all messages would be delivered within less than 1 millisecond in our system. The inserted delay varied from zero to a hundred milliseconds, uniformly distributed on this interval.

3.3.1. Latency of the Group Communication Service

This section describes the latency measured for a peer to join and also leave the group. In order to measure this latency the eight peers were initialized one by one. The procedure we adopted was to allow one peer to join the group after the other peers had successfully been detected by all group members. The latency consisted of the time elapsed between the initial join request message was sent until the peer appeared in the local views of all group members. We then computed the average of all measured latencies. The experiment can be roughly divided in two parts: in the first part, one peer was executed per host until four hosts were running peers. After this part, a second peer was started on each host. Results are shown in **Figure 3**. The graph shows the number of members in the group as a new joining peer is detected.

Figure 3 shows the average latency for all group members to detect that a new member has joined the group. The x-axis corresponds to the number of peers already in the group. It is clear that the latency increases significantly when the fifth peer joins the group—this corresponds to the second part of the experiment, as described above, when more than one peer is instantiated per host. The average latency for all members to detect a new member was 11.8 seconds. It is important to keep in mind that the group management service depends on the heartbeat interval to determine any change in the group composition. In the case study this interval was set to 10 seconds. It is thus possible to state that the latency to detect a new member is, in average, 1.2 times the heartbeat interval.

The latency for detecting that a member has become faulty or has left the group was also measured by making members leave the group one by one. Each member was removed from the group after the remaining peers had

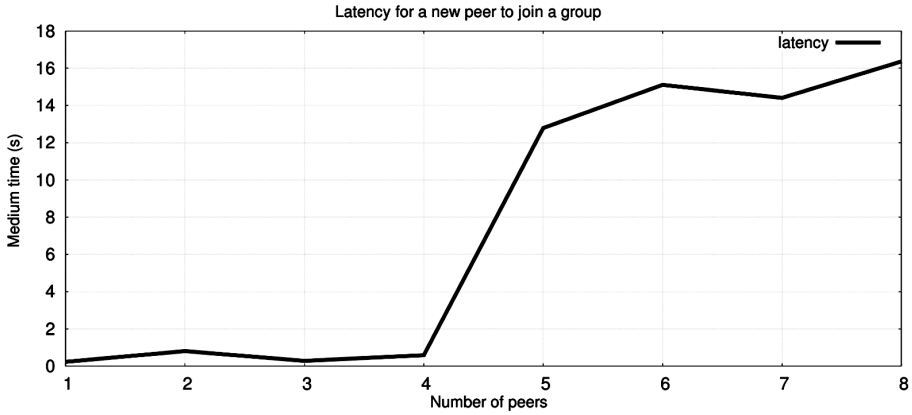

Figure 3. Latency for peers to join a group.

already detected all the members that had previously left the group. The impact of a peer fault on other modules, such as the leader election protocol, was also measured. The graph in **Figure 4** shows the average latency obtained as the number of group members varied.

It is possible to conclude that the latency to detect that a member became faulty/left the group increases slowly as the number of members in the group increases. The latency was, in most cases, between 11 and 13.5 seconds, with an average of 12.8 seconds, or 1.3 heartbeat intervals.

3.3.2. Latency to the Start Downloading a File

The leader election module is used to designate a group member to serve a download request. The time it takes to elect a leader starts when the request is received and completes when the file starts to be transmitted. This latency was also measured in the presence of faults, *i.e.* the elected peer becomes faulty and another member is elected to replace the faulty server. This latency was measured for a varying number of peers in the group. The graph in **Figure 5** shows the results we obtained. It is clear from the figure that the latency increases when the server peer becomes faulty and is replaced. In this experiment, the latency varied between 7 and 13 seconds, with an average of 11 seconds, which corresponds to 1.1 heartbeat intervals.

4. Related Work

The implementation of reliable distributed algorithms in P2P systems is still a research challenge. Due to their dynamic nature, *i.e.* processes enter and leave the system continuously, it is not possible to implement solutions developed for traditional static systems [6]. In particular, reaching agreement in a dynamic distributed system is still an open problem. In this section we make a review of related research efforts. We start mentioning that this work is a extended version of our previous conference paper [11].

SWIM (Scalable Weakly-consistent Infection-style Process Group Membership Protocol) [12] proposes process groups that have weakly-consistent membership knowledge. SWIM is based on a randomized probing protocol, being thus probabilistic. The system employs gossip for disseminating failure detection and membership information. As the size of content-based groups grows, adopting a probabilistic strategy may be an option for guaranteeing scalability. SCAMP (Scalable Membership Protocol) [13] is another probabilistic peer-to-peer membership protocol which is based on partial views which support a gossip style of dissemination.

The approach proposed in [14] relies on a super-peer overlay which is built based on the semantic similarity of content provided. The main purpose is to use the overlay to reduce the cost of queries in unstructured P2P networks. Another related work is the strategy proposed in [15] for deploying fault-tolerant super-peers in structured P2P networks; the proposed strategy is also evaluated with a file sharing application but peers are organized in hierarchical groups that rely on a top-level super-peer. The proposed approach is to make each regular peer logically connect with two or more super peers in other groups so that the failure of a super-peer does not disrupt the provided service.

The evaluation of group membership under churn is reported in [16]. A related problem is reaching agreement in mobile ad hoc networks. In [17] group members are at first unknown, so that before any membership or

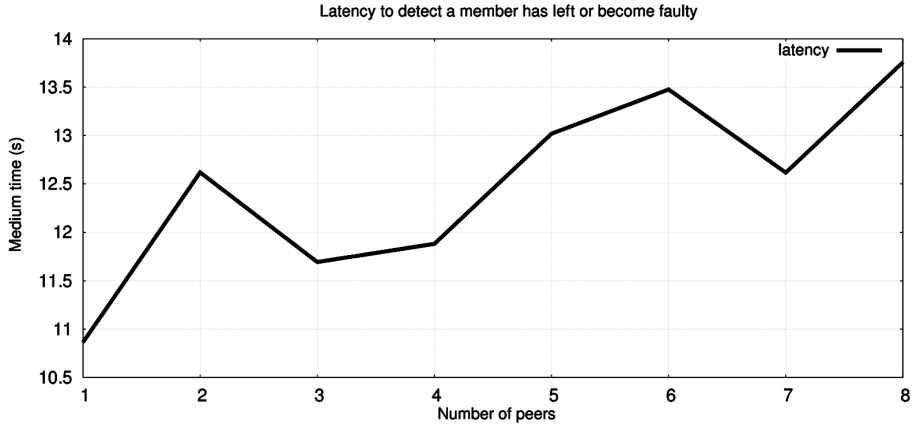

Figure 4. Latency to detect that a member has left or become faulty.

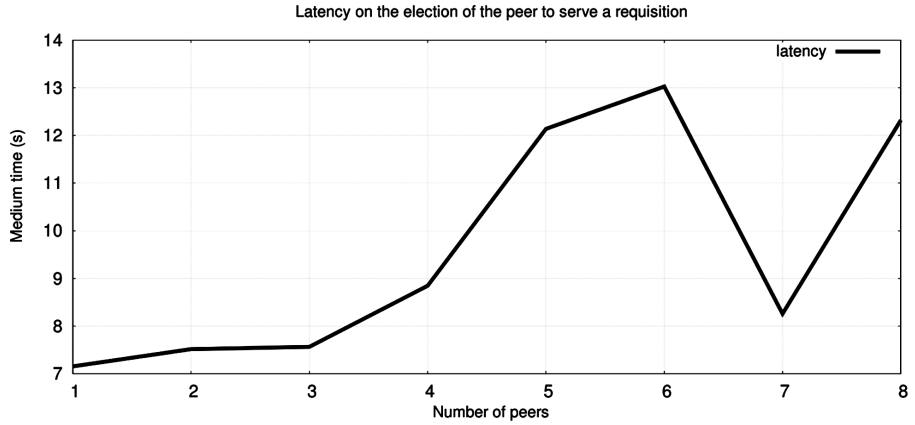

Figure 5. Peer election latency.

failure detection protocol is executed there is a phase in which participants themselves are identified. The authors show basic conditions on participant knowledge and system synchrony required to achieve consensus in such a system.

In [18] the authors propose a multi-layered hierarchical fully distributed group model. Groups support causal delivery of messages, which is achieved by having each peer to hold information on the accuracy of physical clock of all peers and minimum delay time among every pair of peers. The authors evaluate the hierarchical model in terms of the group information size and delay time compared with a flat group model.

Finally, the recent survey [19] gives an overview of different strategies for defining groups in P2P systems, this includes groups formed by user interest or technical criteria such as proximity. The authors present an evalation that does take robustness in consideration, besides other criteria such as suitability and security.

5. Conclusions

The Peer Content Groups (PCGs) defined in this work allow reliable and transparent content access in JXTA-based P2P networks. PCG is based on content-based group abstraction, which allows peers that maintain and are willing to share a given content to cooperate to serve client requests. This abstraction consists of a group management service and a distributed leader election module that designates the server for each request. If this server fails or leaves the system, the group elects a replacement. The service is transparent in the sense that the group interface is identical to a single peer interface, and furthermore server replacement in case of faults is not visible to the client. An example file sharing application was implemented and is described. Evaluation results are shown for the latency for detecting a new joining member, faulty members and the time to download a file.

Future work includes defining a reputation scheme for groups, and defining an approach for distributing load among group members.

References

[1] Theotokis, S.A. and Spinellis, D. (2004) A Survey of Peer-to-Peer Content Distribution Technologies. *ACM Computing Surveys*, **36**, 335-371. http://dx.doi.org/10.1145/1041680.1041681

[2] Zhang, Q., Cheng, L. and Boutaba, R. (2010) Cloud Computing: State-of-the-Art and Research Challenges. *Journal of Internet Services and Applications*, **1**, 7-18. http://dx.doi.org/10.1007/s13174-010-0007-6

[3] Lua, E.K., Crowcroft, J., Pias, M., Sharma, R. and Lim, S. (2005) A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. *IEEE Communications Surveys and Tutorials*, **7**, 986-988.

[4] Clark, D. (2001) Face-to-Face with Peer-to-Peer Computing. *IEEE Computer*, **34**, 18-21.

[5] Schollmeier, R. (2001) A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. *The* 1*st IEEE International Conference on Peer-to-Peer Computing* (*P2P*), August 2001, 101-102.

[6] Charron-Bost, B., Pedone, F. and Schipper, A., Eds. (2010) Replication: Theory and Practice. Springer, Berlin. http://dx.doi.org/10.1007/978-3-642-11294-2

[7] Chockler, G.V., Keidar, I. and Vitenberg, R. (2001) Group Communication Specifications: A Comprehensive Study. *ACM Computing Surveys*, **33**, 427-469. http://dx.doi.org/10.1145/503112.503113

[8] JXTA (2014) The Language and Platform Independent Protocol for P2P Networking. http://jxta.kenai.com

[9] Bünzli, D., Fuzzati, R., Mena, S., Nestmann, U., Rutti, O., Schiper, A. and Wojciechowski, P.T. (2006) Advances in the Design and Implementation of Group Communication Middleware. In: *Dependable Systems*, 172-194.

[10] JGroups (2014) http://www.jgroups.org

[11] Godoi, A.F.B. and Duarte Jr., E.P. (2011) Peer Content Groups for Reliable and Transparent Content Access in P2P Networks. *The* 5*th International Workshop on Peer-to-Peer Network Virtual Environments*, *The* 17*th IEEE International Conference on Parallel and Distributed Systems* (*ICPADS*), 776-781.

[12] Das, A., Gupta, I. and Motivala, A. (2002) SWIM: Scalable Weakly-Consistent Infection-Style Process Group Membership Protocol. *The* 32*nd IEEE/IFIP International Conference on Dependable Systems and Networks* (*DSN*), 303-312.

[13] Ganesh, A., Kermarrec, A.-M. and Massoulie, L. (2003) Peer-to-Peer Membership Management for Gossip-Based Protocols. *IEEE Transactions on Computers*, **52**, 139-149. http://dx.doi.org/10.1109/TC.2003.1176982

[14] Garbacki, P., Epema, D. and van Steen, M. (2007) Optimizing Peer Relationships in a Super-Peer Network. *The* 27*th IEEE International Conference on Distributed Computing Systems* (*ICDCS*), Nuremberg, 25-27 June 2007, 31.

[15] Lin, J.-W., Yang, M.-F. and Tsai, J. (2007) Fault Tolerance for Super Peers of P2P Systems. *The* 13*th IEEE International Symposium on Reliable Distributed Systems* (*SRDS*), Beijing, 17-19 December 2007, 107-114.

[16] Baldoni, R., Mian, A.N., Scipioni, S. and Piergiovanni, S.T. (2005) Churn Resilience of Peer-to-Peer Group Membership: A Performance Analysis. *The* 7*th International Workshop on Distributed Computing* (*IWDC*), *Lecture Notes in Computer Science*, 3741.

[17] Greve, F. and Tixeuil, S. (2007) Knowledge Connectivity vs. Synchrony Requirements for Fault-Tolerant Agreement in Unknown Networks. *The* 37*th IEEE/IFIP International Conference on Dependable Systems and Networks* (*DSN*), Edinburgh, 25-28 June 2007, 82-91.

[18] Tsuneizumi, I., Aikebaier, A., Ikeda, M., Enokido, T. and Takizawa, M. (2011) A Multi-Layered Model for Scalable Group Communication in P2P Overlay Networks. *The* 25*th IEEE International Conference on Advanced Information Networking and Applications* (*AINA*), Biopolis, 22-25 March 2011, 324-331.

[19] Koskela, T., Kassinen, O., Harjula, E. and Ylianttila, M. (2013) P2P Group Management Systems: A Conceptual Analysis. *ACM Computing Surveys*, **45**, Article No. 20. http://dx.doi.org/10.1145/2431211.2431219