Peer-to-Peer Beyond File Sharing: Where are P2P Systems Going?

Renato Lo Cigno
DISI, Univ. of Trento
Trento, Italy
locigno@disi.unitn.it

Tommaso Pecorella
DET, Univ. of Firenze
Firenze, Italy
tommaso.pecorella@unifi.it

Matteo Sereno
DI, Univ. of Torino
Torino, Italy
matteo@di.unito.it

Luca Veltri
DII, Univ. of Parma
Parma, Italy
luca.veltri@unipr.it

Abstract

Are P2P systems and applications here to stay? Or are they a bright meteor whose destiny is to disappear soon? In this paper we try to give a positive answer to the first question, highlighting reasons why the P2P paradigm should become an integral part of computing and communication services and not only oddities for Cyber-geeks.

1. Introduction

Peer-to-Peer (P2P) systems have evolved, during the lifetime of *HotP2P*, from queer objects of research to an economic reality with social impacts. The original application of P2P systems was the exchange of files between peer users, instead of clients "buying" files and servers "selling" them: sharing of illegal contents and exchange of copyrighted material made the rest in creating myth and fame for these systems.

From the advent of Skype^{TM}, however, research on P2P systems started to consider P2P more a communication paradigm for improving performances and decreasing costs, rather than a system to circumvent servers' rights on content. A communication paradigm implies a question: What applications can this paradigm support? Or, in different words, what is the evolution of P2P systems beyond enabling of file sharing among a closed community of peers?

Applications exploiting P2P have emerged in many areas. An incomplete list, just to cite a few, includes content distribution, voice and multimedia.

So P2P research and novel applications are healthy and abundant; but, what is the real potential of the P2P approach to become the dominant communication paradigm for the future Internet services? And, most of all, what is the point we're standing in understanding and developing P2P?

The overview of P2P research and systems beyond file sharing applications presented in this paper is based on the work developed in the Profiles[1] project, but is not limited to the project results. It is intended for opening discussions and, why not, spawning new research.

2. Objects Location and Management

Locating application resources or, more generally, objects is fundamental in P2P systems. An "object" is any resource handled by the P2P application: files, the identity/location of a person for chat/e-mail/voice, processing, services, any other. The resulting function is a Location Service (LS) where object identifiers are dynamically mapped onto information that reports the object location, indicating where, when, and how the object can be reached or obtained. If LS is realized in a distributed P2P fashion, it is referred as Distributed Location Service (DLS). A DLS system provides storing and retrieval services on a distributed table that maps keys to values. The table keys represent the name or identifier of a target object, whereas table values are the information useful to locate and reach the object which the corresponding key is associated to. A common way to realize DLS is by using a Distributed Hash Table (DHT). DHTs provide the mapping functionality in a complete distribute, robust, and scalable way. Examples of DHTs in P2P platforms are Chord, Kademlia, Pastry.

Information Abstracting and Representation – If objects and contacts are represented as standard URIs, then the DLS should provide a storage and retrieval service for binding URIs acting as table key, with one or more mapped contact URIs, identifying where and how the resource can be accessed. Together with each contact URI other information can be stored: expiration time, access priority, description, etc.

To implement a general purpose DLS the following basic components must be defined: *(1)* the data representation of DHT keys and values; *(2)* the used DHT algorithm (Kademlia, Chord, etc.); *(3)* the P2P protocol used for maintaining the DHT (inserting a new DHT nodes, updating the routing tables, etc.); and *(4)* the protocol used to perform basic LS queries on the DLS, and used by both DHT-aware and DHT-unaware nodes. DHT keys and values can be respectively the hash value of the URI and a list of object contact addresses. Each contact address should include: *(i)* URI information regarding the actual address where the object can be contacted or reached: IP address, transport protocol, port number, application protocol, properties (for

[1] Profiles (http://profiles.disi.unitn.it) is an Italian PRIN Project funded with grant No. 2006099023 by the Ministry of Education and Research.

978-1-4244-3750-4/09/$25.00 ©2009 IEEE

instance in case of VoIP user agent the contact address could be: sip:192.168.0.77:5060, while in case of a remote file: http://10.0.5.22:8080/filename); *(ii)* the expiration time or validity interval (e.g. expires=3600); *(iii)* a priority value, to be used when more than one contact URIs are obtained; and *(iv)* a text file reporting the object name or description.

For the second component, in general, any DHT algorithm could be used. Regarding components 3 and 4, basically any RPC (Remote Procedure Call) protocol can be used. The protocol selection is driven by considerations on the characteristics of the protocol (standard or not, binary or ASCII, UDP or TCP based, etc), and the ability of working in presence of firewalls or NATs.

Replication and Lifetime – Given the dynamic nature of P2P systems, where each node is not reliable, the management of the objects replicas (for reliability) and existence (to avoid cluttering the system) is of the utmost importance.

The attention received by management of replicas and lifetime was mainly from a perspective of service sustainability in presence of infinite resources. It is clear that, as P2P systems becomes more popular, this is unacceptable. A first example of system dependent strategy for intelligent replication and lifetime management is presented in [1]. The key idea is measuring peers reliability and intelligently updating the information based on it.

In general, a P2P distributed storage should have automatic means for maintaining the stored information with an availability and dependability comparable if not larger than traditional servers. The task is not easy, but reachable thanks to the resilience of the P2P paradigm, that automatically solves the problem of co-location and correlation that affects servers dependability,

3. Resource Finding and Allocation

Nearly every P2P system is built around an implementation of a DHT, i.e., a distributed system able to efficiently find logical objects in a distributed space. One may think that physical resources can be treated similarly. Point is, this assumption is wrong.

DHTs do a perfect job in finding an object; however, resources need more than being found: they must be reserved, allocated and shared.

Modern P2P systems go beyond the classical file sharing, and resources (bandwidth, CPU, memory, storage, etc.) should be shared in an efficient and effective way. This task is also made complex by the user's dynamics, as only in very specific cases the users are willing to share their resources for free. Greedy behaviors have to be taken into account.

Both issues have been addressed in some way. However the techniques implemented are tailored around a specific application, usually not easily applicable to different contexts.

3.1. Overlay Abstraction and Physical Mapping

The 'classical' P2P approach to the network can be roughly summarized as: do not care about the *IP underly*, P2P is an *overlay*! This approach is simple and appealing, gives the system resilience to network changes and a sort of 'invisibility'.

The complete overlay abstraction, however, has major drawbacks that can hinder the efficiency of P2P and, ultimately, hamper its application.

P2P applications can be roughly classified as follows: *(i)* machine to machine (e.g., file sharing), *(ii)* machine to human (e.g., video streaming), *(iii)* human to human (e.g., P2P telephony). Each type has different needs and constraints and should take advantage, in different ways, of the underlying network structure. In simple terms, peers of an overlay can see the network in terms of bandwidth, delay (RTT), and network proximity (two peers belong to the same organization or ISP). Physical distance plays a minor role instead.

Network 'distances' unfortunately do not have topological properties: network distances do not follow the classical triangle inequality.

The discovery of both node distance and of network topology is a complex issue. Many works (e.g., [2], [3]) have addressed this issue, which is however far from being solved.

So far, many P2P systems were built around their DHT structure, often confusing the P2P network with the DHT itself. The DHT is just one component of a P2P system, exactly like the DNS is not the Internet. The optimization of the overlay network can be seen as a special case of overlay routing or topology creation and is strongly dependent on the P2P application type and the users' distribution. Thus, it is not possible to have a solution universally valid for all the possible situations. In any case, it is important to consider the P2P *application*, the DHT and the actual *data distribution* as three different aspects of the P2P network, the latter with specific requirements over the good usage of the underlying network.

3.2. Network vs. Client Driven Cooperation

In P2P systems the performance is affected not only by the protocols and algorithms, but also by peers cooperation. Cooperation does not only mean bandwidth sharing as in BitTorrent or in P2P-TV applications. In a backup system it is (also) disk space, in data processing it is the CPU time, and often it is the combination of different things. Hence, the very first point is to focus on a broader concept of sharing and how it could be reached.

Consider a real-time video distribution system. Bandwidth is needed to distribute the system, but CPU may be a precious resource if some form of network coding is used. If the video is 'soft-live', meaning that it exists in the system for some time, not only in a single stream, then also storage comes into play. Reputation concepts, game theory and Nash equilibria come into play here. Each application should be able to define its own cooperation metrics, thus separating the P2P communication framework from the P2P application level.

This leads to the main issue in cooperation: users are not necessarily willing to cooperate. The ideal behaviour from the user's point of view is to get whatever they can with the minimal effort, i.e., minimal cooperation.

This is called the *Tragedy of the Commons*: All individuals benefit if all act in an altruistic way but each has an incentive to act selfishly. In P2P systems, this problem is evident in many applications.

Classical game theory assumes that each individual acts rationally. An alternative model from evolutionary game theory assumes that individuals will copy the behavior of others who obtain a higher utility. Nodes can selfishly increase their own performance in a greedy and adaptive way by changing their links and strategy. They do this by copying nodes that appear to be performing better and by making randomized changes with low probability.

Evolutionary game theory offers a solution to the tragedy of commons. Assume that to measure performance (e.g., video quality) an appropriate utility U is defined. Hence, each peer i engaged in the application must periodically compute its utility U_i and compare it against another peer j, randomly selected from the population. If $U_i < U_j$, peer i drops its current links to other peers with high probability p_e, copies all the links of peer j, and adds a link to peer j. Further, peer i copies the strategy of peer j. After such a copy, i adapts its strategy (with low probability p_m) and adapts its links (with higher probability γp_m where $\gamma \gg 1$). Adaptation involves applying a mutation operation, and link mutation involves removing each existing link with high probability p_e as well as adding a single link to a node randomly drawn from the network. The strategy mutation involves changing application behavior with low probability p_m.

The Prisoner's Dilemma represents a classical example of the Tragedy of Commons. Prisoner's Dilemma game captures a situation where a contradiction exists between self-benefit and the collective benefit.

In [4], [5] it is shown how it is possible to establish cooperation in a P2P network where each peer application-level behavior involves nodes playing the Prisoner's Dilemma with randomly selected neighbors. This approach could lead in the future to a new generation of P2P framework, where the P2P framework only gives generic support for cooperation, and the single application will be able to choose what the cooperation means and how to use it.

4. Real-Time Distribution

The are several reasons that make P2P attractive for real-time services. First of all, it does not need support from IP routers like IP-multicast. Second, a participant not only downloads the content but he also uploads it to other participants. There are two enormous consequences of this double role of the participants: *(i)* it shifts a significant portion of the cost from the content provider to the end users; and *(ii)* the P2P paradigm has the potential to scale with group size, as greater demand also generates more resources.

Large Scale Streaming – IPTV is probably the most popular class of applications for large scale streaming. In most successful P2P-IPTV applications the media stream is divided into blocks (usually called chunks) and delivered over the overlay topology in each session, with reciprocal exchanges of useful chunks among the peers.

It is interesting to point out that two features distinguish P2P-IPTV applications from others: *(i)* the scale, with thousands (millions??) of simultaneous users; and *(ii)* bandwidth requirements: bit rates range from 250 to 400 kbit/s, but high quality TV will require bit rates in excess of 1.5 Mbit/s. The combination of these peculiarities yields an applicative scenario that is different from file sharing, but it is also different from *on-demand streaming* and *audio/video conferencing*.

Several P2P-IPTV platforms appeared and provide live video over the Internet, a non-exhaustive list of them includes PPLive, PPStream, SOPCast. These proposals (together with the scientific literature supporting them) differ on a wide range of characteristics. In particular, according to the overlay topology these platforms can be classified in two classes: *tree-based* and *mesh-based*.

In tree-based approaches the overlay is composed of one or several trees that are used to distribute the content to the peers. Tree based applications have a well organized structures and (typically) distribute the streaming by pushing data from a peer to its children peers. One of the major problem of tree based systems is related to resilience to peer churn. A peer departure interrupts the delivery to all peers in the subtree rooted at the departed peer. To minimize the impact of peer departures the tree structure needs to be rebuilt and the applications may have different overlay management strategies aiming to reconstruct the tree structure. Another problem of these architectures is that the upload bandwidth of the leaf peers of the tree is not used. Examples of single tree-based streaming includes ESM [6] and Overcast [7].

To address the drawbacks of the single tree-based architectures multi-tree based approaches have been proposed. In these architectures the source encodes the stream into sub-streams and distributes each sub-stream along a particular overlay tree. The QoS experienced by a peer depends on the number of sub-streams that it receives. The reliability of the system is improved because a peer is not completely disconnected due to a failure of one of its parents on a given tree. Moreover, this approach improves the overall bandwidth utilization as long as each peer is not a leaf in at least one of the trees. Examples of single tree-based streaming includes [8], [9]. Multi-tree proposals are inherently connected to Multiple Description Coded (MDC) content [10], [11].

In the mesh based approach peers form a randomly connected mesh and use gossip-like protocols for the creation and the administration of the overlay. Peers establish and terminate peering relations dynamically. At a given time a peer maintains neighboring relationships with a set of other peers and it may upload/download portion of the streaming content to/from a subset of its neighbors.

Recently [12] provided a comparison between mesh and multi-tree approaches as well as the identification of similarities and differences. Despite the advantages offered by tree-based approaches the results indicated that the mesh-based approach consistently exhibits a superior performance over the tree-based approach. The main drawbacks of tree-based approaches are reported to be the large overhead due to the organization of the tree structures, the difficulty to respond to the dynamics of peers, and the difficulty to achieve an efficient use of the upload bandwidth peers resources.

Nevertheless, the mesh-based approach suffers from other problems; the use of sophisticated video coding techniques such as for instance the MDC, is not straightforward since one has to guarantee that different descriptions travel along independent paths to maximize video quality. Furthermore, mesh-based approaches suffer from the tradeoff between control overhead and delay. In fact, to minimize delay peers must notify their neighbors of available packets as frequently as possible, thus resulting in high control overhead. On the other hand, to reduce control messages notifications must be aggregated over time thus making the delay higher.

An attempt to bring the benefits of tree-based approaches in the mesh-based scenario is presented in [13]. The authors propose a pull-push hybrid protocol where packets are pushed along the trees formed by pull-based protocols.

Other efforts to combine the advantages of push and pull based techniques have been presented in [14]. The key idea of these proposals is that the peers alternate regularly pushing and pulling operations. The push operations are devoted to distribute "fresh" information to other peers. The pull operations to retrieve missing information.

Medium-Small Scale: Conferencing – If large scale P2P-TV attracted enormous attention, to the point of becoming subjects of social research, P2P small scale real-time systems, specifically video conferencing, seem to be less attractive both for research and for applications. Only few papers (see for instance [15], [16]) can be found on the subject, while open source working systems are simply non-existent. A few commercial software tools claim to be P2P, but does not present evidence of the internal workout.

Whether this difference is intrinsic to the problem of video conferencing, which does not lend itself to P2P solutions (we doubt this, but . . .), or to less appealing scientific problems behind it (scaling problems of P2P-TV are appealing and easy to tackle!!), and more complex technical issues to solve, we cannot ascertain.

The fact remains that this specific application and all its "related problems", have received less attention and seem to attract less investments compared to large scale streaming. This is the more puzzling when confronted with the fact that TV services exists, are cheap and of good (technical) quality, while video conferencing remains an almost non-existent service, albeit much invoked by many communities.

5. Privacy and Security

Security is normally among the top topics in networking, and P2P systems are no exception. Often security is not coupled to privacy, even contrasted to it, as if protecting fundamental rights of users (non disclosure and proper handling of sensitive data) jeopardizes the system protection, but this is a misperception.

Privacy vs. Security or Both? – Network and service providers often think about privacy as the possibility for malicious users to damage the system and hide. Indeed, privacy is the right of users to be subject to correct handling of *personal and sensitive data*, and to be protected from linkage activities on the disclosed data. Privacy does not prevent authentication and neither does anonymity with respect to a specific service.

In a client/server model *enforcing* both security and privacy can be *technically* very difficult (recent cases on facebook prove this) because the same juridical person has physical access to all data. In a P2P distributed system enforcing security and privacy at the same time is technically easier, since no single juridical person has physical access to all data.

In P2P systems, threats to security and privacy are related to four "entities":

1) Peer nodes – They can be malicious or simply malfunctioning. Missing or wrong replies and malicious requests/tampering are threats that can lead to overlay partitioning or service deny.

2) Supernodes – These are, in many systems, nodes with higher responsibilities (info sources, repositories, support nodes, . . .). Having higher responsibilities they are also more delicate w.r.t. security and privacy issues.

3) P2P Application Code – It often runs with privileges on peer machines ([17]). If no other privileges are assigned, in any case it uses the network connection and the local hard drive: local access and external communication can lead to information leaks or malicious code installation.

4) Distributed Data – Whatever the application, the data distributed by a P2P system must be trustworthy, and always maintain the original semantic: any modification/deletion of data can result in censorship (in its most general acception).

5.1. Attacks to P2P Systems

The paper [18] provides an introduction to the security threats that menace a DHT based P2P network.

Routing attacks – Many variants of this attack are applicable to P2P networks. It is generally possible when somebody is able to intercept the routing requests to a certain resource and redirect them somewhere else. The consequences of such an attack are denial of service (DoS), privacy invasion or substitution of resources: peers are redirected to a resource which is not the one they wanted.

Partition attacks – New peers relays on a set of nodes known in advance to start populating its peer list. If the initial nodes are malicious ones, then they can present to the newcomer a parallel network they control. If this fake network has a connection to the real one, the victim will still be able to perform successful queries but all its activities will be controlled and monitored.

Retrieval attack – The attacker can take the ownership of a certain resource and then decide who is able to access it and who is not.

Sybil and Eclipse attacks – A sibyl is a replica of a node controlled by an attacker that he uses to intercept requests. For example, in DHT-based P2P network an attacker can use the same IP address but multiple IDs spread all over the ID space in order to intercept as much requests as he can. An eclipse attack is performed by a set of sibyls distributed in the *neighborhood* of a victim resource. The aim of the attacker is to intercept all the requests directed to that resource.

5.2. Reputation and Trust

The reputation is an integral part of the trust concept and it is very important for the establishment of trust relationships between two peers. Particularly, all existing decentralized trust management techniques for P2P communities can be divided in two groups depending on the approach used to establish and evaluate trust relationships between peers: *(i)* credential and policy based, and *(ii)* reputation based. In credential and policy based systems the peers use a set of credentials and a set of policies to determine weather some unknown peer can be trusted or not. This approach is typically used for authorization and access control in open systems, and it is intended for systems with strong protection requirements. Unfortunately in this case the presence of certification authorities is required, introducing a partial centralization. For this reason, only techniques based on pure reputation mechanisms seem to offer a real distributed solution. The basis of each reputation-based technique is a trust computational model that provides mechanisms to evaluate the level of trust toward both a resource and its possessor. Another very important aspect for this approach is the management of reputation data based on the recommendations and experiences of other users. Normally, this data is not signed by certification authorities, but it can be self-signed by the source of the information [19].

Recently, a number of reputation management techniques for P2P networks have been proposed by different researchers. Some of these techniques are: NICE [20], DCRC/CORC [21], Poblano [22], XREP [23], PeerTrust [24], and Fuzzy Model [25]. NICE and DCRC/CORC exploit credential and policy elements: digital signatures of cookies in NICE, peer identification by reputation computation agents using public key in DCPC/CORC. Poblano and XREP also involve some mechanisms with a centralized nature. PeerTrust and Fuzzy Model represent the most complete techniques, as they realize almost all possible mechanisms for evaluation of a peer's trustworthiness.

Some requirements that should be considered when evaluating a reputation technique are: *(i)* possibility to provide recommendations; *(ii)* possibility to "weigh" recommendations; *(iii)* responsibility for the behavior of recommended entities; *(iv)* evaluation of a community context; *(v)* incentives for feedback compilation.

In [26] an analysis of applicability of several existent reputation evaluation techniques to P2P networks is presented. Unfortunately, most of the reputation techniques specifically proposed for P2P networks are heavily based on evaluation mechanisms of successful and unsuccessful downloads suitable for file sharing but not directly applicable to other P2P applications such as collaborative applications, social networking, distributed computing, etc. [27], [28].

6. Experiences from Profiles

The goal of PROFILES was indeed the exploration of P2P applications beyond file sharing, and we are pleased, in concluding this paper, to present some of the contributions we achieved there: we cannot include here all the results developed in the project and we apologize to colleagues whose work is not cited. Additional details can be found on the web site *http://profiles.disi.unitn.it/*.

6.1. Models and Analysis

Inspite of its success the P2P paradigm still need fundamental understanding, so that modeling and analysis activities are in full bloom.

The first set of results PROFILES produced use classical frameworks based on Markov chains as well as semi-analytical techniques based on the numerical solution of semi-Markov processes. For what concerns the use of

Markov chain based framework [29] proposes an analytical model of a real-time video transmission system using P2P to achieve multipoint communication on the current Internet.

A different perspective is taken in [30], [31]. The key point of these contribution is the methodology. The use of stochastic graph theory, adding constraints to the evolution of the process, enables to embed within the stochastic process properties of the protocols and the overlay management. The result is an extremely scalable technique, that allows exploring systems with tens of millions of nodes, albeit with a coarse level of detail.

Also related to the use of graphs as modeling abstraction, and in particular to *Generalized Random Graphs*, [32] contributes to research oriented to make the impact of P2P overlays gentler on the IP network. This work can be seen as the modeling counter-part of the P4P proposal introduced in [2], and it proposes strategies to build network-aware P2P overlay topologies.

Again addressing the problem of P2P overlay an IP network interaction, [33] proposes an analytical framework for the evaluation of different strategies for managing P2P traffic and of its impact on the costs (for the ISP and for the users) of several different parameters (number of users for a given ISP, resource popularity, . . .).

6.2. Middleware and Routing

Middleware and routing in P2P systems are easily confused with the application exploiting the overlay. Almost all P2P systems are built around a specific application, with 'lower' communication layers tailored for that specific application.

Resource allocation – Resource allocation in P2P networks is the problem related with *what resource should be shared (and by what peer) in order to maximize the network efficiency*. This also leads to the companion problem: *how to gracefully force peers to share their resources*.

An algorithm based on the Prisoner's Dilemma has been developed and tested in the PROFILES project. Consider a video streaming application based on unstructured P2P overlay networks.

Each node is characterized by an input bandwidth B_{in} and an output bandwidth B_{out}. The strategy of an individual peer consists of appropriately choosing the band parameters B_{in} and B_{out} (which means changing the peer's behavior at the application level). Further, each peer must also select the list of *distributors* (i.e. peers from which it receives the video streams).

Periodically, each peer generates a measure of *video utility* U, based on the number of received flows N, band parameters B_{in}, B_{out}, etc. Peers compare U against the utility of other peers chosen at random.

Let us consider the case in which the overall video bandwidth B_{vid} is such that $B_{in}+B_{out} \leq B_{vid}$ for all peers.

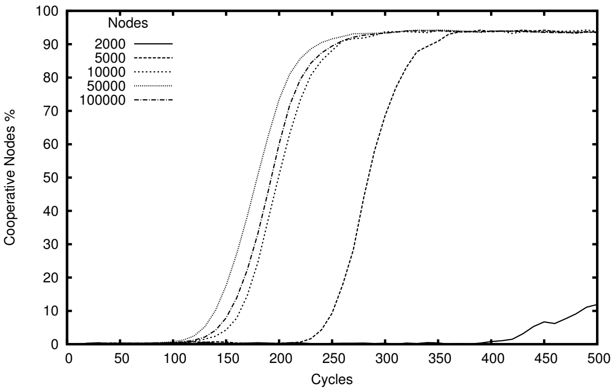

Figure 1. Percentage of cooperative peers for different values of the overall number of peers

This constraint implies that the dilemma between individual and collective benefit is actually present since the available band is not enough for the full quality video reproduction.

Simulation experiments concerning the cooperation/no cooperation dilemma in a P2P network have been carried out using the PeerSim simulator.

Fig. 1 plots the percentage of cooperative peers for different values of the overall number of peers. Notice demonstrates the scalability property of the algorithm as faster convergence is obtained for larger P2P networks.

Routing – Consider once more video distribution, and in particular in tree-based video streaming. The video quality is heavily dependent on two factors: *(i)* the tree construction; and *(ii)* whether the trees take advantage or not of the underlying physical network. In general we can state that the routing problem can be seen as *provided that there are multiple instances of a resource, which one should a node pick up?* A reasonable strategy is chosing the 'best' source according to a set of parameters, e.g., delay, jitter, etc.

One of the research activities in PROFILES project was focused on how to create a DHT-independant structure in order to store resource availability with a greater and more complex structure than the one available through the standard DHT.

The outcome is called Spare Capacity Group (SCG), and the key concept has been borrowed from SplitStream [8] SCG should hold all the nodes that have a resource (video stream) to be re-distributed, along with information about the resource itself. Any client willing to get a resource should be able to ask the SCG for some information and have a list of possible nodes that could fulfill its needs. Then, it is up to the client's decision process to choose the best alternative.

The SCG procedure has been implemented in PeerSim. Two different search procedures have been realized:

- HDRRW - High Degree Restricted Random Walk
- PFW - Probabilistic Flooding Walk

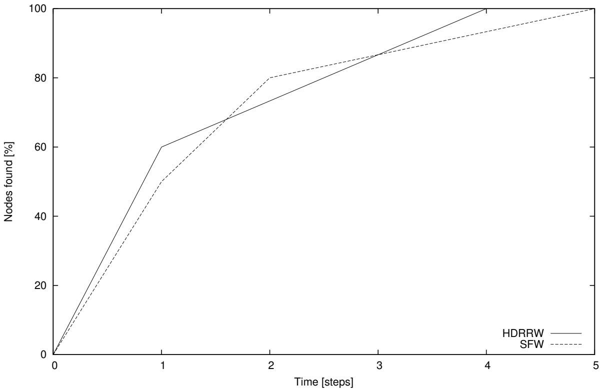

Figure 2. SCG convergence speed

The simulation results show that both algorithms perform quite well, with similar results.

The convergence of the two algorithms is also very close, with PFW being slightly faster. Fig. 2 shows that, with a 10.000 nodes network, both algoritms can find all the resource sharing nodes within 5 steps, with a 'step' being the time unit between two consecutive algorithm calls.

6.3. Applications and Implementation

The DLS concept described in Sect.2 has been realized in PROFILES with two prototype applications. All software is in Java and runs on general purpose PCs, PDAs, or smathphones. The current DLS implementation supports two different DHT algorithms: Chord and Kademlia; and it uses a SIP-based peer protocol developed starting from the open source Java SIP project MjSip [34]. In order to let the DLS to be accessed also by DLS-unaware nodes, a Peer Adapter node has been also introduced and implemented. Currently supported client protocols are SIP and HTTP, but other client protocols may be supported in the future if required. Two different applications exploiting the DLS has been defined and realized: VoIP services, and a distributed web server.

Peer-to-Peer VoIP Services – Pure P2P VoIP calls can be performed by exploiting the DLS simply as a SIP LS. DLS-aware UAs can register their contact into the DHT through a put() request, and can initiate sessions with other users by retrieving the contact information from the DHT and successively exploiting this information for establishing the session.

The implemented VoIP platform is also compatible with legacy SIP UAs, like standard softphones or IP phones. For this purpose, it is sufficient that the legacy UAs are configured to send all SIP requests to a proper SIP Peer Adapter acting as UA's default Outbound Proxy. When a caller UA wants to perform a SIP call, it sends an INVITE request to the Peer Adapter. The Peer Adapter performs a lookup to resolve the target user's address and retrieve its location, then it forwards the INVITE request to the callee UA. Fig. 3 shows an example of P2P SIP call.

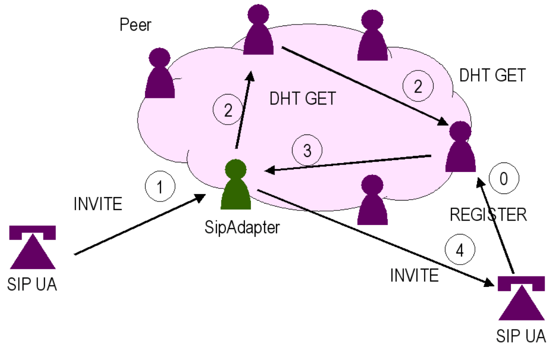

Figure 3. P2P SIP call

Distributed web server – Another DLS-based application is a HTTP distributed virtual server or distributed web server (DWS). In such application, a virtual web server is deployed and distributed amongst all participating peers. This means that the web resources (files) are not stored on the same host but are published by a number of collaborative nodes. Data replication can be achieved by storing the same information on different hosts, and the web site contents can be partitioned among several nodes. Resource registration can be performed with a HTTP PUT request by a proper Publishing Agent present within all DWS peers. A resource can be accessed by a standard HTTP UA (e.g. web browser) as shown in Fig. 4.

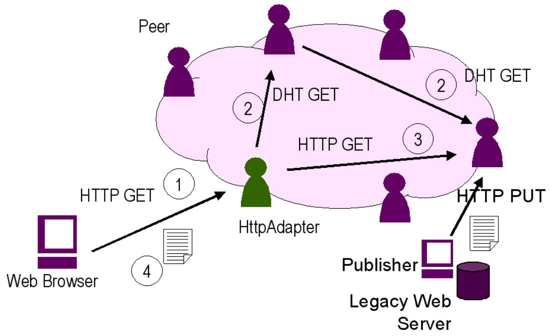

Figure 4. Access to a distributed web server

The access to resources occurs transparently as the resources would all be registered with resource names that refer to the same virtual domain. The end user would not be aware of the fact that resources are distributed among a number of nodes.

Acknowledgements

This work was supported by the PROFILES PRIN Project funded with grant No. 2006099023 by the Ministry of Education and Research: http://profiles.disi.unitn.it.

In some broad sense this paper represents a summary of the Profiles project itself. As authors, but also as project coordinators, we are deeply in debt with all the people that contributed to the success of Profiles, listed in the project wep page.

A particular thanks goes to Leonardo Maccari who directly contributed to work presented in this paper.

References

[1] D. Carra and E. Biersack, "Building a reliable P2P system out of unreliable P2P clients: the case of KAD," in *ACM CoNEXT 2007*, New York, NY, Dec. 10–13, 2007.

[2] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPs and P2P users cooperate for improved performance?" *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 3, pp. 29–40, 2007.

[3] Yang Chen et al., "Myth: An Accurate and Scalable Network Coordinate System under High Node Churn Rate," *15th IEEE Int. Conf. on Networks (ICON)*, Nov. 19–21, 2007.

[4] D. Hales and B. Edmonds, "Applying a socially inspired technique (tags) to improve cooperation in p2p networks," *Systems, Man and Cybernetics, Part A, IEEE Tr. on*, vol. 35, no. 3, pp. 385–395, May 2005.

[5] D. Hales and S. Arteconi, "Slacer: a self-organizing protocol for coordination in peer-to-peer networks," *Intelligent Systems, IEEE*, vol. 21, no. 2, pp. 29–35, March-April 2006.

[6] Y. Chu, S. Rao, and H. Zhang, "A case for end system multicast," in *ACM SIGMETRICS*, Santa Clara, CA, June 17–21, 2000.

[7] J. Jannotti et al., "Overcast: Reliable Multicasting with an Overlay Network," in *USENIX OSDI 2000*, San Diego, CA, Oct. 22-25 2000.

[8] M. Castro et al., "SplitStream: High-Bandwidth Multicast in Cooperative Environments," in *19th ACM SOSP*, Bolton Landing, NY, Oct. 19–22 2003.

[9] D. Kostic et al., "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh," in *19th ACM SOSP*, Bolton Landing, NY, Oct. 19–22 2003.

[10] V. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Tr. on Inform. Theory*, vol. 39, no. 3, pp. 821–834, 1993.

[11] T. Tillo, M. Grangetto, and G. Olmo, "Redundant slice optimal allocation for H.264 multiple description coding," *IEEE Tr. on Circuits and Systems for Video Technology*, vol. 18, no. 1, 2008.

[12] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live p2p streaming approaches," in *26th IEEE INFOCOM*, Anchorage, AK, May 2007.

[13] M. Zhang et al., "Understanding the Power of Pull-Based Streaming Protocol: Can We Do Better?" *IEEE JSAC*, vol. 25, no. 9, pp. 1678–1694, 2007.

[14] T, Locher et Al., "Push-to-Pull Peer-to-Peer Live Streaming," in *21st DISC*, Lemesos, Cyprus, Sept. 24–26, 2007.

[15] M. Civanlar, O. Ozkasap, and T. Celebi, "Peer-to-peer multipoint videoconferencing," in *ICIP'04*, Oct. 24–27, 2004.

[16] Y. Liu, "On the minimum delay peer-to-peer video streaming: how realtime can it be?" in *ACM MULTIMEDIA'07*, Augsburg, Germany, Sept. 24–29 2007.

[17] D. S. Wallach, "A survey of peer-to-peer security issues," in *Int. Symposium on SW Security*, Tokyo, JP, Nov. 8-10 2002.

[18] E. Sit and R. Morris, "Security considerations for peer-to-peer distributed hash tables," in *1st Int. Workshop on P2P Systems (IPTPS)*, 2002.

[19] P. Bonatti et al., "An integration of reputation-based and policy-based trust management," in *Semantic Web Policy Workshop*, Galway, Ireland, Nov. 5–9 2005.

[20] S. Lee et al., "Cooperative peer groups in NICE," in *IEEE INFOCOM'03*, San Francisco, CA, 2003.

[21] M. Gupta et al., "A Reputation System for Peer-to-Peer Networks," in *NOSSDAV '03*, Monterey, CA, June 1–3, 2003.

[22] R.Chen and W. Yeager. Poblano: A Distributed Trust Model for Peer-to-Peer Networks. [Online]. Available: http://www.jxta.org/docs/trust.pdf

[23] E. Damiani et al., "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," in *9th ACM Conf. on Comput. and Commun. Security*, Washington DC, 2002.

[24] L. Xiong and L. Liu, "A Reputation-Based Trust Model for Peer-to-Peer Ecommerce Communities," in *IEEE Conf. on E-Commerce (CEC'03)*, June 2003.

[25] V. Grishchenko, "A fuzzy model for context-dependent reputation," in *Trust, Security and Reputation Workshop at ISWC'04*, Hiroshima, Japan, 2004.

[26] N. Fedotova, M. Bertucci, and L. Veltri, "Reputation Management Techniques in DHT-based Peer-to-Peer Networks," in *2nd Int. Conf. on Internet and Web Applications and Services*, Mauritius, May 2007.

[27] S. D. Kamvar, M. Schlosser, and H. Garsia-Molina, "The Eigen Trust Algorithm for Reputation Management in P2P Networks," in *12th Int. WWW Conf.*, May 2003.

[28] S. Y. Lee et al., "A Reputation Management System in Structured Peer-to-Peer Networks," in *14th IEEE Int. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprise (WETICE'05)*, 2005.

[29] G. Incarbone, G. Schembra, and A. Lombardo, "Applying P2P to Achieve Real-Time Multimedia Communications: Modeling and Performance Analysis," in *IEEE Globecom*, Washington, DC, 2007.

[30] D. Carra, R. Lo Cigno, and E. W. Biersack, "Graph Based Analysis of Mesh Overlay Streaming Systems," *IEEE JSAC*, vol. 25, no. 9, pp. 1667–1677, 2007.

[31] ——, "Stochastic Graph Processes for Performance Evaluation of Content Delivery Applications in Overlay Networks," *IEEE Tr. Parallel Distrib. Syst.*, vol. 19, no. 2, pp. 247–261, 2008.

[32] R. Gaeta and M. Sereno, "Random graphs as models of hierarchical peer-to-peer networks," *Performance Evaluation*, vol. 64, no. 9-12, pp. 838–855, 2007.

[33] M. Garetto et al., "A modeling framework to understand the tussle between ISPs and peer-to-peer file-sharing users," *Performance Evaluation*, vol. 64, no. 9-12, pp. 819–837, 2007.

[34] "MjSip project home page," http://www.mjsip.org.