A Survey of Peer-to-Peer Overlay Schemes: Effectiveness, Efficiency and Security

Michele Amoretti*

Information Engineering Department, University of Parma, Via Usberti 181/a, 43100 Parma, Italy

Received: February 17, 2009; Accepted: May 11, 2009; Revised: June 2, 2009

Abstract: In distributed computing, the peer-to-peer paradigm enables two or more entities to collaborate spontaneously in an overlay network of equals (peers) by using appropriate information and communication schemes without the necessity for central coordination. The key concept of the peer-to-peer paradigm is leveraging idle resources to do something useful, based on a collaborative approach. The increasing academic and industrial interest is resulting in the definition of standards and writing of patents.

In this paper we propose a categorization for the peer-to-peer overlay schemes and a survey of the most popular ones, comparing each other with respect to effectiveness and security. Most of them have been or are being used in content sharing systems, that over the last few years have enjoyed explosive popularity. Others are used in parallel and distributed computing, massively multi-player gaming, Internet streaming, ambient intelligence, etc. Considering such a wide range of applications, we discuss the importance of reputation management in supporting trust management among peer participants.

Keywords: Peer-to-peer, resource sharing, overlay network, security.

1. INTRODUCTION

The main goal of a distributed system is to connect users and resources in a transparent, open, and scalable way. Ideally this arrangement is drastically more fault tolerant and powerful than many combinations of stand-alone computer systems. Nevertheless, large-scale distributed applications are becoming more and more demanding in terms of efficiency and flexibility of the technological infrastructure, for which traditional solutions based on the client/server paradigm are no more suitable.

This paper focuses on the enforcement of the peer-to-peer (P2P) paradigm to distributed systems, enabling two or more entities to collaborate spontaneously in a network of equals (peers) by using appropriate information and communication systems without the necessity for central coordination [1]. Furthermore, a peer-to-peer system is a complex system, because it is composed of several interconnected parts that as a whole exhibit one or more properties (i.e. behavior) which cannot be easily inferred from the properties of the individual parts.

The P2P paradigm has emerged as a highly appealing solution for scalable and high-throughput resource sharing among decentralized computational entities, by using appropriate information and communication systems without the necessity for central coordination. In distributed systems based on the peer-to-peer paradigm all participating processes have the same importance. In contrast with the client/server approach, in which resource providers and resource consumers are clearly distinct, on the contrary peers usually play both roles. One of the early driving forces behind the usage of the peer-to-peer paradigm in distributed computing is that there are many PCs in homes and offices that lie idle for large chunks of time. The key concept of the peer-to-peer paradigm is leveraging idle resources to do something useful, like cycle sharing or content sharing. Peers share their resources (bandwidth, cache, CPU, storage, files, services, applications), to a different extent. Peers that share a minimal amount of resources but consume a lot of resources are called free riders. Usually, some resources are implicitly shared by all peers (e.g. cache for storing local or remote resource advertisements, bandwidth for routing incoming messages, etc.) - we call them *basic shared resources*. In mobile ad-hoc networks, peers are allowed to limit the provisioning of basic shared resources in order to save device batteries. Resources that usually cannot be acquired (by replication), but may only be directly used upon contracting with their hosts, are called *consumable resources*.

The lack of centralized control makes network and system security the most important features in most P2P systems, and acts as integral component of the design process. Since P2P systems inherently rely on the dependence of peers with each other, security implications arise from abusing the trust between peers. In a traditional client/server model, internal data need not be exposed to the client, but with P2P some internals must be exposed to fellow peers in the name of distributing the workload. Attackers can leverage this in compromising P2P networks. In this paper we propose a survey of the most popular P2P overlay schemes and their applications, taking into account the possible security flaws by which they can be affected, and the solutions that are eventually applied.

We consider not only academic research but also patents. Some of them are very interesting, and maybe will contribute

*Address correspondence to this author at the Information Engineering Department, University of Parma, Via Usberti 181/a, 43100 Parma, Italy; Tel: 0039 0521 906147; Fax: 0039 0521 905723; E-mail: michele.amoretti@unipr.it to overcome the major problem affecting P2P design, i.e. the lack of widely accepted standard platforms. During the last ten years, many different P2P architectural blueprints and protocols have been proposed. Unfortunately each developer promoted his/her brainchild targeted to a specific application and with a restricted set of functionalities, without any sort of significant coordination and standardization effort. This fact has often produced overlapping and redundancy of functionalities and a complete lack of interoperability.

Probably the most significant work in the direction of an unique signaling platform and protocol suite is the JXTA project [2]. JXTA is a set of open source protocols, originally conceived by Sun Microsystems, for creating an overlay network in which any connected peer can interact and exchange messages with other peers and resources directly, regardless the topology of the underlying network. JXTA protocols are defined as a set of XML messages and provides all mechanisms for publishing and searching resources, and for realizing communication channel between peers. The JXTA platform provides core building blocks (IDs, advertisements, peergroups, pipes) and a default set of core policies, which can be replaced if necessary. JXTA protocols have been implemented in Java and C, using a particular overlay scheme that is discussed in section 2 of this paper. We noticed that JXTA authors owns a patent, called “Peer-to-peer computing architecture” [3], which basically includes all JXTA protocols (without explicit reference).

Other important P2P standardization attempts have been done. The Internet Engineering Task Force (IETF) has formed working groups to address specific issues of P2P networking:

- Peer-to-Peer SIP signaling (P2PSIP)
- Application-Layer Traffic Optimization (ALTO)
- Low Extra Delay Background Transport (LEDBAT)

P2PSIP [4] is the most active P2P WG, having provided several proposals and a specific signaling protocol [5]. As stated by the P2PSIP working group itself, the goal is to develop protocols and mechanisms for establishing and managing sessions completely handled by peers. The primary objective of such protocols is to provide a common platform for user and resource location in a complete distributed environment. Although it was started for using for end-to-end real-time session initiation, P2PSIP could apply also to any other interactive or transactional applications.

The paper is organized as follows. In section 2 we illustrate the design issues related to peer-to-peer systems, with particular emphasis on security aspects. In section 3 we propose a categorization of overlay schemes based on the topology awareness of peers and the placement of information about shared resources. In section 4 we discuss the most important protocols and algorithms, considering content sharing but also other application domains. In section 5 we focus on reputation management, which is an interesting solution for supporting trust management in peer-to-peer systems. Finally, in section 6 we present some concluding remarks and we discuss future developments in the field.

2. DESIGN ISSUES

The operation of any peer-to-peer system relies on a network of peer software/hardware nodes, and connections (links) between them. This network is formed on top of - and independently from - the underlying physical computer (typically IP) network, and is thus referred to as an *overlay network*. The topology, structure, and degree of centralization of the overlay network, and the message routing and location mechanisms it employs for messages and resources are crucial to the operation of the system, as they affect its scalability, security, fault tolerance, and self-maintainability.

In recent years, the research on P2P systems has focused on designing robust overlay schemes (defining the rules for bootstrapping, connectivity, message routing, caching), usually targeting specific applications. In the following we briefly recall the most intriguing P2P design challenges, in order to have a base for the categorization of overlay schemes proposed sections 3 and for the survey of the most popular ones presented in section 4. In our opinion, a P2P system should be effective, efficient and secure, both in general and with respect to its specific application(s).

2.1. Effectiveness & Efficiency

The focus of effectiveness is the achievement of a goal, by performing the right actions. Efficiency means doing things in the most economical way (good input to output ratio). What is effective is not necessarily efficient, in P2P systems. Here we illustrate the main issues that affect the effectiveness and efficiency of P2P architectures.

The **scalability** of an overlay scheme measures its effectiveness and efficiency, with respect to the target application (s), when applied to large situations (e.g. large workloads or large number of participating nodes). A distributed system should be inherently more scalable if using the P2P paradigm, rather than the client/server approach. But some P2P overlay schemes scale better than others, with respect to resource discovery effectiveness and performance, bandwidth occupancy, etc. For example, a message routing protocol is considered scalable with respect to network size, if the number of message propagations that are necessary to find a resource grows as $O(\log N)$, where N is the number of nodes in the network.

The effectiveness and efficiency of P2P protocols for resource sharing usually depends on how peer are connected. One key operation is **bootstrapping**, the initial discovery of other nodes participating in the network. Nascent peers need to perform such an operation in order to join the network. Bootstrapping usually includes operations needed to repair overlays that have split into disconnected subgraphs [6]. Another important operation is **connectivity management**, i.e. the maintenance of connections or exchange of topology information for peers that are already connected to the network at large.

Search performance and **consistency** are two important measures for the sharing of dynamic contents (e.g. in P2P storage systems). Search performance concerns how fast the users locate and obtain copies of requested resources (time complexity) and how many nodes must be involved in that process (space complexity). Consistency concerns how old the acquired data (resource descriptions or shared content) are with respect to the actual available resources.

Stability refers to the ability of the overlay scheme to guarantee the correctness of its functionalities regardless of the churn rate of the network. The churn rate is a measure of the number of peers joining and leaving the system over a specific period of time. Users of the peer-to-peer system join and leave the network randomly, which makes the overlay network dynamic and unstable in nature. This dynamical behavior of the peers frequently partitions the network into smaller fragments which results in the breakdown of communication among peers. Thus, design and tuning of the overlay scheme should be made taking into account different churn rate profiles, depending on the distributed application the scheme should support.

Finally, **load balancing** is the measure of how fair is the distribution of computational, storage and bandwidth consumption among the nodes participating in the network. Whilst this issue is well defined and basically solved for systems with relatively simple search paradigms, only few known solutions are appropriate or applicable for similarity-search networks (e.g. supporting semantic or fuzzy resource discovery).

Another issue that potentially can hinder the deployment of large-scale P2P applications is **asymmetric bandwidth** in the access network. In particular, the uploading capability of each peer can become a bottleneck in the system. This can significantly impact ISP (Internet service providers) and how ISP perform traffic dimensioning. Moreover, a large portion of the Internet bandwidth is occupied by P2P applications, where many ISP have enforced traffic engineering mechanisms, in particular for inter-domain traffic. For content sharing, this implies considerable slowdown in performance; but for streaming applications, this can be fatal. Finally, NAT and firewalls can impose fundamental limitations on the pair-wise host connectivity in the overlay network. It is well-known that a significant portion of broadband users experience NAT or firewall problems, and this requires particular attention.

2.2. Security

Peer-to-peer architectures present a particular challenge for providing high levels of privacy, confidentiality, integrity, and authenticity, due to their open and autonomous nature. Preserving integrity and authenticity of resources means safeguarding the accuracy and completeness of data and processing methods. Unauthorized entities cannot change data; adversaries cannot substitute a forged document for a requested one. Privacy and confidentiality mean ensuring that data is accessible only to those authorized to have access, and that there is control over what data is collected, how it is used, and how it is maintained. A malicious node might give erroneous responses to requests, both at the application level, returning false data, or at the network level, returning false routes and partitioning the network. Moreover, the P2P system must be robust against a conspiracy of a *malicious collective*, i.e. a group of nodes acting in concert to attack reliable ones. Attackers may have a number of goals, including traffic analysis against systems that try to provide anonymous communication, and censorship against systems that try to provide high availability.

Security attacks in P2P systems can be classified into two broad categories: passive and active [7]. Passive attacks are those in which the attacker just monitors activity and maintains an inert state. The most significant passive attacks are:

- **Eavesdropping**. It involves capturing and storing all traffic between some set of peers searching for some sensitive information (such as personal data or passwords).
- **Traffic analysis**. Here the attacker not only captures data but tries to obtain more information by analyzing its behavior and looking for patterns, even when its content remains unknown.

In active attacks, communications are disrupted by the deletion, modification or insertion of data. The most common attacks of this kind are:

- **Spoofing**. One peer impersonates another, or some outside attacker transforms communications data in order to simulate such an outcome.
- **Man-in-the-middle**. The attacker intercepts communications between two parties, relaying messages in such a manner that both of them still believe they are directly communicating. This category includes data alteration between endpoints.
- **Playback or Replay**. Some data exchange between two legitimate peers is intercepted by the attacker in order to reuse the exact data at a later time and make it look like a real exchange. Even if message content is encrypted, such attacks can succeed so long as duplicate communications are allowed and the attacker can deduce the effect of such a repeat.
- **Local Data Alteration**. This goes beyond the assumption that attacks may only come from the network and supposes that the attacker has local access to the peer, where he/she can try to modify the local data in order to subvert it in some malicious way.
- **No-forwarding**. An important issue that looms over P2P networks is blocking and throttling of P2P traffic. According to a 2007 Internet study, 69% of Internet traffic in Germany is P2P, with HTTP way behind at 10%. Within P2P traffic, BitTorrent [8] accounts for 67%, with the next highest being eDonkey at 29% [9]. Given the staggering proportion of Internet traffic accounted for by P2P applications, especially BitTorrent (from the numbers above, BitTorrent alone accounts for nearly 50% of the Germany's Internet traffic), it is not surprising that ISPs are starting to block ports on which well-known file sharing applications run. For example, Comcast recently started to throttle and drop packets of BitTorrent traffic, effectively blocking its customers from running the software [10].
- **Free Riding**. So far we generally assumed that peers are willing to collaborate with one another. However, this is not always true in reality. In a typical P2P system, some users tend to be selfish, for example, saving resources

and taking free rides (such users are called *free riders*). Potentially this has a serious impact on the performance of cooperative peers. Some researchers have explored the benefits of enabling virtual communities to self-organize and introduce incentives as a resource sharing and cooperation, arguing that what is missing from today's peer-to-peer systems, should be seen both as a goal and a means for self-organized virtual communities to be built and fostered [11]. Ongoing research efforts for designing effective incentive mechanisms in P2P systems, based on principles from game theory are beginning to take on a more psychological and information processing direction. However, it is not possible to create a general-purpose framework. For example, in file sharing, credit-based or reputation-based mechanisms can be adopted, but this is difficult to do for streaming applications. The reason is that in live streaming applications it is difficult to collect credits or a reputation due to the timing requirement. How to design an incentive mechanism to combat the possible misbehaving peers remains an open problem. It has been shown that the tit-for-tat strategy (adopted by file sharing applications like BitTorrent [8]) does not work for live streaming applications [12]. This is because each peer only maintains streaming data in a window with a short time period, which implies that it is nearly impossible for a descendant of a peer in one sub-stream to be an ancestor of other sub-stream.

- **Distributed Denial of Service (DDoS)**. In a traditional denial-of-service (DoS) attack, a server is usually the target of massive connections, rendering the server inoperable. In a P2P network, attackers can make use of the querying nature of P2P networks to overload the network. In the case of the query flooding P2P network, the attack is straightforward: simply send a massive number of queries to peers, and the resulting broadcast storm will render portions of the network inoperable. More recently, attacks can harness the P2P network as an agent to attack some *other* target, such as web sites. Essentially, peers in the network are subverted to request files from a target, overwhelming the victim with enormous bandwidth usage. An example of this kind of attack surfaced in 2007 in the Direct Connect network with users using the DC++ file sharing application [13].
- **Network Poisoning**. Another approach towards attacking a P2P network is to inject useless data (poison) into the system. Since P2P networks must implement a lookup service in some way, an attacker can inject large amounts of useless lookup key-value pairs into the index. Bogus items in the index could slow down query times or, worse, yield invalid queries results. In fact, poisoning a P2P network has already been witnessed on the Internet as large publishing organizations attempt to lessen the potential losses of pirated media by attacking the FastTrack P2P network [14]. Poisoning can also be used as fodder for DDoS attacks. This can be accomplished in two ways, by index poisoning or route table poisoning. In index poisoning, fake records are inserted into the index pointing to a target IP and port number. When a peer goes to search for a resource, it would receive bogus location information from a poisoned index, either from a central directory or from another peer. The requesting peer then makes a connection to the target, perhaps confusion the target or, if the target accepts the connection, a TCP-connection DDoS comes into effect. In route table poisoning, the attack leverages the fact that almost all P2P clients need to maintain some kind of routing state of the current peers with which it is connected. The attacker dupes peers into adding bogus neighbors into each peer's route table, and in some cases, this as simple as making an announce message pointing to the target. The result is that the target receives a flood of connection requests, and the target will likely reject them.

Trust management is another challenging problem, defining how to enable a peer to decide whether to trust another peer, in the absence of a central trust managing authority. Trust is important when sharing data or processing power, and crucial for e-commerce applications such as auctioning. According to Gambetta’s definition [15], trust (or, symmetrically, distrust) is a particular level of the subjective probability with which a peer will perform a particular action, both before we can monitor such action (or independently of its capacity of ever to be able to monitor it) and in a context in which it affects our own action. It is important to remark that in general trust is non-transitive [16], i.e. if peer A trusts peer B and peer B trusts peer C, it does not follow that peer A trusts peer C. Moreover, as trust is subjective, peer B and peer C could have different degrees of trust in peer A. Several trust decision strategies have been proposed by the research community. A survey on them would be out of the scope of this paper, but we devote section 5 to the discussion of distributed approaches for defining the **reputation** of a peer, that is a suitable solution to support trust decision in peer-to-peer networks.

3. DESIGN STRATEGIES FOR OVERLAY SCHEMES

Current P2P systems do commonly only address a subset of the issues that are listed in section 2, for the large number of trade-offs and constraints due to the different dimensions and purposes such systems are demanded to address. In general, each P2P system is based on an *overlay scheme*, which defines how peers are connected, how messages are propagated among nodes to share resources and information about them, and which security mechanisms are adopted.

In our opinion, *the placement of information about shared resources* plays an important role in the characterization of an overlay scheme. Information about shared resources can be:

- published to a central server,
- or published to other peers,
- or locally stored by resource owners and not published.

The first approach leads to *hybrid overlay schemes* (based on the **Hybrid Model - HM**), so called because they are based on the client/server paradigm in resource publication and discovery, while the peer-to-peer approach is used for resource consumption. Centralized servers can also be used to support trust among peers, for example by playing the role of Certification Authorities (CAs) [17].

The other approaches lead to *decentralized overlay schemes*, only relying on local information available at each node (such networks are often referred as "pure" P2P systems). Decentralized P2P systems can be divided in two groups, depending on the topology awareness of peers. A decentralized P2P overlay is *unstructured* (based on the **Decentralized Unstructured Model - DUM**) if links among peers (being them actual or potential connections) can be represented by a random graph, whose characteristics are unknown to the peers, and not relevant to their message routing strategies. On the contrary, a decentralized P2P overlay is *structured* (based on the **Decentralized Structured Model - DSM**) if its topology is controlled and shaped in a way that resources (or resource advertisements) are placed at appropriate locations.

To improve the performance (with respect to scalability, lookup performance and stability) of P2P networks, *layered overlay schemes* (based on the **Layered Model - LM**) have been studied and implemented [18,19]. Such overlays are characterized by interacting layers, each one being organized according to one of the "flat" models (HM, DUM, or DSM).

Table **1** summarizes the advantages and disadvantages of HM, DUM and DSM. The success that the Hybrid Model has obtained in recent years (Napster, BitTorrent, etc.- illustrated in section 4) is clearly justified: almost all effectiveness and efficiency attributes have good values, with the exception of scalability, and many security issues are solved.

A detailed discussion is provided in the following subsections, where for each overlay scheme we define its main general characteristics, considering the strengths and flaws, then we present recent related patents.

3.1. Hybrid Overlay Schemes

In peer-to-peer systems based on the HM model, illustrated in Fig. (**1**), peers connect to one or more central servers in which they publish information about the resources they offer for sharing. Each central server maintains, for each resource, the list of owners, in some case partially replicating the lists of other known servers. Upon request from a peer, the central directory provides the list of peers that match the request, or the best peer in terms of bandwidth and availability. Further interactions occur directly between the resource provider and the consumer. If the central server is not unique, queries may also be forwarded to neighbor servers.

Matsubara, *et al*. [20] propose an HM-based P2P network in which a server manages information relating to resources that are being shared among member peers. For example, shared files that are located on the system of a peer A are listed in the management server. Likewise, shared files that are stored on the system of another peer B are listed in the management server. In this way, the management server provides directory services to the members in a P2P network of files to be shared among the peer members. The innovation, with respect to other HM systems, is that

Table 1. Contrast between Hierarchical Model (HM), Decentralized Unstructured Model (DUM) and Decentralized Structure Model (DSM). The number of nodes is N, and each node has d neighbors (it is the average value, in case of DUM). Attributes are divided in two groups: effectiveness/efficiency (green), and security (orange)

	HM	DUM	DSM
Scalability	Low	Medium	High
Bootstrapping	Simple	Complex	Simple
Connectivity Mgmt	Simple	Complex	Complex
Time Complexity	*1*	*O(N)*	*O(logN)*
Space Complexity	*1*	*O(d)*	*O(logN)*
Consistency	High	Low	High
Stability	High	High	Low
Load Balancing	No	No	Yes
Eavesdropping	No	Yes	Yes
Traffic Analysis	Yes	Yes	No
Spoofing	No	Yes	Yes
Man-in-the-middle	Yes	Yes	Yes
Playback or Replay	Yes	Yes	Yes
Local Data Alteration	Yes	Yes	Yes
No-forwarding	No	Yes	No
Free Riding	Yes	Yes	Yes
DDoS	No	Yes	No
Network Poisoning	No	Yes	No

Information of interest can be subscribed to by peer clients, and a notification server can then notify subscribing peer clients of modified information. To this purpose, a communication channel is provided between the notification server and the management server. The proposed architecture is highly suitable for sharing all forms of electronic content (e.g., documents, multimedia files, images, audio files, video files, folders, directories, and so on).

The success of Napster, eMule and BitTorrent file sharing systems (discussed in section 4) demonstrates the effectiveness of the Hybrid Model. The main reason is that HM-based peer-to-peer systems are only moderately affected by security issues. Centralized servers manage file transfer and allow more control which makes it much harder faking IP addresses, port numbers, etc. Unfortunately, for the same reason anonymity is difficult to achieve.

On the other hand, HM-based overlay schemes are absolutely not suitable for P2P streaming or gaming applications, since central servers would quickly become bottlenecks. Indeed, most servers dramatically lower their efficiency, measured in terms of responsiveness, when the

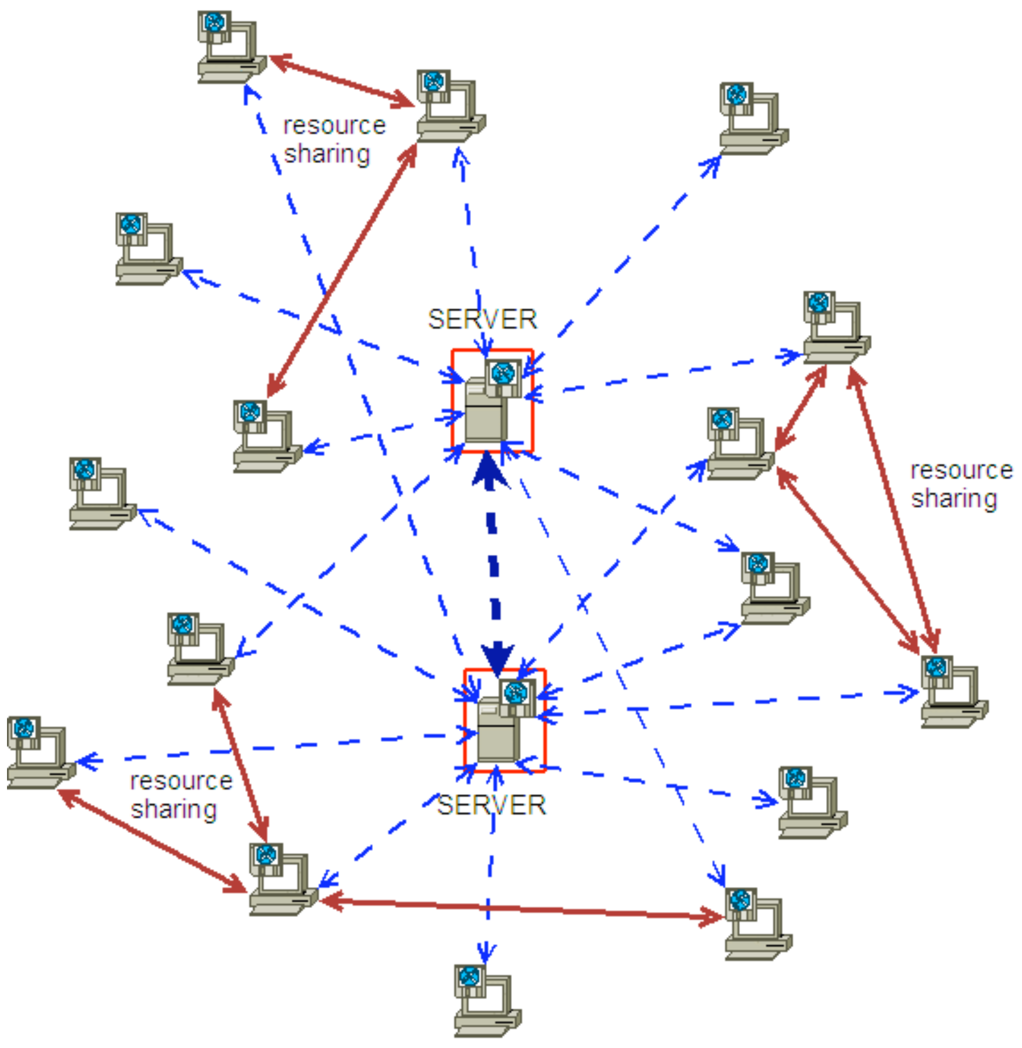


Fig. (1). Typical peer-to-peer system based on the Hybrid Model (HM). Dashed lines represent the peer-to-server and server-to-server exchange of information about shared resources. Continuous lines represent peer-to-peer resource sharing (e.g. content upload/download).

number of connected peers drastically increases (to several hundred thousands).

3.2. Decentralized Overlay Schemes

As illustrated in Fig. (**2**), in the Decentralized Unstructured Model (DUM) information about resources is distributed among peers. For applications like distributed gaming and multimedia streaming, the (DUM) is the most effective, provided that control messaging does not waste network bandwidth. Each peer propagates requests to directly connected peers, according to some strategy, e.g. by flooding the network with messages, a strategy that consumes a lot of network bandwidth, and hence does not prove to be very scalable, but it is efficient in limited communities such as company networks. To improve scalability, caching of recent research requests can be used, together with probabilistic flooding (i.e. each peer propagates messages to a random number of neighbors). Moreover, if a unique identity is added to the message, nodes can delete reoccurrences of the same message and stop loops from forming. In large networks it may be necessary to include some kind of time-to-live (TTL) counter to stop the message flooding the network.

Manion, *et al.* [21] propose generic application programming interfaces and methods that provide connectivity management in a DUM-based P2P network (for which the term “graph” is used throughout the patent). More specifically, the patent presents new and improved P2P application programming interface (API) that could be applied to implement (by subclassing) every kind of DUM-based protocol. The API supports almost all basic functionalities of a P2P system: network creation and access; node retrieval; network monitoring; addition, modification, deletion and management of data records; direct communication between nodes. Moreover, the API allows the addition of a security provider to a network, the setting and retrieval of presence information, and the registering for event notifications.

Massoulie, *et al*. [22] propose that each node in the unstructured network periodically tests logical network links among neighboring nodes to determine whether the links should be reorganized. A scheme called Metropolis is used to determine the probability with which the links are reorga-

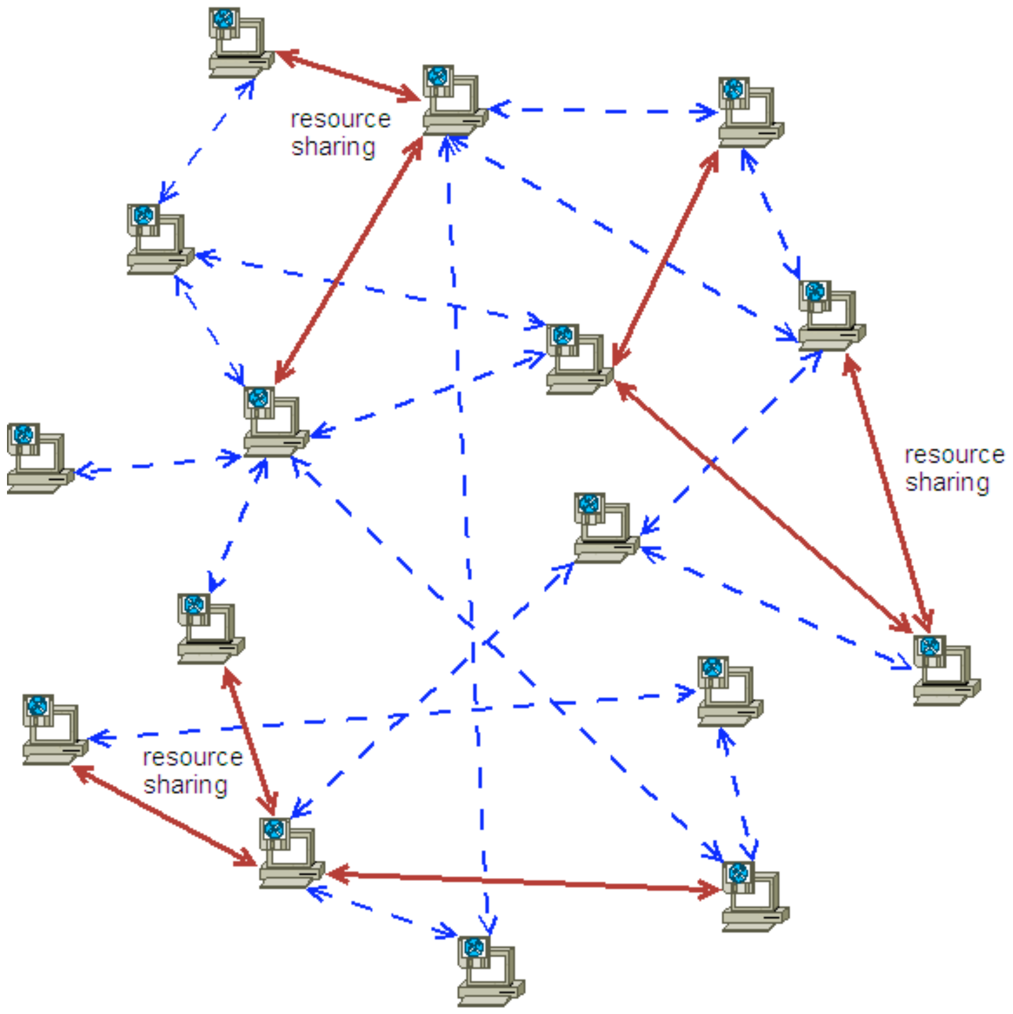
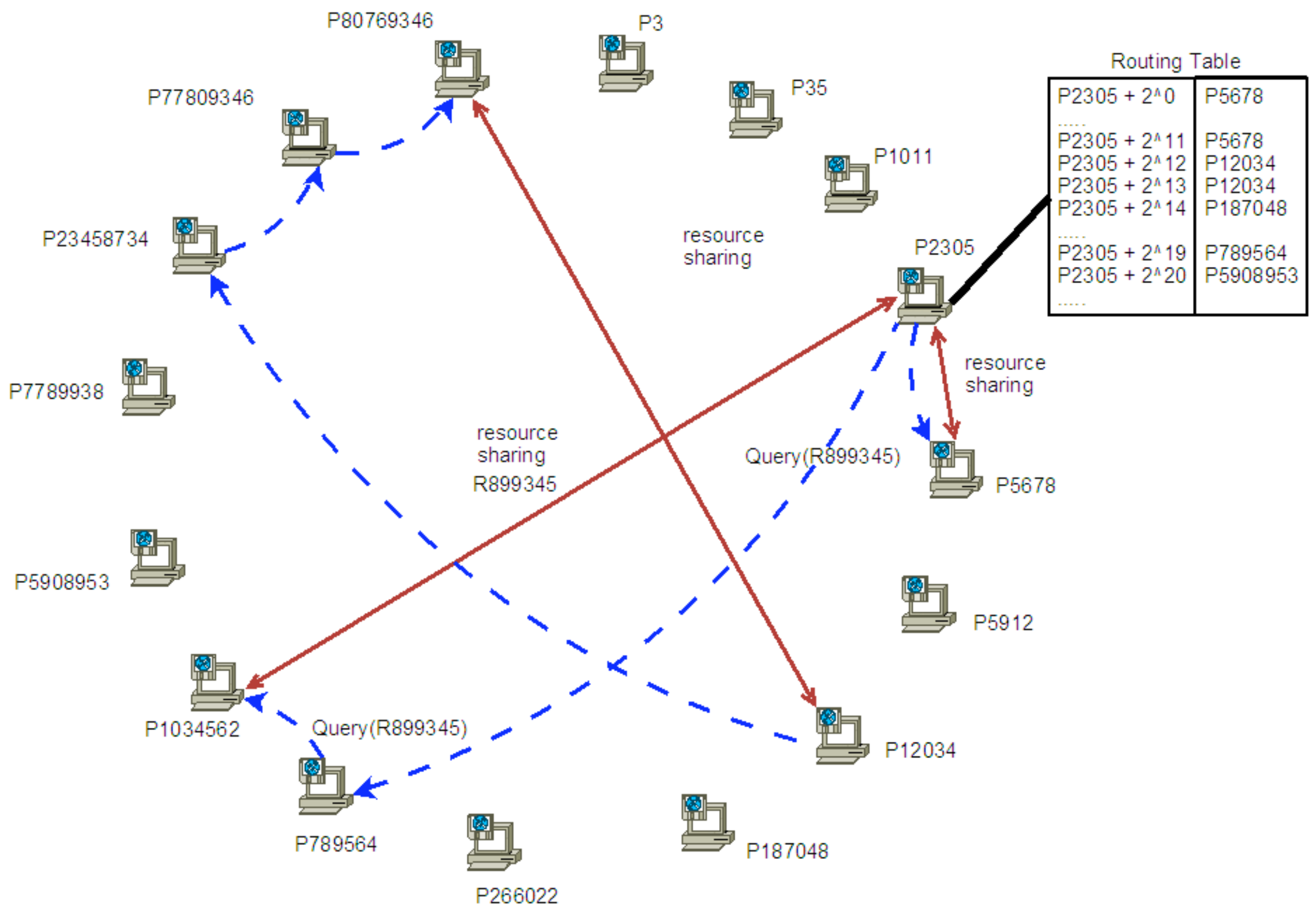


Fig. (2). Typical peer-to-peer system based on the Decentralized Unstructured Model (DUM). Dashed lines represent the peer-to-peer exchange of information about shared resources. Continuous lines represent peer-to-peer resource sharing (e.g. content upload/download).

nized. This probability is computed based on the change in link costs and/or node degrees that would be effected by a proposed reorganization. The Metropolis scheme tends to maintain a consistency among the degrees of the nodes, thereby providing strong failure resilience.

Peer-to-peer architectures based on the Decentralized Structured Model (DSM), illustrated in Fig. (**3**), are characterized by a globally consistent protocol ensuring that any node can efficiently route a search to some peer that has the desired resource, even if the resource is extremely rare. Such a guarantee necessitates a more structured pattern of overlay links. By far the most common type of structured P2P network is the distributed hash table (DHT). Each resource must be identified by a unique key and associated to a description (including a pointer to the resource owner): key and description form a <key, value> pair associated to the resource. Each peer is assigned a random ID in same space of resource keys, and it is responsible for storing <key, value> pairs for a limited subset of the entire key space. When a resource description is published, the peer routes the associated key/value pair towards the peer with the ID that is most similar to the block ID. This process is repeated until the nearest peer ID is the current peer's ID. In DSM-based overlay schemes, the responsibility of storing information about shared resources is much more fairly distributed among peers than in DUM-based ones. When a peer makes a request for a resource, the query is propagated towards the peer with the ID which is most similar to the resource ID.

Takeda, *et al.* [23] propose a method for efficiently directing broadcast messages to nodes of an overlay network. Broadcast messages include an End ID parameter specifying the range of key values for nodes that should receive the broadcast message. Each node of an overlay network maintains a list of finger nodes (neighbors) and their respective key values. In some implementations, each node is assigned a key value randomly. In some implementations, each node is assigned a key value based upon the results of a hash function of one or more attributes of the node. Upon receiving a broadcast message, a node assigns a finger node a new End ID value based upon the End ID value of the broadcast message or the key value of an adjacent finger node. The node compares a finger node's new End ID value with the finger node's key value to determine whether to forward the broadcast message to that finger node. A broadcast message forwarded to a finger node includes an End ID parameter equal to the new End ID value determined



P2305 + 2^0	P5678
.....	
P2305 + 2^11	P5678
P2305 + 2^12	P12034
P2305 + 2^13	P12034
P2305 + 2^14	P187048
.....	
P2305 + 2^19	P789564
P2305 + 2^20	P5908953
.....	

Fig. (3). Typical peer-to-peer system based on the Decentralized Structured Model (DSM)-this one in particular recalling Chord's ring. Dashed lines represent the peer-to-peer propagation of queries for shared resources. Continuous lines represent peer-to-peer resource sharing (e.g. content upload/download). Each peer owns a routing table which targets a very limited number of other peers (chosen according to a deterministic strategy). The routing table serves also for publishing information about shared resources (if the advertising strategy is fair, such information is evenly distributed among peers - on the average, all peers have the same amount of responsibility).

for the finger node. Nodes can aggregate response messages from its finger nodes. Basically, this invention defines a general SM framework, for which Chord can be considered a suitable embodiment. An interesting feature is related to preventing the spread of unauthorized broadcast messages. Upon receiving a broadcast message, a node first determines whether the broadcast message is authentic, for example by checking a cryptographic signature. If a broadcast message is authentic, it is processed and potentially forwarded to other nodes as described above. Otherwise, the broadcast message is ignored.

Another patent [24] introduces a data overlay, and particularly the SOMO tree structure, that provides an elegant and flexible mechanism for collecting or disseminating information. The mechanism is flexible in the sense that it specifies neither the type of information it should gather and/or disseminate, nor the operation invoked to process this information. In other words, SOMO operations are programmable. Further, using the abstraction of a data overlay (especially the host routing shortcut), the SOMO tree structure's performance is also insensitive to the particular implementation of the hosting DHT system. In general, both the gathering and dissemination phases are $O(\log_k N)$ bounded, where N is the total number of nodes in the P2P system. Each operation in SOMO involves no more than k+1 interactions, making it fully distributed. Further, data in the SOMO tree structure can be regenerated in $O(\log_k N)$ time. The SOMO tree self-organizes and self-heals in the same time bound.

Decentralization (both unstructured and structured) introduces new problems in terms of security and trust into these systems. Most existing approaches perform well in cooperative scenarios, but are doomed if this assumption does not hold. Also, the large number of participants in these systems raises new questions, such as whom to trust and cooperate with or how to ensure basic security properties such as confidentiality, authenticity or non-repudiation, and how to support these functionalities without introducing centralization again for being able to meet these requirements. Two basic approaches for securing decentralized P2P systems are *message encryption* and *peer anonymity*.

By encrypting P2P traffic, the hope is that not only will the data be safely encrypted, but more importantly, the P2P data stream is encrypted and not easily detectable. With the actual connection stream completely encrypted, it becomes much harder for the P2P traffic to be detected, and, thus, attacked, blocked, or throttled.

With peer anonymity, the identity of nodes and users is protected on the network, something that encryption alone cannot ensure. While true anonymity cannot really exist on a network, an anonymous P2P provides enough anonymity such that it is extremely difficult to find the source or destination of a data stream. It does this by making all peers on the network universal senders and universal receivers, thus making it practically impossible to determine if a peer is receiving a chunk of data or simply passing it through. The majority of anonymous peer-to-peer systems are "friend-to-friend" networks. These are P2P networks in which each peer (node) only connects to a small number of other nodes. Only the direct neighbors of a node know its IP address. Communication with remote nodes is provided by sending messages hop-to-hop across this overlay network. Routing messages in this way allows these networks to trade efficient routing for anonymity. There is no way to find the IP address of a remote node, and direct neighbors can achieve a level of anonymity by claiming that they are just forwarding requests and files for other nodes. In anonymous DUM-based overlay networks, the TTL counter is usually probabilistic (as determinism would allow attackers to find the sender/ receiver), e.g. the time-to-live is reduced by a value proportional to the number of results found at a node and the number of connections the search message is forwarded to.

There is a danger that an attacker will be able to spy on the activity of their direct neighbors, and thus find out which files the neighbor is requesting or offering. Some systems contain faults that leak this information while others allow an attacker to be up to 50% certain of what their neighbor is doing. None of the current systems try to make it hard for an attacker to work out whether or not someone is running the P2P software. A list of research papers on anonymity can be found at the Free Haven site [25]. Freenet [26] is probably the most famous anonymity-oriented P2P network (see section 4 for details).

In this context, a recent patent [27] addresses the limitations of existing anonymous peer-to-peer routing protocols, in which certain transmitted fields assume a relatively small subset of values. When such a subset of values is known, and the public key used to encrypt the field values is also known, the encrypted value can be deduced through trail and error comparison of encrypted values. To avoid this type of attack, the described protocol proposes an encoding mechanism that rearranges transmitted data, to effectively strengthen the encryption mechanism. Hosts that service data requests, and all intermediate hosts that route the data request, are unaware or unsure of the identification of the host that made the request. The requesting host can explicitly determine the data transfer route. Consequently, the requesting host can exclude from a data transfer route any host through which the requesting host does not wish to route data. Accordingly, data need not pass through any host that the requesting host suspects may be compromised. The described protocol enables data integrity to be checked, while maintaining user anonymity. The data request, and the data response, are routed through various intermediate hosts, and any manipulation of the data request or data response by any intermediate host can be identified. Anonymity and data integrity validation are established through the use of public and private key pairs, while error detecting codes are used to validate data integrity.

Against network poisoning, DSM protocols have a mechanism to remove stale peers from the routing table, updating it constantly. Thus, after the burst of traffic to the target, the target is removed from the route tables of connecting peers [28].

To cope with the attacks illustrated in section 2, other than encryption and peer anonymity, the security policies adopted at the overlay P2P network level usually consist of key management, authentication, admission control, and authorization (as in [29]). In the context of decentralized P2P systems, several trust management approaches have been proposed. The approach we prefer is based on the cooperation of peers in order to define the reputation of resource providers. We devote section 5 to the discussion of such solution, after the presentation of the most popular P2P overlay schemes and applications provided in section 4.

A general advantage of unstructured overlays is that they can support complex queries more effectively than structured overlays. Due to the nature of hash functions, fuzzy search algorithms required for keyword based searching are hard to implement in DHTs because a small change in the input will result in a completely different output and thus another key and not a nearby one. Thus only precise lookups, i.e. based on unique content identifiers can be performed on a DHT without large overhead.

It is also commonly believed that structured graphs are more expensive to maintain than unstructured graphs and that the constraints imposed by the structure make it harder to exploit heterogeneity to improve scalability. For these reasons, many research studies and concrete projects have targeted layered (LM-based) systems, outperforming flat architectures [30, 31].

3.3. Layered Overlay Schemes

In layered architectures, peers are grouped into layers, each one being organized according to a flat overlay scheme (HM, DUM or DSM). The interaction among layers is usually defined by an application-specific protocol. A typical example is the 2-layered architecture, in which peers with higher bandwidth and process capacity act as supernodes, assuming the responsibility of propagating messages, while peers with less capacities (leaf nodes) are only resource providers and consumers, and need to connect to the supernode layer in order to publish/discover shared resources. An example of 2-layered scheme is illustrated in Fig. (**4**).

A general-purpose 2-layered overlay scheme is the one defined in the JXTA specification [2]. In JXTA there are two primary node types:

- edge peers (leaf nodes), which are able to publish resource descriptions, and to send query/reply messages, but they do not propagate queries;

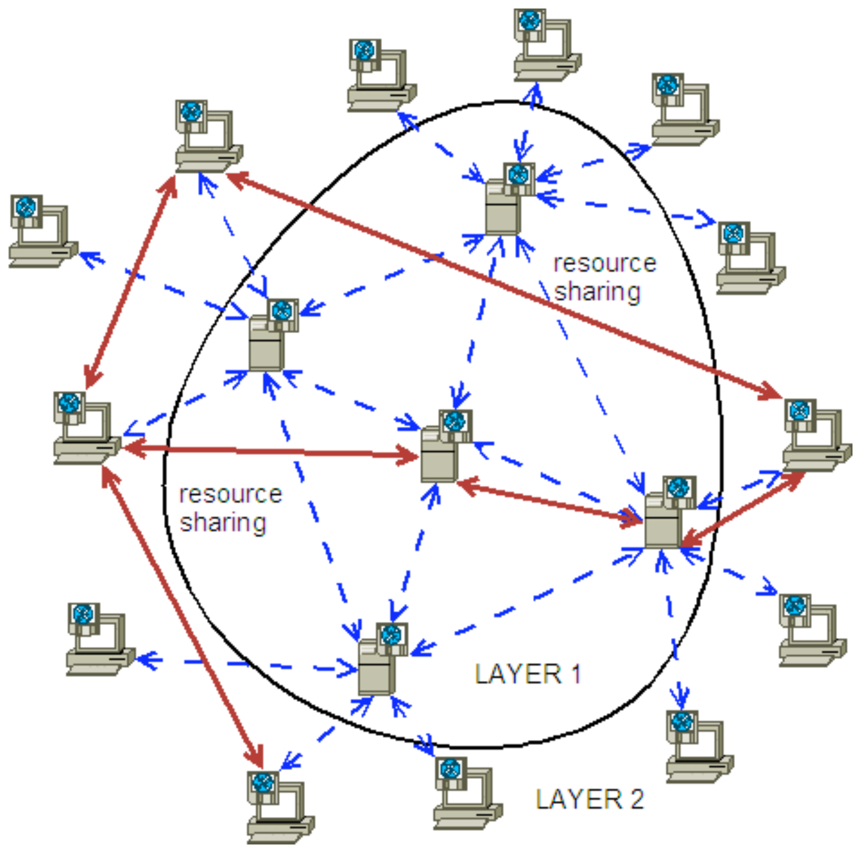


Fig. (4). Example peer-to-peer system based on a 2-layered mode. Dashed lines represent the exchange of information about shared resources - in this function, each leaf node is allowed to interact only with its supernode. Continuous lines represent peer-to-peer resource sharing (e.g. content upload/download).

- rendezvous super-peers (supernodes), which have agreed to cache pointers (indices) to edge peers that cache the corresponding advertisement (the XML description of a shareable resource), and are able to propagate queries.

Each edge peer is connected to one rendezvous super-peer. Each rendezvous has a Rendezvous Peer View (RPV), which is an ordered list of other known rendezvous. The rendezvous layer is organized according to strategy that is based on the DUM. Shareable entities (resources and services) are described by XML documents named advertisements, whose entries are attribute/value pairs. Message routing for sharing and searching advertisements in the JXTA rendezvous network is based on two components: the Shared Resource Distributed Index (SRDI), and the walker. The SRDI module implemented in each JXTA peer is used on one hand to extract entries from advertisements and push them to the network, and on the other hand for lookups. The walker is used for routing when no index information is locally available. The more the supernode network is near to completeness, i.e. the RPV is consistent across all supernodes, the more the routing algorithm is efficient. The rendezvous approach has been described by JXTA designers in patent [32].

Another solution which can be put under the LM umbrella is the HALO scheme [33], which is based on high-degree supernode search, i.e. messages are routed choosing at each step the highest-degree neighbor. The idea for this strategy comes from the observation that the strategy adopted by JXTA gives its best when the supernode network is highly connected. HALO routes messages towards best connected nodes, and uses the same selection function used by JXTA if neighbors are less connected than current peer, and for the final step. If many neighbors have the same highest node degree, the target is chosen by proximity of its ID with the message ID.

Yap proposes a method [34] for caching data in a layered peer-to-peer system comprising the steps of: (a) establishing a performance criterion; (b) arranging the layered peer-to-peer system as a plurality of interconnected hierarchical groups of peers by (c) assigning each of the plurality of peers to at least one group as a first function of the performance criterion; (d) processing the data into a plurality of hierarchical data partitions; (e) allocating, as a second function of the performance criterion, each said data partition to at least one of the groups; and (f) caching each data partition in at least one peer in the corresponding group.

4. POPULAR P2P OVERLAY SCHEMES

Over the last few years, peer-to-peer content sharing systems have enjoyed explosive popularity. Thus, most research efforts have been spent on designing effective, efficient and secure protocols for such kinds of applications. Nevertheless, we must observe that the basic principles of

HM, DUM, DSM and LM can be applied to any other kind of distributed application involving peer-to-peer resource sharing, i.e. cooperative advertising, discovery and consumption of resources. In the following we discuss the most popular overlay schemes, grouped by application domain. Cross-domain schemes are highlighted.

4.1. Content Sharing

The content sharing problem is well explained in a recent patent that defines a method for sharing files between a group of computer systems [35]. The files shared by a group are associated with a group folder. A group folder is represented by a folder created by the file system of each member of the group. The folder at each member contains a file for each file that is shared by the group. The files in the folder of a member may be actual files stored at that member, which contains the content of the shared file, or virtual files identifying the actual file that is stored at another member. When a member accesses a virtual file, the file sharing system detects the access and requests that the file owner provide a copy of the file to the accessing member on a peer-to-peer basis. Whenever a shared file is modified, the file owner sends updated metadata for that file to the other members of the group.

In this context, **Soulseek** [36] is a file-sharing network and application based on the HM scheme. It is used mostly to exchange music, although users are able to share a variety of files. The central server coordinates searches and hosts chat rooms, but it does not actually plays a part in the transfer of files, which takes place directly between the concerned users. Being one of the first HM protocols, it has many limitations with respect to others. In particular, it does not allow multisource download of files.

Also based on the HM scheme, the **Napster** [37] protocol was designed with the same purposes of Soulseek, but it gained much more success because of several free implementations, both open source (gnapster, Knapster, etc.) and closed source (the official Napster client), and also several related utilities. Residing on user machines, files never pass through central Napster servers, which provide the ability to search for particular files and initiate a direct transfer between peers. OpenNap [38] extends the Napster protocol to allow sharing of any media type, and the ability to link servers together.

The **eDonkey** protocol (also known as eDonkey2000 or eD2k), HM-based, allows to create file sharing networks for the exchange of multimedia content. Client programs connect to the network to share files, and publish to servers that can be set up by anyone. Servers act as communication hubs for the clients and allow users to locate files within the network. There are many programs that act as the client part of the network. Most notably, eDonkey2000, the original client by MetaMachine, closed-source but freeware, and no longer maintained but very popular in its day; and eMule [39], a free program for Windows written in Visual C++ and licensed under the GNU GPL. eMule is in constant development and currently represents about 90% of all clients in the eD2k network. In eDonkey, free riders are penalized by the following mechanism. A client ID is provided to a peer by the contacted server at their connection handshake, and it is valid only through the lifetime of a peer-server TCP connection, although in case the peer has a high ID it will be assigned the same ID by all servers until its IP address changes. Identifiers are divided to low IDs and high IDs. Each eMule servers typically assigns a peer with a low ID when the peer can't accept incoming connections. Having a low ID restricts the peer's use of the eMule network, since a low ID peer has no public IP to which other peers can connect to, thus all communication must be done through the eMule servers. This increases the server's computational overhead and results in reluctance of servers to accept low ID peers.

Currently the most popular HM-based systems are those based on the **BitTorrent** scheme [8], for which central servers store information about trackers, that are responsible for helping peers find each other. The BitTorrent protocol focuses on high data transfer speed rather than on search capabilities. Currently BitTorrent is the largest file sharing network (30% of the global peer-to-peer traffic, and 15% of the overall Internet traffic). A specificity of BitTorrent is the notion of torrent, which defines a session of transfer of a single content to a set of peers. A peer joins an existing torrent by downloading a related .torrent file from one of the torrent Web server. The .torrent file comes to the leecher along with the IP address of the tracker of the torrent. When joining the torrent, the peer asks to the tracker a list of IP address of peers to build its initial peer set, i.e. the list of other peers it knows about. A peer can only send data to a subset of its peer set, called active peer set. The "choke algorithm" determines the peers being part of the active peer set. Files transferred using BitTorrent are split into pieces (of 256KB), and each piece is split into blocks (of 16KB). Blocks are the transmission units on the network, but the protocol only accounts for transferred pieces. Each downloader reports to all its peers what pieces it has, thus each peer knows the distribution of the pieces for each peer in its peer set. The piece selection strategy used in BitTorrent is based on rarest first strategy. BitTorrent encrypts both the header and the payload, for each transferred piece. Using only 60-80 bits for the cipher, the aim is not to protect the data but instead to simply obfuscate the stream enough so that it is not detectable without incurring much of a performance hit. Although it is still possible to detect encrypted BitTorrent streams using sophisticated methods based on pattern and timing of the traffic, in practice, it is much harder to filter encrypted streams now. Encryption of P2P traffic seems to be picking up, as currently about 20% of BitTorrent traffic is encrypted [9].

The **Gnutella** DUM-based protocol [40] was published in late 1999 by Nullsoft (inventor of WinAmp media player), and now it is at the 4th release. The protocol defines the way in which peers communicate over the Gnutella overlay network, that is through a set of descriptors whose inter-peer exchange is governed by a set of rules. The messages allowed in the Gnutella network can be grouped as follows: Group Membership (PING and PONG descriptors, for peer discovery queries/replies), Search (QUERY and QUERY HIT descriptors, for file discovery queries/replies), and File Transfer (GET and PUSH descriptors, for file exchange between peers). Every PING or QUERY message received by a node is forwarded to all the neighbors of the node (flooding). The specification makes no recommendations as to the frequency at which a peer should send PING descriptors, although peer developers should make every attempt to minimize PING traffic on the network. To avoid network congestion, the PING and QUERY messages are always associated to a Time To Live (TTL), which is the max number of times the descriptor can be forwarded before it is removed from the network. The number of times the descriptor has been forwarded is named Hops.

TTL(0) = TTL(i) + Hops(i)

PONG and QUERY HIT descriptors may only be sent along the same path that carried the incoming PING and QUERY messages. A peer receiving a descriptor with the same Payload Descriptor and Descriptor ID as one it has received before, should attempt to avoid forwarding the descriptor to any connected peer. Many peer applications based on the Gnutella protocol are available for Windows, Linux and Mac OS X; the most famous is LimeWire [41].

The **Freenet** DUM-based P2P scheme [26] was conceptualized by Clarke in 1999, and public development of the open source reference implementation began in early 2000. Freenet is a distributed information storage system which represents a prime example of how anonymity can be built into a P2P application. In designing Freenet, the authors focused on: privacy for information producers, consumers and holders; resistance to information censorship; high availability and reliability through decentralization; and efficient, scalable, and adaptive storage routing. Freenet uses a forwarding scheme for messages to ensure that the original requestor of a service cannot be tracked. It increases anonymity by using probabilistic algorithms so that origins cannot be easily tracked by analyzing network traffic. According to Kleinberg [42], Freenet should find data in around $O(\log_2 n)$ hops, if peers are connected in a small-world network. However, this is only true on a mature Freenet network, and the typical churn rate, due to nodes not running continually and people trying it out and then leaving, may prevent it. Apart from this, the price to pay for anonymity, i.e. forwarding data chunks through several peers rather than directly from source to destination, is a high transfer overhead which can be accepted only for a limited set of applications.

In academics, **Distributed K-ary Search (DKS)** [43] is probably the most known DSM-based protocol, including Chord [44] and Pastry [45]. Each instance of DKS is a fully decentralized overlay network characterized by three parameters: N the maximum number of nodes that can be in the network, k the search arity within the network, and f the degree of fault tolerance. Once these parameters are instantiated, the resulting network has several desirable properties, e.g. each lookup request is resolved in at most $\log_k(N)$ overlay hops. Each node has to maintain only $(k - 1) \log_k(N) + 1$ addresses of other nodes for routing purposes.

Kademlia [46] is another famous DSM-based protocol, used for example in recent versions of the eMule client (as an alternative to the traditional eDonkey protocol). Many of Kademlia’s benefits result from its use of the XOR metric for distance between points in the key space. XOR is symmetric, allowing Kademlia participants to receive lookup queries from precisely the same distribution of nodes contained in their routing tables. To locate nodes near a particular ID, Kademlia uses a single routing algorithm from start to finish. In contrast, other systems use one algorithm to get near the target ID and another for the last few hops. Kademlia furthermore introduces a concurrency parameter, α, that lets people trade a constant factor in bandwidth for asynchronous lowest latency hop selection and delay-free fault recovery. Finally, Kademlia is the first peer-to-peer system to exploit the fact that node failures are inversely related to uptime.

FastTrack [47] is the LM-based protocol used by file sharing applications such as KaZaA Media Desktop (shortly, KaZaA) [48] and iMesh [49]. Moreover, it is used by Skype [30], which is a VoIP system (see section 4.5). FastTrack extends the Gnutella protocol with supernodes, to improve scalability. Supernodes act as temporary indexing servers and help support the stability of the network. These supernodes stay outside the control of any company. A peer hosted by a powerful computer with a fast network connection automatically becomes a supernode, effectively acting as a temporary indexing server for other, slower leaf peers. In order to be able to initially connect to the network, a list of supernode IP numbers is hardcoded in the peer. As soon as a peer finds a working supernode, it requests a list of currently active supernodes, to be used for future connection attempts. Each leaf peer uploads a list of files it intends to share to its supernode. It also sends search requests to this supernode. Supernodes communicate between each others in order to satisfy search requests. Peers are directly connected to exchange files; this transfer is based on the HTTP protocol. Programmers from the open source community have reverse engineered the portion of the protocol dealing with leaf-to-supernode communication, but the supernode-to-supernode communication protocol remains largely unknown.

The **Gnutella2** [50] network is a self-organizing collection of interconnected nodes cooperating to enable productive distributed activities. According to the Layered Model, not all of the nodes participating in the system are equal: there are two primary node types, ”hubs” and ”leaves”. Hub nodes devote substantial resources to the network, and as a result their capacity to participate in higher level network functions is limited. Only the most capable nodes are selected to act as hubs. Nodes operating as Gnutella2 hubs have a set of responsibilities to meet. Hubs are highly interconnected, forming a ”hub network” or ”hub web”, with each hub maintaining a connection to 5-30 other ”neighboring” hubs. The number of hub interconnections must scale up with the overall size of the network. Each hub also accepts connections from a large collection of leaf nodes, typically 200-300 depending on available resources. Leaf nodes are considered to be the ”edge” of the network. In practice, leaves simultaneously connect to two hubs, however, from the point of view of the hubs each leaf is considered a dead end. The goal is to maximize the number of leaves and minimize the number of hubs, however, due to the limited nature of resources the maximum viable ratio of leaves to hubs is limited. This quantity is known as the ”leaf density”. The Gnutella2 resource search mechanism can be described as an ”iterative crawl” of the network, with a series of important optimizations derived from the network topology and components. A search peer iteratively contacts known hubs, that match the query against the cached hash tables of their neighbors. If a table hit occurs, the query is forwarded once only. Nodes which actually receive the filtered query process it and send results to the search peer directly.

4.2. Distributed Storage

Tedeschi, *et al*. [51] propose a "policy architecture" to develop distributed storage systems, which divides the control and data plane, where the control plane embodies the design policy of the system over a data plane that provides a set of common mechanisms. Using such an approach, the authors build a distributed storage systems spanning a large portion of the design space. A popular distributed storage systems based on the Layered Model (LM) is Wuala [52], which uses a Chord-like overlay scheme to organize a group of supernodes, each one managing a cluster of storage nodes.

In the same context, Lv, *et al*. [53] experimentally showed that replication improves the search performance of DUM-based P2P networks. To optimize network-wide search performance given limited storage capacity, more replicas are preferred for more frequently accessed objects. Cohen and Shenker [54] proved that the search time and traffic under random walk search is minimized when the number of replicas for each object is proportional to the square root of its query rate. Tewari and Kleinrock [55] showed that under controlled flooding search, the search traffic is minimized under the same square-root replica distribution, whereas the search time is minimized when the number of replicas for each object is linearly proportional to its query rate. Kangasharju, *et al*. [56] developed a logarithmic replication distribution to maximize the content availability under intermittent connectivity. However, none of the above work has considered keeping the replicas consistent with the authoritative contents. In general, there are two classes of methods to maintain consistency: push-based and pull-based. In push-based methods, the content owners keep track of the replica locations and send invalidation messages or updated contents to the replicas whenever the contents are modified. In contrast, pull-based methods are replica-driven. The replicas, when considered outdated, are validated before serving new requests. Tang *et al*. [40] propose to assign each replica an expiration time (time-to-live, TTL) beyond which the replica stops serving new requests unless it is validated. While TTL-based consistency is widely explored in many client-server applications, there has been no study on TTL-based consistency in P2P networks. The main contribution of [57] is an analytical model that studies the search performance and the freshness of P2P content sharing under TTL-based consistency. Due to the random nature of request routing, P2P networks are fundamentally different from most existing TTL-based systems in that every node with a valid replica has the potential to serve any other node. The authors identify and discuss the factors that affect the performance of P2P content sharing under TTL-based consistency. Simulation results indicate a trade-off between search performance and freshness: the search cost sublinearly falls off with decreasing freshness of P2P content sharing.

Another method for maintaining consistency of a shared space across multiple endpoints in a peer-to-peer collaborative computer system has been proposed by Richardson, *et al*. [58]. Deltas containing data change commands are organized in a persistent data structure called a delta log, owned by each peer. The delta log is organized into blocks, which are the largest division in the delta log. In turn, blocks contain groups, groups contain chains and chains contain deltas. Delta blocks are used to implement priority deltas that are used to limit the collection of data change commands that must be transferred. Within a block the deltas are organized by groups, each of which is a set of deltas organized into chains. The delta group is used to determine which deltas to purge. The chains are ordered by increasing creator identifier of the endpoint that created the chain. Organizing the delta log in this fashion allows the log to be "walked" to detect convergence problems. To achieve causality-preservation, each delta has a list of dependencies representing other deltas that must be executed before the current delta can be executed. The dynamics manager uses the ability to do (execute) and undo commands to perform roll back and roll forward operations on deltas in order to achieve convergence.

4.3. Parallel & Distributed Computing

Parallel and distributed applications based on the P2P paradigm split a large task into smaller sub-pieces that can execute in parallel over a number of independent peer nodes (componentized applications). Most often, the same task is performed on each peer using a different set of parameters (compute-intensive applications). Example tasks are code breaking, market evaluation, scientific simulations.

Recently, the Grid computing research community started using the P2P paradigm for accomplishing efficient resource discovery, which is one of the fundamental requirements of Grid systems [59]. Resource discovery involves searching for resources that match the user's application requirements. Various kinds of solutions to Grid resource discovery have been developed, including the centralized and hierarchical information server approach. However, these approaches have serious limitations in regards to scalability, fault-tolerance and network congestion, with respect to solutions based on P2P. An almost complete survey on peer-to-peer resource sharing models for Grid systems is proposed in a recent work of Trunfio, *et al*. [60].

P2P networks have also begun to attract attention from scientists in other disciplines, especially those that deal with large datasets such as bioinformatics. In such context, P2P networks can be used for example to run programs designed to identify known and new protein sequences using high-throughput methods [61]. A P2P network provides methods for accessing distributed resources with minimal maintenance cost. It also provides scalable techniques to search through large amounts of resources scattered through the network. Furthermore, joining or leaving the network becomes a simple task. These properties of P2P networks make the technology an ideal candidate to implement search through proteomics laboratories. A proteomics laboratory acting as a peer in a P2P network will share its complete or partial data repository so that other peers and itself can benefit from it.

Dewey [62] provides a method to allocate tasks among a plurality of processes within a distributed processing environment. Each process secures a list of elements and performs a first operation on an element of the list to produce a result corresponding to one of the processes. If the result corresponds to the process processing the element, the process performs a second operation on the element. Each process then processes the next element on the list until all the elements of the list are processed. After the process processes all the elements on the list it further processes each element remaining on the list and performs the second operation on the elements remaining on the list. The distributed processing environment can be for example a peer-to-peer virtual storage system and each process runs on a controller controlling a plurality of storage systems.

4.4. Business

P2P networks have already been used in business areas, but it is still in the beginning stages. Besides file sharing, companies are also interested in distributing computing, content distribution, e-marketplace, distributed search engines, groupware and office automation via P2P networks. At the same time, P2P is not fully used as it still faces a lot of security issues (see section 3 for a detailed discussion).

Among others, Microsoft Office Groove [63] is a desktop application designed for document collaboration in teams with members who are regularly off-line or who do not share the same network security clearance. Groove was initially developed by Groove Networks, until Microsoft's acquisition in March 2005. Groove's core concept is the shared workspace, which consists of a set of files to be shared, plus some aids for group collaboration. Groove users can create workspaces, add documents, and invite other Groove members to a workspace. A user that responds to an invitation is made an active member of that workspace. Each member has privately editable copy of the workspace. Users interact and collaborate in the common workspace which is a private virtual location. All changes are tracked by Groove, sent to all members and all copies of the workspace are synchronized via the network in a peer-to-peer manner. When participating users are off-line, changes for their workspaces copies are queued, either on a Office Groove Server that mediates the workspace or via other participants (in a peer-to-peer fashion), to be sent to users when they come on-line. When multiple users edit one document at the same time, changes may conflict and multiple versions will be shown until an editor decides which changes will become final.

Chen, *et al*. [64] propose a collaborative business process for modeling inter-enterprise collaboration - e.g., peer-to-peer (P2P) or business-to-business (B2B) interaction - that involves at least two players from two different enterprises is defined. The collaborative business process has a plurality of work nodes. Each work node has a task-role identifier for identifying a particular player to execute each node. A first collaborative process manager (FCPM) associated with the first player is provided to execute a first instance of the collaborative business process. A second collaborative process manager (SCPM) associated with the second player is provided to execute a second instance of the collaborative business process.

4.5. VoIP & Multimedia Streaming

Nowadays, traditional telephony is being flanked by Voice over IP (VoIP), which is a family of transmission technologies for delivery of voice communications over the Internet or other packet-switched networks. Skype, one of the most widely used Internet phone applications, is based on the FastTrack protocol (originally developed for the KaZaA file sharing network) [30] that we illustrated in section 4.1. Furthermore, many research organizations are trying to apply P2P networks to cellular networks.

Another kind of Internet-scale multimedia application which is getting enormous success is live P2P audio/video streaming (P2PTV), which is usually based on the Decentralized Unstructured Model. TVU is an Internet TV broadcasting network that uses P2PTV technology to offer its broadcasters global reach and low costs [65]. Joost is another known P2PTV Internet application [66]. The peer-to-peer layer comes from the Joltid company, which consists of the original management and development team behind KaZaA and the FastTrack peer-to-peer network, and also provided the peer-to-peer layer of Skype. TVants is a P2PTV software, whose core technology is based mainly on the BitTorrent protocol [8], for which a single peer playing the stream exchanges (sends or receives) streaming data packets with a few peers, in order to maximize the bandwidth usage and obtain the most optimal streaming quality [67]. The TVants peer allows users to watch 600 different channels from China or foreign countries, and to share their hard disk video local files with anyone online with broadband connection. LiveStation is a P2PTV platform being developed by Skinkers Ltd. based on peer-to-peer technology acquired from Microsoft Research [68]. Social networking features have been added that include the ability to chat with other viewers and also find out what others are watching through a user generated rating system.

Since P2PTV targets a lot of people, it needs group communication functionalities to be implemented in the hosts, on the edge of the network. Protocols for live P2P streaming application can be classified in three categories:

- source-driven
- receiver-driven
- data-driven

In the source-driven approach, there are a data plan, which is used to send data to all peers, and a control plan, which is used to manage the group and dynamicity of peers (arrivals and departures). With Peercast [69], the control plan is a tree whose root is the source of the stream, and the data plan uses the same tree. Other control plans like that of ZigZag [70] defines hierarchical organization of peers into a cluster, with the data plan being again a tree built on top of the control plan.

With receiver-driven approach, control plan and data plan are clearly separated similarly to source-driven approach, and control plan can be a tree, a cluster or a mesh. Conversely, data plan is a tree and it is rooted at the receiver side instead of the source side. receiver-driven approach is usually related to data encoding like layered coding or multiple description coding as in Peerstreaming [71]. In [72], a "PeerStreamer" provides receiver-driven peer-to-peer media streaming for loosely coupled P2P networks. Peers in the network perform only simple operations, may cache all or part of the streaming media, do not collaborate with other peers, may be unreliable, and may drop offline or come online during any given streaming session. Clients in the network operate in real-time to coordinate peers, stream media from multiple peers, perform load balancing, handle online/offline states of peers, and perform decoding and rendering the streaming media. The PeerStreamer uses high rate erasure resilient coding to allow multiple serving peers to hold partial media without conflict, such that clients simply retrieve fixed numbers of erasure coded blocks regardless of where and what specific blocks are retrieved. Alternatively, the PeerStreamer uses embedded coded media to vary streaming bitrates according to available serving bandwidths and client queue status.

Unlike other solutions, the data-driven approach does not clearly separate control plan and data plan. Group members (peers) exchange control messages about data availability in the network. Each peer chooses its neighborhood according to the data it needs, searching for example by means of epidemic algorithms [73], like in CoolStreaming [74], PULSE [75] and PRIME [76].

4.6. Gaming

Massively multiplayer games (MMGs) are becoming popular nowadays. In MMGs, a huge number of players are in the same virtual world, where they play roles, move and interact with each other and with other objects in the world. To overcome the limited scalability and robustness problems inherent to centralized solutions, P2P overlay networks are emerging as a promising architecture for MMGs [77]. A P2P overlay network can be easily applicable to MMGs since players are likely to contribute their communication and computation resources for interesting game-play.

The authors in [78] describe interactive action games in which bandwidth demand is reduced by estimating what players are paying attention to, and the resource and interest heterogeneity are managed by disseminating updates via a multicast system designed for the special requirements of games.

The authors of [79] define a very generic gaming system that includes at least two gaming components, each one including a controller and a communications interface. The gaming system also includes a communication link to allow the controllers of the gaming components to communicate with other controllers of other gaming components on a peer-to-peer basis through the communication interfaces.

Examples of commercial MMG based on P2P are Rakion [80], Gunbound [81], Rumble Fighter [82] and Soldier Front [83]. All of these games have the players connect to each other, forming a Dum-based or DSM-based overlay network, rather than connecting to one server.

4.7. Education & Academia

Due to the fast distribution and large storage space features, many organizations are trying to apply P2P networks for educational and academic purposes. For instance, Pennsylvania State University, MIT and Simon Fraser University are carrying on a project called LionShare designed for facilitating file sharing among educational institutions globally [84]. The LionShare project is dedicated to harnessing the promise of P2P file sharing and the integration of P2P with organizational services to create a collaborative environment for use in academic communities. The LionShare Peer application is built around the themes of collaboration, security, personal responsibility, and access control of shared resources, along with access to large digital repositories. LionShare Peers operate within a P2P network using code originating from the Limewire 4.0 open source project. Limewire uses the Gnutella protocol (illustrated in section 2) for the basic search and retrieval functions.

Edutella is a peer-to-peer network for exchanging information about learning objects, rather than files [85]. It is built with semantic web technology applying the latest P2P research. Building blocks of Edutella are JXTA, a general-purpose P2P framework (see section 1.4), and RDF, which is a W3C framework for representing resources in the Web.

4.8. Ambient Intelligence

The concept of ambient intelligence (AmI), which refers to a digital environment that proactively supports people in their daily lives, was introduced by the information Society Technologies Advisory Group (ISTAG) of the European Commission [86]. AmI overlaps with other concepts, such as ubiquitous computing, pervasive computing, context awareness, embedded systems and artificial intelligence [87].

AmI systems require a light and flexible middleware layer, providing facilities for building both client/server and peer-to-peer applications, supporting standard data and service descriptions, and allowing run-time component pluggability. A state-of-art example is PERSONA [88], aiming at advancing the AmI paradigm through the harmonization of Ambient Assisted Living (AAL) technologies and concepts for the development of sustainable and affordable solutions for the social inclusion and independent living of senior citizens, integrated in a common semantic framework. Project PERSONA seeks to develop a scalable open standard technological platform to build a broad range of AAL services, which are the services that the end user perceives as the final services and what he pays for. Each AAL service is composed by a hardware infrastructure and software components together with a user interface to communicate with the user. Software components, interacting in a peer-to-peer (DUM-based) fashion, may be publishers/consumers of context events, or providers/requestors of services.

5. REPUTATION MANAGEMENT

Several strategies have been proposed in the context of reputation management in P2P networks (e.g. [89]), most of them referring to content sharing networks. We propose a classification with three categories of models, and we discuss their benefits and drawbacks.

5.1. Local Evaluation

The simplest approach is the Local Evaluation model, for which after each transaction the consumer evaluates the quality of the retrieved resource/service and updates the local reputation value of the provider. In this model, reputation and trust are coinciding concepts.

For example, in [68] the reputation/trust of a peer computed by another peer is the number of satisfactory transactions versus the total number of transactions. This model is very straightforward to implement, and it does not flood the network with messages, since reputation information is not exchanged among peers. This is also the main drawback of the Local Evaluation approach which is almost inapplicable when the network becomes very large and dynamic, as, in this case, it is highly probable that discovered services come from unknown (i.e. untrusted) peers.

Gupta, *et al*. [90] propose a security infrastructure and methods that inhibit the ability of a malicious node from disrupting the normal operations of a peer-to-peer network. The methods of the invention allow both secure and insecure identities to be used by nodes by making them self-verifying. When necessary or opportunistic, ID ownership is validated by piggybacking the validation on existing messages. The probability of connecting initially to a malicious node is reduced by randomly selecting to which node to connect. Further, information from malicious nodes is identified and can be disregarded by maintaining information about prior communications that will require a future response. Denial of service attacks are inhibited by allowing the node to disregard requests when its resource utilization exceeds a predetermined limit. The ability for a malicious node to remove a valid node is reduced by requiring that revocation certificates be signed by the node to be removed.

5.2. Voting

In a system based on the Voting model [91-93], the consumer chooses the best provider according to its own previous experiences and those provided by other peers. The main advantage of this approach is that each peer can rely on a distributed knowledge base.

In particular EigenTrust [92] can be considered as a touchstone, among Voting-based solutions. The main objective of EigenTrust is the definition of a global trust value for each peer, i.e. an absolute value of trust that the whole system assigns to each peer. Local trust values are aggregated based on transitive trust, i.e. considering that if a peer provides a good service, then probably it also provides good advices. In our opinion this assumption appears too strong; as suggested in [80], transitive trust should be conditional, rather than assumed a priori. On the other side, the EigenTrust algorithm has many interesting features, such as self-policing, anonymity preservation, overhead minimization. Moreover, EigenTrust is robust against malicious collectives, i.e. groups of peers which provide corrupted services and agree on support each others with lying assessments. EigenTrust is meant to be used to decrease the number of downloads of inauthentic resources in peer-to-peer file sharing networks.

A more recent work is [94], whose authors propose to use fuzzy techniques in the design of reputation systems based on collecting and aggregating peers' opinions. Fuzzy techniques are used in the evaluation and synthesis of all the opinions expressed by peers. The behavior of the proposed system is described by comparison with probabilistic approaches.

Traversat, *et al*. [95] propose a system and method for providing peer groups in a peer-to-peer environment. Peers may create new peer groups, may discover existing peer groups and join the existing peer groups. In the proposed system, when a peer applies to a peer group, other members may verify the application credential by voting.

An unconventional voting solution is proposed in [96], where a peer node may launch a mobile agent on a network including an itinerary of peer nodes to be visited by the mobile agent and an indication of an area of interest. The mobile agent may determine if a visited peer node stores trust evaluations for other peer nodes as providers of contents and data (codats) relevant to the area of interest and, if so, the trust evaluations may be stored as payload data in the mobile agent. After completing the itinerary, the mobile agent may return the payload to the initiating peer node. The initiating peer node may use the trust evaluations collected by the mobile agent in determining or adjusting trust evaluations for peer nodes as providers of codats relevant to the area of interest.

5.3. Transaction Certificates

The third possible strategy is to use Transaction Certificates [97], generated by involved parties and including the score assigned to just completed transactions. At the end of a successful service completion, the client signs a satisfaction certificate (a form of digital certificate) and gives it to the server. The satisfaction certificate is a proof that the server provided a good service to the client. The server on receiving the satisfaction certificate sends it to the nodes known to it. On the other hand, if a server provides a bad service, then that information is readily propagated, first by a client to the nodes known to it and then recursively to other nodes known to each of them.

When searching for the best provider, networked peers can retrieve and compare transaction certificates in order to assign reputation values to possible providers. The advantage of this approach is that nodes are encouraged in providing good resources since they obtain an immediate reward (in fact, reputation can be considered as ”money”). Moreover, malicious collectives are ineffective against this model. The main drawback is that when a peer leaves a group, spontaneously or forcedly, a new affiliation certificate must be generated and shared among other members.

Yeager, *et al*. [98] propose a decentralized, distributed trust mechanism that may be used in peer-to-peer platforms, to implement trust relationships based on data relevance between peers on a network and to implement trust relationships between peers and content and data (codat). The trust mechanism may provide a trust spectrum of multiple levels wherein unique peer identities may be established to enable authentication and the assignment of the peers' associated access policies within a peer group. The trust spectrum may have Certificate Authority signed certificates as a maximum level of security, and self-signed certificates as a minimum level of security. Since a certificate is one form of codat, the trust mechanism may be applied to a peer group member's collection of signed certificates for a given peer group.

The same authors propose mechanisms for representing trust between peers or systems in decentralized networking environments including peer-to-peer networking environments [99]. Trust may include both direct trust between two peers and trust in a pipeline of peers along which codat may be passed. Implementations may provide a mechanism for a peer to represent and rate the trustworthiness of other peers as providers of codat relevant to the peer's interest. To evaluate trust in another peer as a provider of codats in the area of interest, trust may be represented with two components, confidence and risk. Implementations may provide mechanisms for measuring the components and determining trust from the components, and may also provide mechanisms for feeding back trust information to the providing peer and for propagating trust information to other peers.

5.4. Discussion

We already pointed that the Local Evaluation model is easy to implement but of little use if the set of providers shows high dynamicity. Consequently, we focus here on the Voting and Transaction Certificates models, which both rely on information exchange among peers.

The first difference is that in the Voting model advices are generated on demand and their motivation cannot be verified. Conversely, in the Transaction Certificates scheme each advice is clearly and objectively tracked, for which it is possible to check for the trustworthiness of the issuer and reduce the influence of malicious collectives.

With respect to overhead, in the Voting scheme there is only one kind of message exchange, the one in which consumers collect advices about providers offering resources they are interested in. Interactions among peers are more complex in the Transaction Certificates model, and certificates may flood the network if propagation constraints are not set.

Concerning vulnerability, both schemes are resistant to one shot attacks, in which a malicious peer acts unfairly and immediately leaves the network. The Voting model is vulnerable to malicious collectives, where sets of malicious peers collaborate in an attempt to subvert the system. In particular, this situation is dangerous if malicious peers frequently change their identity. The Transaction Certificates model can be subject to many kinds of attack, ranging from DoS attacks against CAs, to false certificate emission, to attacks on certificate storage sites. This issue can be solved by replicating certificates, which introduces overhead for maintaining data consistence.

Referring to network churn, the Voting approach is quite resilient, since each peer is responsible for few information (only its own advices). Thus, even if a peer unexpectedly leaves the network the information loss is unimportant with respect to the whole collection of advices. Another aspect to consider is that stable malicious peers are more easy to detect than free riders. On the other side, honest providers are encouraged to stay connected even if they do not need resources. In the Transaction Certificates model, CAs are required to be stable nodes, too.

With respect to storage requirements, the Voting model is simple since each node is responsible for its own advices. On the other side, the Transaction Certificates model requires that a set of copies of each issued certificate is spread in the network, with all peers (or a subset) responsible for data maintenance. In the Voting scheme, storage requirements increase with the number of active providers involved in transactions, while in the other scheme its growth is proportional to the total number of performed transactions.

6. CURRENT & FUTURE DEVELOPMENTS

While P2P systems are deployed and used on a large scale, many research questions remain open. For example, current P2P systems form their overlay networks based on application layer information and ignore the structure and policies of the underlying network infrastructure. As a result, peers often connect to peers in a remote network even though an equivalent peer would be available in the local network. This can lead to the inefficient use of network resources and to conflicts between application- and network-level routing.

Moreover, next foreseen step is effectively using ontologies to describe not only shared resources, but also services to be composed and executed with the contribution of several peers. In a recent paper [100] we propose a structured peer-to-peer architectural model in which connectivity rules are interest-based, and service location information is placed in a distributed hash table (DHT), according to a deterministic strategy driven by semantic matching between service profiles and peer interests. The basic idea is to build a key space in which each ID provides complete information about the categories to which the resource (i.e. peer or service) is associated. In this context, the concept of interest corresponds to the categories to which a service belongs, or those for which the peer searches the network.

Finally, we cannot forget that a peer-to-peer system is a complex system, because it is composed of several interconnected parts (the peers) that as a whole exhibit one or more properties (i.e. behavior) which are not easily inferred from the properties of the individual parts. For example, in DUM-based architectures, the topology is a global property of the system that cannot be inferred from the configuration or from the history of each node. Another example is the time complexity, which depends on the number of nodes, on how they are connected and on how they behave (some of them can be fair in sharing their resources, while others can be free riders). The reaction of a peer to direct or indirect inputs from the environment is defined by its internal configuration, which can be either based on static rules shared by every peer (protocols), or on an adaptive plan. The latter determines successive modifications of the peer’s configuration, in response to the environment, and turns the P2P network in a complex adaptive system (CAS).

As we illustrated in section 2, many considerable peer-to-peer protocols have been recently proposed. They can be grouped in few architectural models, taking into account the placement of information about shared resources (hybrid or decentralized), and the logical organization (unstructured or structured). The behavior of a peer-to-peer system based on protocols follows a pre-established pattern.

On the other side, there is a lack of common understanding about adaptiveness. In our view, peers' internal structure may change in order to adapt to the environment. For example, consider a search algorithm whose parameters' values change over time in a different way for each peer depending on local performance indicators. The evolution of a structure can be phylogenetic, for which memoryless transformations are applied to the structure to modify it, or cultural or epigenetic, which assumes learning and knowledge transmission. In general, adaptive peer-to-peer networks emulate the ability of biological systems to cope with unforeseen scenarios, variations in the environment or presence of deviant peers.

In a recent paper [101], we proposed the Adaptive Evolutionary Framework (AEF) for peer-to-peer architectures. To show the potential of AEF, we used it to define a resource sharing scheme in which the evolutionary aspect is driven by a genetic algorithm. In our opinion, the research community should study other possible AEF implementations than genetic algorithms, considering also more complicated environments and applications, for which adaptiveness is not a plus but a fundamental requirement. This research topic could lead to novel technological solutions and, of course, to a number of interesting patents.

ACKNOWLEDGEMENT

This work has been partially supported by the European Commission in the frame of the Project METABO (7th Framework Programme, ICT Priority, Grant #FP7-ICT-2007-1- 216270).

CONFLICT OF INTEREST

The author declares no conflict of interest.

REFERENCES

[1] Barkai D. Peer-to-peer computing: Technologies for sharing and collaborating on the net. Intel Press 2002.

[2] Traversat B, Arora A, Abdelaziz M, *et al.* Project JXTA 2.0 super-peer virtual network, Technical Paper, Sun MicroSystems 2003.

[3] Traversat, B.A., Slaughter, G.L., Saulpaugh, A.M.M., Duigou, M.J., Pouyoul, E., Hugly, J.-C., Yeager, W.J., Pabla, K., Gong, L., Joy, W.N., Clary, M.J.: EP1229442 (**2007**).

[4] Peer-to-Peer Session Initiation Protocol (p2psip) working group. http://www.ietf.org/html.charters/p2psip-charter.html

[5] Jennings C, Lowekamp B, Rescorla E, Baset S, Schulzrinne H. Resource location and discovery (reload) base protocol. IETF Internet draft, December 2008.

[6] GauthierDickey C, Grothoff C. Bootstrapping of peer-to-peer networks. Proc Inter Workshop on Dependable and Sustainable Peer-to-Peer Systems, Turku, Finland, July 2008.

[7] Govoni D, Soto JC. *JXTA and security,* JXTA: Java P2P Programming 2002.

[8] BitTorrent official site. http://www.bittorrent.org

[9] Ipoque GmbH. Internet Study 2007.

[10] Slashdot. Comcast continues to block peer to peer traffic. 2007. http://yro.slashdot.org/article.pl?sid=07/12/01/0011253

[11] Antoniadis P, Le Grand B. Incentives for resource sharing in self-organized communities: From economics to social psychology. *Proc Inter Conf on Digital Information Management (ICDIM '07),* Lyon, France 2007.

[12] Li B, Yin H. Peer-to-peer live video streaming on the internet: Issues, existing approaches, and challenges. IEEE Com Mag 2007; 45(6).

[13] Netcraft. P2P Networks Hijacked for DDoS Attacks 2007.

[14] Liang J, Naoumov N, Ross K. The Index Poisoning Attack on P2P File-Sharing Systems. Proc IEEE INFOCOM, Barcelona, Spain, April 2006.

[15] Gambetta D. Can We Trust Trust? Trust: Making and breaking cooperative relations, D. Gambetta (ed.), Basil Blackwell, Oxford 1990.

[16] Abdul-Rahman A, Halles S. A distributed trust model. *Proc ACM Workshop New Security Paradigms (NSPW '97),* Langdale, Cumbria, U.K., September 1997.

[17] Amoretti M, Bisi M, Zanichelli F, Conte G. Introducing Secure Peergroups in SP2A, *Proc 2nd IEEE Inter Workshop on Hot Topics in Peer-to-Peer Systems,* co-located with Mobiquitous 2005, San Diego, California July 2005.

[18] Garces-Erice L, Biersack EM, Felber PA, Ross KW, Urvoy-Keller G. *Hierarchical Peer-to-Peer Systems, Proc Inter Conf on Parallel and Distributed Computing (Euro-Par 2003),* Klagenfurt, Austria, August 2003.

[19] Peng Z, Duan Z, Qi JJ, Cao Y, Lv E. HP2P: A Hybrid Hierarchical P2P Network. *Proc 1st Inter Conf on the Digital Society*, Gaudeloupe, Januarty 2007.

[20] Matsubara, D., Miki, K.: US20087467190 (**2008**).

[21] Manion, T.R., Dewey, J.L., Donner, R.D., Senkeresty, S.A., Gupta, R., Parks, U.W., Anderson, N.W.: US20087430747 (**2008**).

[22] Massoulie, L., Kermarrec, A., Ganesh, A.: US20087386606 (**2008**).

[23] Takeda, Y., Berkey, H., White, P.R., Vass, A.: US20087468952 (**2008**).

[24] Zhang, Z., Shi, S.: US20077313565 (**2007**).

[25] Free Haven site. http://freehaven.net/anonbib/date.html

[26] Clarke I, Sandberg O, Wiley B, Hong TW. Freenet: A distributed anonymous information storage and retrieval system. Lecture Notes in Computer Science 2001; 2209.

[27] Hariharan, R., Kannan, R.: US20077159108 (**2007**).

[28] Naoumov N, Ross K. Exploiting P2P systems for DDoS attacks. *Proc Inter Workshop on Peer-to-Peer Information Management (P2PIM),* Hong Kong, May 2006.

[29] Somin, G. M., Mowers, D., Gavrilescu, A.: US20077188254 (**2007**).

[30] Baset SA, Schulzrinne HG. An analysis of the skype peer-to-peer internet telephony protocol. *Proc 25th IEEE Inter Conf on Computer Communications (INFOCOM 2006),* Barcelona, Spain April 2006.

[31] Kleis M, Lua EK, Zhou X. Hierarchical peer-to-peer networks using lightweight superpeer topologies. *Proc 10th IEEE Symposium on Computers and Communication (ICSS05),* La Manga del Mar Menor, Cartagena, Spain, June 2005.

[32] Traversat, B., Gong, L., Yeager, W., Abdelaziz, M., Duigou, M. J., Poyoul, E., Hugly, J.-C., Joy, W. N., Clary, M. J.: US20077206841 (**2007**).

[33] Agosti M, Amoretti M, Zanichelli F, Conte G. P2PAM: A Framework for peer-to-peer architectural modeling based on peersim. *Proc of ICST/ACM SIMUTools 2008,* Marseille, France, March 2008.

[34] Yap, S.T.: US20087394817 (**2008**).

[35] Aboulhosn, A.L., Chen, R., Koo, D.M., Vineberg, D.J., Wald, J.F., Murphy, S.: US20056938042 (**2005**).

[36] Soulseek official site. http://www.slsknet.org

[37] Napster official site. http://free.napster.com

[38] OpenNap-NG official site. http://opennap-ng.sourceforge.net

[39] eMule official site. http://www.emule-project.net

[40] Gnutella RFC homepage. http://rfc-gnutella.sourceforge.net

[41] Limewire official site. http://www.limewire.com

[42] Kleinberg J. The small-world phenomenon: An algorithmic perspective. *Proc 32nd ACM Symposium on Theory of Computing (STOC '00),* Portland, USA, May 2000.

[43] Onana Alima L, El-Ansary S, Brand P, Haridi S. DKS(N,k,f): A family of low communication, scalable and fault-tolerant infrastructures for p2p communications. *Proc of the 3rd IEEE/ACM*

Inter Symposium on Cluster Computing and the Grid (CCGRID '03), Tokyo, Japan, 2003.

[44] Stoica I, Morris R, Liben-Nowell D, *et al.* Chord: A scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Trans on Networking 2003; 11(1).

[45] Rowstron A, Druschel P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. *Proc of IFIP/ACM Middleware, Heidelberg*, Germany, November 2001.

[46] Maymounkov P, Mazierès D. Kademlia: A Peer-to-peer information system based on the xor metric. *Proc of 1st Inter Workshop on Peer-to-Peer Systems (IPTPS '02),* MIT Faculty Club, Cambridge, MA, USA, March 2002.

[47] FastTrack documentation on WikiPedia. http://en.wikipedia.org/wiki/FastTrack

[48] KaZaA official site. http://www.kazaa.com

[49] iMesh official site. http://www.imesh.com

[50] Gnutella 2 developer wiki. http://g2.trillinux.org/index.php?title=Main_Page

[51] Tedeschi C, Desprez F, Caron E. Efficiency of tree-structured peer-to-peer service discovery systems. *Proc HotP2P '08*, Miami, FL, USA, April 2008.

[52] Wuala, the social online storage. http://www.wuala.com/en

[53] Lv Q, Cao P, Cohen E, Li K, Shenker S. Search and Replication in Unstructured Peer-to-Peer Networks. *Proc ACM Inter Conf on Supercomputing (ICS '02),* New York, USA, June 2002.

[54] Cohen E, Shenker S. Replication strategies in unstructured peer-to-peer networks. *Proc ACM SIGCOMM '02*, Pittsburgh, PA, USA, August 2002.

[55] Tewari S, Kleinrock L. Proportional replication in peer-to-peer networks. *Proc 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM),* Barcelona, Spain, April 2006.

[56] Kangasharju J, Ross K, Turner D. Adaptive content management in structured P2P communities. *Proc Inter ICST Conf on Scalable Information Systems (INFOSCALE '06),* Hong Kong, China, June 2006.

[57] Tang X, Xu J, Lee WC. Analysis of TTL-based consistency in unstructured peer-to-peer networks. IEEE Trans Parallel Distributed Sys 2008; 19(12).

[58] Richardson, R.L., Ozzie, R.E., Ozzie, J.E.: US20087340502 (**2008**).

[59] Ranjan R, Chan L, Harwood A, Karunasekera S, Buyya R. Decentralised resource discovery service for large scale federated grids. *Proc 3rd IEEE Inter Conf on e-Science and Grid Computing (e-Science 2007),* Bangalore, India, December 2007.

[60] Trunfio P, Talia D, Papadakis H, *et al.* Peer-to-peer resource discovery in grids: Models and systems. Future Generation Computer Systems 2007; 23(7).

[61] Gerloff D, Quan X, Walton C, *et al.* Bioinformatics scenarios". Deliverable D6.1, OpenKnowledge 2006.

[62] Dewey, D.W.: US20077209932 (**2007**).

[63] Microsoft Office Groove homepage. http://office.microsoft.com/en-us/groove/default.aspx

[64] Chen, Q., Hsu, M.: US20077236939 (**2007**).

[65] TVU networks official site. http://pages.tvunetworks.com

[66] Joost official site. http://www.joost.com

[67] TVants official site. http://www.tvants.com

[68] LiveStation official site. http://www.livestation.com

[69] Bawa M, Deshpande H, Garcia-Molina H. Transience of peers & streaming media. SIGCOMM Comput Commun Rev 2003; 33(1).

[70] Tran D, Hua K, Do T. Zigzag: An efficient peer-to-peer scheme for media streaming. Proc. *22nd Annual Joint Conf of the IEEE Computer and Communications Societies (INFOCOM*), San Francisco, USA March 2003.

[71] Li J. Peerstreaming: A practical receiver-driven peer-to-peer media streaming system. *Proc 7th IEEE Workshop on Multimedia Signal Processing,* Shanghai, China October 2005.

[72] Li, J.: US20077174385, (**2007**).

[73] Eugster PT, Guerraoui R, Kermarrec AM, Massoulie L. From epidemics to distributed computing. IEEE Comput 2004; 37(5).

[74] Zhang X, Liu J, Li B, Yum TP. CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming. *Proc 24th Annual Joint Conf of the IEEE Computer and Communications Societies (INFOCOM),* Miami, USA, March 2005.

[75] Pianese F, Perino D, Keller J, Biersack EW. PULSE : An Adaptive, incentive-based, unstructured P2P live streaming system. IEEE Trans on multimedia 2007; 9(8).

[76] Magharei N, Rasti AH. Prime: Peer-to-peer receiver-driven mesh-based streaming. *Proc 26th Annual Joint Conf of the IEEE Computer and Communications Societies (INFOCOM),* Anchorage , Alaska , USA May 2007.

[77] Buyukkaya E, Abdallah M. Data management in voronoi-based P2P gaming. *Proc 5th IEEE Consumer Communications and Networking Conf (CCNC 2008),* 2008.

[78] Bharambe A, Douceur JR, Lorch JR, *et al.* Donnybrook: Enabling large-scale, high-speed, peer-to-peer games. *Proc ACM SIGCOMM*, Seattle, WA, USA August 2008.

[79] Russell, G. K., Shelby, M. B., Jordan, J. R., Norton, J. R.: US20077270605 (**2007**).

[80] Rakion official site. http://rakion.softnyx.net

[81] Gunbound official site. http://gunbound.softnyx.net

[82] Rumble Fighter official site. http://rf.ogplanet.com

[83] Soldier Front official site. http://sfront.ijji.com

[84] LionShare official site. http://lionshare.its.psu.edu

[85] Nejdl W, Wolf B, Qu C. "EDUTELLA: A P2P networking infrastructure based on RDF". *Proc ACM WWW2002*, Honolulu, Hawaii, USA May 2002.

[86] IST Advisory Group. Scenarios for ambient intelligence in 2010. European Commission, 2001.

[87] Ramos C, Augusto JC, Shapiro D, Ambient Intelligence - the next step for artificial intelligence. IEEE Intell Syst 2008; 23(2).

[88] Avatangelou E, Dommarco RF, Klein M, *et al.* Conjoint PERSONA-SOPRANO Workshop. *Proc 1st European Conference on Ambient Intelligence (AmI-07),* Darmstadt, Germany November 2007.

[89] Rasmusson L, Jansson S. Simulated social control for secure internet commerce. Proc ACM Workshop New Security Paradigms (NSPW '96), Lake Arrowhead, USA September 1996.

[90] Gupta, R., Gavrilescu, A., Miller, J. L., Wheeler, G. A.: US20087444372 (**2008**).

[91] Aringhieri R, Damiani E, De Capitani di Vimercati S, Samarati P. Assessing efficiency of trust management in peer-to-peer systems. *Proc 14th IEEE Int'l Workshops Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE '05),* Linkoping, Sweden June 2005.

[92] Kamvar SD, Schlosser MT, Garcia-Molina H. The eigentrust algorithm for reputation management in p2p networks. *Proc of the 12th Inter Conf on World Wide Web*, Budapest, Hungary May 2003.

[93] Song S, Hwang K, Zhou R, Kwok YK. Trusted P2P transactions with fuzzy reputation aggregation, IEEE Internet Computing 2005; 9(6).

[94] Aringhiri R, Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P. Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems, J Am Soc Infor Sci Tech 2006; 57(4).

[95] Traversat, B., Gong, L., Yeager, W., Abdelaziz, M., Duigou, M. J., Poyoul, E., Hugly, J.-C., Joy, W. N., Clary, M. J.: US20087437440 (**2008**).

[96] Yeager, W., Chen, R. Y.: US20077213047 (**2007**).

[97] Gupta R, Somani AK. Reputation management framework and its use as currency in large-scale peer-to-peer networks. *Proc of the 4th IEEE Inter Conf on Peer-to-Peer Computing,* Zurich, Switzerland, August 2004.

[98] Yeager, W., Chen, R. Y.: US20087383433 (**2008**).

[99] Yeager, W., Chen, R. Y.: US20077308496 (**2007**).

[100] Amoretti M, Agosti M, Zanichelli F. Interest-based overlay construction and message routing in service-oriented peer-to-peer networks. *Proc IASTED Parallel and Distributed Computing and Networks (PDCN 2008*), Innsbruck, Austria, February 2008.
http://www.ipoque.com/resources/internet-studies/internet-study-2007

[101] Amoretti M. A Framework for Evolutionary Peer-to-Peer Overlay Schemes. Proc. *European Workshops on the Applications of Evolutionary Computation (EvoWorkshops 2009),* Tubingen, Germany, April 2009.