Mobile Peer-to-peer systems: Overview, issues and potential usages

Introduction

Mobile devices are now everywhere. These personal devices are used for interpersonal communications (phone call, video call, SMS) but they are also increasingly used to access the internet or to retrieve information from connected applications while on the go. In several poor countries, mobile Internet access even surpasses fixed internet access. Meanwhile, the mobile internet world is a much more controlled and constrained environment. Limitations come from the capacity of the terminal, from the connectivity technologies available on the device or from restriction policies applied by mobile network operators. Nevertheless, efforts are made to use mobile IP network connectivity at its full capabilities, and to allow mobile terminals to share contents they are now able to produce: contextual information, multimedia documents… In order to circumvent the constraints imposed by mobile network operators, one can imagine developing an alternative to centralized network models in order to give the user the possibility to access a new range of services. The best way to do it is to set up a peer-to-peer platform.

In this article, we will try to figure out how the principles of peer-to-peer networks apply to mobile networks, and how they can be used to serve mobile communication or contextual applications. First, we will present peer-to-peer networks from a general point of view. Then, we will see how the concepts of peer-to-peer networks can be used in two kinds of mobile networks: mobile ad-hoc networks and mobile operated cellular networks. After that, we will focus on some of the problematic related to the use of peer-to-peer network technologies in mobile cellular networks. Finally, we will give some examples showing how peer-to-peer technologies can be used in mobile applications.

General presentation of peer to peer networks

For many people, peer-to-peer is a synonym for illegal file sharing. In the context of the workshop, peer-to-peer can be more accurately described as a collaborative form of content production, be it an article on the wiki, a set of open source specifications for electronic components or a multimedia content. But peer-to-peer can be given a rather technological definition.

From a computer science perspective, as stated in Wikipedia, “A peer-to-peer computer network is a network that relies on computing power at the edges (ends) of a connection rather than in the network itself. Peer-to-peer networks are used for sharing content like audio, video, data or anything in digital format. Peer-to-peer network can also mean grid computing.”

According to this definition, all the nodes are equal in a peer-to-peer network, in opposition to the client-server model. All the resources are spread in the network and shared among the nodes participating in the network: computing power, bandwidth or storage capacity. This provides robustness to the system because the failure of one node in the network does not harm the collaboration of the others, whereas a server crash is dramatic in a client-server model.

Moreover, providing a service in a peer-to-peer network is less costly than in a client server model. Indeed, when you want to propose a service to a broad number of users in a client-server fashion, you have to design your server in order to handle the requests for service from your targeted audience. You may require a high bandwidth, a lot of processing power and a significant storage capacity to allow every possible user to access it, while in a peer-to-peer network the cost is shared by all the nodes in the network.

Nevertheless peer-to-peer networks also have some drawbacks. It can be tedious to locate resources in a peer-to-peer network, whereas the resource locating mechanism is fairly simple in a client-server model as all the information is centralized. This is the reason why some peer-to-peer applications use servers to discover resources in the network (typically, Napster (1)). This is one of the characteristics that can be used to categorize peer-to-peer networks.

Categorization of peer-to-peer networks

Peer-to-peer networks have been built using different kinds of architectures and internal logics. Those systems were designed in order to adapt to the specificities of the networks on top of which they operate and to the characteristics of the applications using them. For instance, some applications like real time communications require being able to identify a single node quickly in a large network, while others such as file sharing focus on locating the same resource in different end nodes in order to retrieve the resource more reliably. Thus peer-to-peer applications and overlays adopted a wide range of approaches to solve the specific issues they want to tackle, and they can be categorized according to these approaches.

Several criteria can be used to characterize peer-to-peer networks. First, we can discriminate peer-to-peer systems according to the organization of the nodes. Sure, every node should be able to connect to any other node in a pure, peer-to-peer approach. Meanwhile, in order to allow the operation of the network when it is quite large, several networks adopt organization principles. In those networks, nodes are organized together in order to limit the number of connections they have to maintain and to keep message routing feasible in the network. These organization principles often aim at optimizing the network for application-specific purposes.

Besides, network can be discriminated according to the role nodes can play in the network. Ideally, every node is strictly equivalent to the others, but in some networks, disparities appear because some nodes have more resources available, or on the contrary are not capable enough to operate on the network. In order to address this kind of issues, some peer-to-peer networks such as JXTA (2) or Skype(3) have introduced the concept of supernodes. These nodes are specific nodes that hold more resources than most of the other peers. Least capable nodes can delegate some specific operations to these peers. They can also play a more important role in routing operations as they are often more stable in the peer-to-peer network, i.e. present for a longer period of time.

At last, the resource discovery mechanism adopted in the peer-to-peer network is an important criterion, because the way nodes locate and access resources shared within the network is what ties these networks together. This is the main problem to address when moving an application from a client-server model to a peer-to-peer model, and efficient delivery methods are a key to the operation of these decentralized systems.

These three criteria help defining five classes of peer-to-peer networks (4): Directory networks, Flooding networks, Distributed Hash Table networks, Epidemic networks and Social networks.

Directory networks (Picture: Directory.jpg) are peer-to-peer networks where connected peers register the resources they share on a directory. This directory can be centralized, or distributed among a set of directory servers or supernodes. In those network, resource discovery is performed by interrogating the directory, which can also be used to perform bootstrapping operations for peers willing to connect to the network. This kind of peer-to-peer network is the closest from the classical client-server model, and was mainly used in Napster, Emule (5) or Edonkey (6), which are quite old peer-to-peer applications. The centralization of the directory is a weakness in the system, which armed the operation of Napster when it was asked to stop its operation. Meanwhile, these systems allowed peer-to-peer networks to gain popularity and soon their weaknesses were addressed by flooding networks.

Flooding networks (Picture: Flooding.jpg) appeared to bypass the relative centralization of directory networks. In these networks, resource discovery is performed by sending messages to every neighboring peer. When a node receives such a discovery message, it looks up in its own resources if it has a match, and if not, it relays this message to its own neighbors. When a node has a resource matching the request, it sends back an answer to the peer that relayed the message, and the answer makes its way to the originator of the request. From this description, it can be intuitively understood that this process generates a lot of messages on the network. Besides, a given node can receive the same request from multiple neighbors. This drawback is the main issue of flooding networks. Indeed, these networks are not efficient because there is a big signaling overhead, i.e. the number of messages sent on the network for discovery and network management is too high compared to the useful traffic. Despite this drawback, flooding networks have been used in the first versions of the Gnutella (7) network, and proved being successful in the absence of more efficient alternatives.

Distributed hash table networks are now among the most efficient peer-to-peer networks. In these networks, peers are logically organized. This organization follows a resource distribution algorithm based on global resource identifiers. These identifiers are computed using hashing functions. The nodes in the network organize themselves on a logical ring, where their location is given by their hash identifier. When a resource is shared on the network, its localization is under the responsibility of the node whose hash is the closest to the resource's hash identifier. When a node wants to locate this resource, it sends a message on the network to the responsible node using a logical routing algorithm. The algorithms behind these networks are less simple than in the two other kinds of network, but the use of a logical organization in the network makes discovery operations faster, and the network can operate in a much more efficient way while the nodes don't have to maintain as many connections to neighboring nodes as in flooding systems. Besides, these network can use the concept of supernodes by setting up parallel logical networks only gathering those more powerful nodes to take in charge specific operations, thus using the most interesting concept of directory networks. Most of the research work done in the peer-to-peer area is done on these networks, and they are adopted in research projects such as Chord (8) or Pastry (9), standardized peer-to-peer communication systems or new applications such as Azureus (10).

Epidemic networks have appeared recently to propose another approach to the problems solved by distributed hash table networks. Indeed, epidemic networks tend to adopt a random approach to build a peer-to-peer network overlay. In these networks, peers tend to organize themselves in order to maximize the use of a given resource (Bandwidth, computing power…). Peers in the network exchange information about this resource with their neighbors of course, but also with other nodes participating in the network selected randomly. This random selection aims at avoiding deadlocks or performance degradation due to rapid, dynamic changes in the network. These networks are not implemented in many systems, but promising researches are made within the framework of the Tribbler (11) application or of the BuddyCast (12) project.

Social network are peer-to-peer networks where peers connect according to a previous knowledge of the other peers. Indeed, those peers connect because they know each other, and want to share resources among their community. Those networks may become more and more popular in the future because of the level of trust between the peers. This trust is a good way to tackle peer-to-peer network monitoring from copyright owners or people who want to reduce the sharing of illegal downloads on peer-to-peer systems. Nevertheless, these networks may result in a rather inefficient sharing of resources because of the size of the communities where resources are shared.

Applying peer-to-peer concepts in mobile networks.

The huge improvements of mobile networks in term of bandwidth and the new capabilities of mobile devices such as personal digital assistants or mobile phones raised some interests in order to develop peer-to-peer applications for mobile devices. In our view, mobile peer-to-peer systems are systems where mobile devices can collaborate together and with fixed devices without the intervention of a central server. These systems can either connect together spontaneously in an ad-hoc fashion, or use telecom operator's mobile networks to connect to peer-to-peer systems on which they may collaborate with fixed peers.

Mobile Ad-Hoc Networks or MANETs are spontaneous networks established between wireless devices without any router intervention. These networks do not organize according to a predefined topology, and they may be set up for a quite short period of time. These networks may be standalone, or connected to a larger network if one device can share its connection to such a network. The spontaneous networks established between Bluetooth devices are perfect examples of mobile ad-hoc networks. In these networks, the routing of packets must be done by the end node himself. This raises a problem if two nodes must communicate indirectly through multiple hosts. A large number of researches have been made to address this problem, and two kinds of protocols have been designed for mobile ad-hoc networks. In networks using proactive routing protocols, every host in the network keeps track of the routes to reach every other host in the network. As one may expect, proactive routing is not scalable in networks that are by definition unstable. Using distance vector routing would lead to slow convergence of the routing algorithm after a network change, and link state routing would require a lot of calculations in the hosts every time a change in the network is detected i.e. often according to the nature of ad-hoc networks. The reactive routing protocols are much more scalable. In this approach, a route is established only when needed. This may slow down the establishment of a connection but hosts don't need to handle the management of any routing table.

Those networks are interesting in many regards, but the scope of these networks is quite narrow. Indeed, MANETs are at most citywide networks, while one may want to cooperate with peers who are farther than that. Mobile collaboration to peer-to-peer networks using telecom operator's 2G or 3G data networks offers the possibility to collaborate in larger networks. This kind of application using cellular data networks has benefited from the development of UMTS or EDGE networks which now cover a larger area in most countries where those networks are operated, and from the evolution of the pricing model for these connections, which are now affordable. Flat rate offers even reach the market in countries where the competition between mobile operators is strong. Besides, mobile devices now have enough storage capacity and computing power to participate in peer-to-peer networks, and they even have cameras and microphone that allow them to produce and edit multimedia content that can be shared on those networks. But there are some specific issues to tackle in order to collaborate to peer-to-peer networks using GPRS or UMTS IP connectivity.

Most of these issues are related to the structure of mobile cellular networks and to network management policies adopted by most telecom operators. First of all, when we consider the architecture of mobile cellular networks, UMTS, HSDPA or GPRS, we can observe that those networks have a pyramidal structure. On the schema (Picture: GPRS Architecture.jpg) you can see below, mobile devices are connected to either a node-B or a base station (BTS), but the data takes the form of a regular IP datagram on the interface between the SGSN and the GGSN. This has an influence because the traffic and signaling can't take shortcuts when a mobile peer tries to connect to a peer that is under the responsibility of the same node-B or BTS. Besides, in such networks, mobile peers don't have the possibility to broadcast messages on the network, so this kind of messages can't be used for bootstrapping or discovery operations. In mobile networks, the possibility for a given mobile to establish a data connection to an IP network is a costly resource for mobile network operators, which restrict the use of some protocols or transport methods on their networks. To tackle these limitations, first the peer-to-peer systems have to be efficient and to avoid sending too many discovery or network maintenance messages on the network in order to reduce its footprint. Thus, the discovery mechanism and the architecture of the peer-to-peer network have to be adapted to this constraint. Besides, in order to maintain its reachability in networks where network address translation is often done in a very ephemeral way, mobile peers have to refresh regularly their connection to their neighbors in order that they know how to reach them. At last, even if mobile devices are more capable than a few years ago, they are still less powerful than fixed workstations who may participate in the same peer-to-peer systems. For instance, the operation of these mobile peers on the peer-to-peer network has a power cost, and reduce the battery life of these devices through the intensive use of data connections.

The use of the concept of supernode is a good way to tackle these different issues. Indeed, in fixed-mobile peer-to-peer networks, mobile peers could collaborate with a fixed peer for tedious operations such as the discovery of resources within the network or maintaining the connectivity of the mobile peer through the operator's network address translation system. In case the mobile operator restricts the use of some protocols, the fixed peer can also take in charge the network protocol translation operations. This way, all the operations that are made difficult by the structure of the mobile data network or by management policies are done by the fixed peers. Those peers then provide the mobile peer with the capability to fully interact with the other peers in the network. They should be selected according to an automatic mechanism, based on their stability in the network or on the amount of available resources they share. As we have seen before, this supernode concept is available on both centralized and distributed hash table networks. Given the less centralized nature of distributed hash table networks, this kind of system is particularly adapted to the implementation of peer-to-peer systems in a mobile cellular data network environment.

Among the different experimentations related to mobile peer-to-peer systems, JXME is particularly interesting. JXME (13) stands for JXTA Mobile Edition. It is a side project of JXTA, an open source peer-to-peer framework sponsored by Sun which implements a broad set of discovery and network management mechanisms to set up large peer-to-peer systems. JXME takes advantage of the protocols designed by the JXTA community to set up a virtual network layer on top of GPRS or UMTS network, using the ability of mobile devices to use the TCP/IP protocol stack with small modifications. Two versions of JXME have been developed. The first version is known as the proxied version because mobile JXME peers had to use a fixed relay peer to access the services of the JXTA network. The relay, which had to be configured manually, acted on behalf of the mobile peer to forward queries on the network and to trim received advertisements. The development of the first version of JXME has been stopped when the definition of JXTA 2.0 protocol stack has been published. Now, the JXME community tries to develop a proxyless version of the JXME platform. This platform will be compatible with fixed hosts running JXTA 2.0. Mobile hosts running the second version of JXME are expected to support the reception of messages in binary format as well as in XML format, as JXME will incorporate an XML parser. Besides, they should not use any statically-configured relay to perform operations on the JXTA network, and thus they should be able to propagate their advertisements alone. This project is very promising but lacks being used in a real world application. Nevertheless its use of the supernode concept and its adoption of java as a programming language make it a good project to experiment mobile and hybrid peer-to-peer systems.

Applications of mobile peer-to-peer systems

In the last few years, several developers have implemented mobile clients to popular peer-to-peer file exchange systems such as Gnutella or Bittorrent (14). These clients have not been really successful because they don't make an efficient use of the network connectivity, resulting in high network consumption. But peer-to-peer technologies can be used in a much more clever way to serve typically mobile usages.

First, peer-to-peer systems can be used to set up real-time collaborative applications. Communication can either be the main purpose of these applications, like in Skype, or an enabler serving another purpose. For instance, mobile peer-to-peer systems may be used for multiplayer gaming in order to allow players involved in the same game to exchange information together on their positioning and actions. In those applications, the use of peer-to-peer systems compared to a classical client-server approach significantly reduces the load on the server, and also helps reducing the message transmission delay, which is critical in real-time collaboration applications. For those applications, peer-to-peer systems prove being more scalable as they adapt to the number of connected peers by design.

Besides, peer-to-peer systems can also be used by mobile devices to share the content they are able to produce with the others. Indeed, mobiles can now be considered as multimedia content production endpoints because they have a camera and microphone. Besides, those devices contain a lot of personal multimedia content (personal pictures, music…) that a user may want to share with its community. At last, these devices can also broadcast live audio and video contents. On the internet, a range of services such as Qik (15) provide users with the ability to stream media to their community, but these services are centralized. Those services could be proposed in a peer-to-peer fashion, which would make it a pure software service, while centralized content sharing platforms require maintaining an infrastructure to store the content or to give access to the mobile broadcasting the content. Such peer-to-peer systems are called application layer multicast services because they mock up the functioning of multicast networks to spread content among peers.

Mobiles can also be used as contextual information sources. This information comprises location of course, but also local usage of the device (on a phone call, on mute…), agenda or list of contacts in the phone book. This information may be used in a broad range of contextual services, form social platform to community local information platforms. The use of this information raises privacy and security concerns, but peer-to-peer solves most of the issues raised by the upload of this information on a centralized platform. Indeed, if the information is not shared using a centralized platform, then nobody is able to gather global information by monitoring the service on a central point, and user keep control on their contextual information.

Such principles also apply to more static personal information. As most users have a special relationship to their device, and don’t share it with other people contrarily to a fixed computer, we can see these devices as personal markers, holding information about their user. Thus, mobile devices can be seen as personal data repositories on which user have a full control on what they want to share with others. Applications allowing sharing such information already exist in both the fixed and mobile world, with projects like Opera Unite (16) or Nokia’s Mobile web server (17), but these initiatives rely on services provided by either Nokia or Opera in order to locate the user. Those services could benefit from peer-to-peer systems as this would decentralize the discovery service. This way, those services would not rely on a small set of central nodes to operate correctly. Besides, those services could be used as data backend for social network services. Thus, all the data shared on these services would be located on a user premise, and thus it would be under full control of the user. This approach of peer-to-peer systems goes against the global cloud computing trend, where all the data is moved to servers in the network, and decentralized systems tend to give an original answer to most concerns related to data portability and user privacy on those cloud services.

Conclusion

In this article, we have presented peer-to-peer systems from a technical point of view, and we explained how they could apply to mobile networks. Although only few mobile applications use these concepts, the potential of mobile peer-to-peer allows us to envision a broad range of applications taking advantage of the decentralized nature of these networks to propose social services. Besides, peer-to-peer systems could give a technically interesting alternative to build social network services on which data are kept under the full control and responsibility of the user.

References

(1) Napster: http://www.napster.com/

(2) JXTA: https://jxta.dev.java.net/

(3) Skype: http://www.skype.com/

(4) *Pair-à-Pair: Architectures et Services*, Fabrice le Fessant (in french) http://www.forumatena.org/presentations/FabriceLeFessant.pdf

(5) Emule: http://www.emule-project.net/

(6) Edonkey: http://en.wikipedia.org/wiki/EDonkey_network

(7) Gnutella: http://www.the-gdf.org/

(8) Chord: http://pdos.csail.mit.edu/chord/

(9) Pastry: http://research.microsoft.com/en-us/um/people/antr/Pastry/

(10) Azureus: http://azureus.sourceforge.net/

(11) Tribbler: http://www.tribler.org/

(12) Buddycast: http://www.jet.net/buddycast/

(13) JXME (JXTA for J2ME): https://jxta-jxme.dev.java.net/

(14) Bittorrent: http://www.bittorrent.com/

(15) Qik: http://www.qik.com/

(16) Opera Unite: http://unite.opera.com/

(17) Nokia Mobile web server: http://opensource.nokia.com/projects/mobile-web-server/