In the Beginning was the Peer-to-Peer…

Ekaterina Chtcherbina, Thomas Wieland

Siemens AG, Corporate Technology
Munich, Germany
E-Mail: ekaterina.chtcherbina@siemens.com, thomas.wieland@siemens.com

Abstract

Peer-to-Peer systems have become extremely popular about one, two years ago when researchers all over the world started to talk about the "potential" of such kind of systems. It is especially interesting taking in consideration the fact that centralized systems appeared later than pure peer-to-peer systems did, but very soon pushed the peer-to-peer completely out of the scene. The "killer" application that would prove the 100% necessity of fully decentralized systems or their 100% benefit in comparison to the centralized systems is still not found as it was not found 30 years ago when the first peer-to-peer systems appeared. Nevertheless, we do talk about peer-to-peer today as about technology of high importance. We do not want to dedicate this paper to the search for a killer application for peer-to-peer technology or to attempts to find the ways to replace the centralized systems. Instead, we will talk about other class of systems – "ad-hoc" – and about peer-to-peer as enabling technology for ad-hoc systems. We would like to analyze why do we need ad-hoc systems, discuss their features and requirements, and based on this analysis discover the opportunities of peer-to-peer networking. We also provide a list of requirements to the peer-to-peer systems as well as to the existing network technologies that need to be fulfilled in order to fully enable ad-hoc networking in the future.

Ad-Hoc Systems: Five Reasons to Exist

The term "ad-hoc" is often used to refer to any kind of dynamic environment, where network is fashioned from "whatever is spontaneous available". An ad-hoc network is the collection of mobile (not necessarily wireless) nodes without the required intervention of a centralized access point or existing infrastructure. The connectivity between the nodes is dynamic and can be broken as the nodes move about the network.

Very often discussions of benefits of ad-hoc systems in comparison to standard Internet technologies can be found. Among those benefits are cost saving, direct access, native way of communication, and solution for the problem of bottlenecks in the network. At the same time there is a whole set of scenarios and situations where "ad-hoc" features of the system are a must. Let us look at five reasons for ad-hoc systems to exist.

The first reason is "*No infrastructure*". Here we are talking about situations when infrastructure does not exist yet or the connectivity to the infrastructure is temporarily broken. Places like construction sites, warehouses, streets, harbors, or pure natural sites might be not equipped with IT networking infrastructure. People, on one hand, still desire means of communication even if the infrastructure is not available. Somebody in the mountains, for instance, might require help and want to make an emergency call. Today, if there is no infrastructure, people cannot do that. On the other hand, if there were a possibility just to find other people who are somewhere near the person who needs help, it could help a lot in various situations. Also, when the connectivity to the infrastructure is broken for any reasons but interactions are required, the need of finding other nodes in the network without accessing the infrastructure can arise. A centralized approach cannot help in these situations because it is not possible to access the centralized services without an infrastructure.

The second reason is "*Changing environments*". This is not the same as mobility but refers to the ability of integrating in a new situation without changing a location. Referring to the changing environment we can talk, e.g., about a system that should work in the home environment as well as in the office environment providing a relevant behavior according to the environment's self-description. The system in the changing environment should be able to adapt itself to a new environment and use the network resources provided in this environment without requiring a manual change, i.e. a user interaction. A system should be smart enough to obtain the information for the situational behavior from the environment itself.

The third reason is "*Mobility*". Mobility means that participants of the network are changing their location during the usage of the system. When talking about mobility, often scenarios of cars rushing through the streets of a city or many people walking around in public places like shopping centers or stadiums are presented. But mobility can also mean that a device stands on one place for several hours and is then moved to another place to reside there for another couple of hours. Each time a network node moves (or is moved), the entire topology of network changes. Two nodes that were just close together and could easily interact with each other, might be far apart in the next moment requiring some dedicated "routers" in between in order to establish any kind of

communication. Or it might happen that a node moves out of the scope of the network and loses the connection completely, while other nodes come into the reach and want to participate. Ad-hoc systems are able to work under these circumstances and even can handle the additional constraints implied by mobile nodes, such as limited bandwidth and power.

The fourth reason is "*No planning*". This is when the behavior or an event in the future cannot be planned or forecasted. To this extent, an ad-hoc system can be regarded as indeterministic. What looks like the edge to chaos at the first glance, turns out to be in fact a particular strength. A system that is able to react to circumstances that have not been anticipated when it was designed is much more flexible and adaptable. An example is finding a service by given parameters that might reside anywhere in the network. As the availability and accessibility of this service can change continuously, the system cannot plan the time and place when this service will be discovered. We should also keep in mind that even centralized systems are not as deterministic as they seem to be. Especially the response times and sequences in concurrency situations depend on so many factors that they cannot be forecasted reliably.

And the fifth reason is "*Instability*". This does not mean instability in the sense of frequent crashes due to errors in the application or operating system software. We want to refer here to system immanent instability by design, usually caused by external forces out of the scope of the system. Examples are rapidly changing conditions for wireless connectivity (e.g. by interferences or shieldings) or fault tolerance for massive hardware failures or disasters. If the system is highly unstable, it is worth being ad-hoc in order to be able switch to another working environment if the problem occurs. This might happen also in industrial scenarios where everything must be highly stable. Anyway, instability happens and systems must be able to react to that.

Ad-hoc systems become more and more important because of the growing mobility of the society and as a result of the growing need of "easy-to-use", "always-in-the-pocket" software systems. If we talk today about the ideal software systems in the area of personal communications, business solutions, or home automation, they need to be "simple", "easy to configure", "always available", "stable when necessary" and "secure". This is what ad-hoc networking is designed for.

Personal communication is a technology area where everybody has his or her own experiences and preferences because it is more or less part of every-day life. In the discussion about peer-to-peer networking these experiences tend to push aside an area that is certainly even more important: the intelligence in devices. More and more formerly "dumb" electrical devices carry sophisticated software systems with them that enable them to get in contact with other systems, receive and process orders. To fully exploit the benefits of this intelligence and the value of the network, it is necessary that the nodes of the system communicate with each other providing also their intelligence to the network. The natural use case is plugging them together and let information exchange start. This is what ad-hoc networking means.

Features and Challenges of Ad-Hoc Systems

Having discussed the reasons for ad-hoc systems to exist, we can now formulate the features that should be available in these systems.

Self-Organization

One of the most important features is self-organizing. The self-organization means the ability of the system to change its structure as a function of participants of the system and its environment taken together. It is important to emphasize that participants only cannot create a self-organizing system because the system does not organize itself independent of the environment. In order to form a self-organizing system all participants have to discover and get the information about the surrounding environment. There are two possible mechanisms of implementation of this idea: one is discovery messaging (pull mechanism) and another is an advertisement messaging (push mechanism). Discovery mechanism is used to get the information about the system environment on demand whenever the participant needs this information. Advertisement is used to publish information that reflects changes of the environment without a demand coming from the system. In fact, discovery or advertisement are rarely used alone, but commonly a combination of them is employed in order to optimize the performance of the system. In that case the system provides a caching mechanism for storing advertisements providing the stored information about the environment on demand whenever requested.

The challenge in a search mechanism is to find a right balance between discovery and advertisement messages as well as the appropriate frequency of those messages. In order to optimize the search mechanism in the wide area network, a look-up service is required. The challenge for setting up such a look-up service is to integrate a right number of providers of the look-up service within the network in order to optimize again finding of information, stability and storage of information. This is an optimization function with three degrees of freedom.

Flexibility

Flexibility is another feature of the ad-hoc systems. Flexibility means an ability of the system to adapt its behavior to a new unknown unpredictable situation. In other words, the system has to be able to provide a situational behavior, e.g. change the behavior based on the changing circumstances. Ad-hoc system should be able

1. to support dynamic service composition
2. to change the network if the connectivity is broken
3. to react on new service supplies and much more.

The main challenge by flexibility is unstable connectivity and the session hand-over in case of change of the network topology or change of the central access point.

Dynamic Reactivity

Taking in consideration the "mobility" and "no planning" aspects of the ad-hoc system, they need to be dynamic. That means they should be able to provide a continuously productive activity or change. If the environment requires change of activity or presentation of the activity in another form, the system must be able to react on this requirement in a productive and efficient way. This dynamic adaptation capability is required for example for service and device adaptation based on context awareness and local information.

Challenges are multiple device support as well as obtaining and actively reacting on the up-to-date environment information.

Openess

Ad-hoc systems need to be open. This is a very simple requirement, which is very difficult to implement. "Open" means accessible from all or nearly all sides. Of course, talking about software systems, we cannot require the system to be completely open for other systems (also taking in consideration technical and security issues), but instead we say that "open" means based on open standards. Open standards is a key that allows more devices and applications to participate in the process of self-organization of the network and later on in interactions, this way increasing the value of the network.

In order to become open, the system must be not only interoperable with other systems on the protocol level but also to be able to act as a player in the network. That means providing also the basic cross-platform functionality that will enable networking on the application level.

Decentralization

Ad-hoc systems are often fully decentralized like in the example without infrastructure. They might be "often" decentralized but not necessarily always. Ad-hoc means fashioned from whatever is spontaneous available. If a server is available, it can also participate in the ad-hoc networking. That means, sometimes they can be controlled by a central point (user management, total error handling etc) but they might be able to adapt to a situation where no centralized approach is applicable and each node of the network should be a server and a client simultaneously (a so-called servent). In order to provide this capability ad-hoc systems need fully decentralized search and communication mechanisms.

One of the biggest challenges is a multi-hop routing in the network consisting of mobile devices with the power limitation as well as wireless connectivity with the limited bandwidth. Looking at the business side of the problem, there are no billing and payment mechanisms that would allow the ad-hoc network to use the available resources of separate individuals. This is, at least, a major problem for the practical acceptance in many realizations of ad-hoc systems: How can a user be persuaded to let others make use of his resources, i.e. (mis-) use him as a router? Another challenge in the decentralized environment is decentralized security trust model that would allow the participants of the network to initialize trust concept in a distributed manner [Sch01].

Mobility

Taking again the mobility aspect of the ad-hoc networking into consideration, the systems need to be often mobile. That means being physically mobile as well as on the logical level being capable to be productive in the changing environment. On the application level there is a need to support changing network technologies (LAN, WLAN, Bluetooth, GPRS) and addressing in mobile environment. An intelligent support for wireless

technologies is also required such as choosing an optimal multi-hop routing or choosing a TTL for a request based on the network topology and available bandwidth.

The mobility aspect requires a "smart" support for wireless technologies. That means analyzing the problems on the transport level and providing support for them on the application level. One of the problems in the wireless standard IEEE802.11 is that the TCP performance goes down due to the limits of the bandwidth. Ad-hoc mode of the same standard has several problems with the multicasting. In order for the system to work efficiently with those wireless technologies, the system must be aware of the problems and react properly. That also can affect frequency of requests sent to the network to get the desired number of successful deliveries.

Architecture of the Ad-hoc System

One of the examples of the system that should be self-organizing, flexible, dynamic, open, often decentralized and often mobile, is a system that is targeting mobile community support [GSG01]. It is worth taking this example in order to analyze unsolved issues in the ad-hoc environment and the ways this might be solved in the future. Therefore, we would like to discuss our project "Ad-hoc Awareness for Mobile Users" and we will often refer to the idea of ad-hoc awareness in order to make the whole discussion more clear.

We are discussing here the application that allows people to find other people or to find services based on some criteria defined by the user. The application supports mobility of the users and the ad-hoc way of communication meaning wireless connectivity without a need for access points, e.g. fully decentralization of nodes. The mobile user can find people or information in a fully decentralized environment. The application supports search for nodes and resources, messaging, and profile matching service where in profile the user can store the criteria for the resource he is looking for.

An ad-hoc system requires an appropriate transport, middleware and the application running on the top of the middleware. If the transport level and the middleware are provided already, the whole range of applications can be built on the top of the existing infrastructure. The main question today is: does this infrastructure that would allow us to build ad-hoc systems exist already?

On the transport level of the ad-hoc system there is a need for protocols that support an ad-hoc way of communication. Today there are several ad-hoc wireless technologies like WLAN (IEEE 802.11), Bluetooth, HomeRF, HiperLAN/2 on the market. The choice of the transport protocol depends on system requirements like network topology, desired accessibility, speed of nodes and distance between the participants, real-time interactions and grouping of network members.

Talking about WLAN that suits very well for the mobile community support type of applications, we need to highlight several issues related to this type of connectivity. Wireless LAN in the ad-hoc mode provides a unique feature to build a private network in a dynamic flexible way. The connectivity range is up to 100 m and the nodes can move around the network. According to the results obtained during our simulation of the ad-hoc network, about 1000 nodes can participate in the network receiving 80% of sent data, if message about 20Kb size is sent every 20 sec, and if nodes do not move. This is a satisfying result, taking in consideration the wireless aspect of the connectivity. But the quality significantly goes down with the introduction of mobility – as soon as nodes start to move around the network, the number of participants must be significantly reduced in order to provide the same level of connectivity. The speed influences the quality a lot, and this is a killing factor for mobile networks. Also the performance of TCP IP goes significantly down, due to the loss of packets and the need for redundant re-sending of those. In other words, the system does not scale due to the scalability problems of the transport layer. And this fact causes the problems on the application level of course, because it is not possible to build a system that scales on the top of transport level that does not scale.

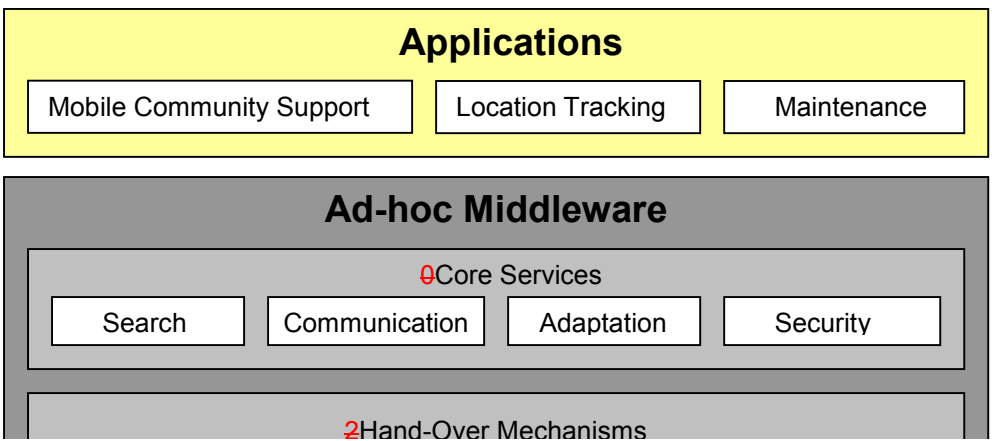


Fig. 1. Ad-hoc communication can be separated in three layers of protocols, middleware and applications.

Peer-to-Peer as a Middleware Platform for Ad-hoc Systems

If we take ad-hoc systems and their features that have been analyzed in the beginning in one hand, and we take peer-to-peer technology and the functionality that peer-to-peer systems are targeting on in another hand, then we can see that they have very much in common. We would like to show that peer-to-peer platforms such as JXTA might become a middleware for ad-hoc systems that will enable wide range of ad-hoc networking in the future.

Peer-to-Peer networking is an adaptive, self-configuring network, which does not rely on central servers. This is a type of network in which each workstation has in principal equivalent capabilities and responsibilities. Peer-to-peer networking is based on three blocks of basic functionality that enables "equal"-art of communication between nodes in the network: discovery mechanism, messaging and a security trust concept.

Discovery mechanisms

A discovery mechanism (search) allows peers to find other peers and context in the network. Because the centralized look-up service can be unavailable, peer-to-peer systems often provide a fully decentralized search mechanisms (Gnutella [So01], JXTA [Gon01]) or a hybrid semi-decentralized search (JXTA). It depends on a network size and topology (see below).

Peer-to-Peer technology allows for efficient use of resources and robustness in the decentralized environment. It provides mechanisms for finding required devices and resources in the network and to interact with them directly (search and messaging mechanisms). Existing distributed networking technologies apart from Peer-to-Peer (such as DCOM, CORBA, Web Services) currently rely on a priory known directory servers (registry, naming server, UDDI). That requires pre-configuration and limits the flexibility and ability to dynamically react to the change of the environment of the system. Distributed auto-configuration technologies such as Jini also rely on directory servers. Peer-to-peer concept of distributed discovery (plug and play) implemented in UPnP technology [GS00] and in JXTA search mechanism is more suitable for self-organizing networks.

As we have seen, search mechanisms are a key element for efficient and productive self-organization of the ad-hoc systems. The simple case is that only the peers shall be searched that are directly connected to the peer launching the search. This task can be completed in a sequential or even parallel way, depending on the available computing and network resources.

More interesting is the wide area search that comprises peers that cannot be "seen" directly. Several approaches have been realized in existing platforms that either are built on top of the network topology directly or target at matching of profiles or content.

Building on top of the network topology means that the search is related to the degree of decentralization of the network. If there is a central instance where the desired pieces of information are deposited, it can and should be used for the search, for this certainly accelerates the search considerably. In the pure decentralized case, when there is no such server, the search is carried out from one peer to the next or to all that are accessible from it, respectively. This technique is used for example in Gnutella. To avoiding an over-flooding of the network with search requests, each of these requests is assigned a maximal depth, the search horizon. Further optimizations

can be done based on the reduction of the depth by introducing connection weights, say according to the number or kind of hops, the round-trip times or similar.

The content-based search follows a different approach. One option there is to designate some peers and save hash-tables about the contents of their environment on them. During a search, only the peers that carry such tables, called the "super peers", need to be queried. This technique is a hybrid solution in the sense that it transfers concepts from the client/server world to a decentralized environment. The disadvantage of hash-tables is that only content with a few attributes can be processed and encoded reasonably. More complex information – as it is common in real-world situations – can hardly be managed this way.

An alternative is to avoid searching a peer completely but to rely on it declaring the relevant information on its own. This search technique follows the publisher/subscriber design pattern; each peer published the services it wants to offer to other peers on some sort of "advertisement board". A search algorithm can limit itself to reading these boards and need not waste time searching through the entire peer.

We can also combine both techniques. If all peers describe their offers on "advertisement boards", super-peers can query the services provided by peers in their environment and cache them. In a global view, the search is carried out in a decentralized way from one super-peer to another; in fact a much sparser P2P network than the original one must be combed. Seen locally the search is centralistic because the super-peer plays a server role for the other ones. Such an algorithm is part of the P2P platform JXTA, for example.

The coexistence of different approaches makes clear that there is not such thing like an optimal search strategy. Like in other search problems it depends to a large degree on the application, i.e. on the kind of information that we look for.

Messaging Mechanisms

A messaging mechanism (JXTA, Jabber [Ora01]) has several problems to overcome in the peer-to-peer context. The first one is a multi-hop routing, which is the problem of the mobile environment. The second one, is disruptive connectivity and firewalls. Current peer-to-peer platforms provide caching mechanisms (relaying) for overcoming firewalls problem. Caching mechanism allows the system to store information about other peers in the caching-service of the local area network that is used for accessing those peers from another local area network.

The multi-hop routing for the messaging in the ad-hoc environment is a hot topic. There is a need for algorithms that would allow for efficient multi-hop routing, although taking into consideration the mobility aspect and power consumption at the end devices. As long as multi-hop routing is not supported on the transport layer, the network can be built today only based on the connectivity within the transmitting range of the hardware (e.g. 100 m for the wireless LAN). If the node is out of the range, there is no possibility to send a message, although a rule of transitivity could be applied in practice – if node A sees node B and node B sees node C, this implies that node A sees node C. But for that a smart routing algorithm is required. The question is how to send the message avoiding loops and infinite routes? Choosing a node that should be the next node the message is sent to, certain rules should be applied. One of the considerations is to send the message to the node of high potential that is chosen from all available nodes also applying the ad-hoc rules. The node with high potential is just a node that is capable of resending the message. At the same time the route must be cached at each node of the route in order to record the way for further usage like sending the acknowledgement for the message back. But the problem with the one-route algorithms is that the branch can come to end somewhere and no nodes will be available at some point. That means the message won't reach the goal but will be lost in the network. These considerations make clear that multi-hop routing cannot simply follow the techniques that are employed in wire-line networks, but should be based on a multi-way approach that is an optimal one for non-deterministic system, where the behavior and the state of the system cannot be predicted, so the probability of delivery of the message with the one-way algorithm is very low.

Open Issues

A security trust concept is also provided by peer-to-peer technology, as a concept of groups where all peers are trusted within one group. This group concept (JXTA, Groove [SO00]) allows for solving the problem of decentralized initial trust establishment. Anyway, the trust establishment is still an issue for open networks. The trust of the single participant of the network is based on the experience of the node and gathering the experiences together requires an algorithm for distributed voting.

Also flexibility is often supported by easy or no configuration required from the user of the system. And this self-configuration mechanism is also available in the peer-to-peer concept (UPnP, JXTA).

There are still several open issues that need to be solved before the ad-hoc networking will become a reality. As we said, ad-hoc systems require dynamic adaptation capability. This includes location adaptation, service adaptation to a device, taking in consideration context awareness information. These features are not supported by any of the existing Peer-to-Peer platforms.

Also mobility aspect is not fully supported by Peer-to-Peer platforms. The addressing issue has been partly solved in some platforms. For example, in JXTA the peer receives a peer_id, which is an abstraction of the peer,

used to address requests to this peer. Peer_id is independent from IP-addresses or any other network-based addressing mechanism. The existing mechanisms of "on the fly" address assignment like DHCP, autoIP, mobile IP also helps to minimize the configuration efforts. But name to address mapping is not really solved because the problem of finding an exact peer is still a problem in peer-to-peer networks. Another problem is a hand-over when changing the network or type of the network – vertical and session hand-over. The peer-to-peer platforms today are not able to handle these situations. If the connectivity changes from WLAN to Bluetooth the system today is still not able to adapt to this change.

Conclusion

Ad-hoc systems provide a dynamic, flexible way of networking. There are several technologies that enable partly ad-hoc systems today, providing features such as dynamic discovery, wireless connectivity and information exchange. But in reality ad-hoc systems have a lot of limitations and do not provide the majority of issues they are targeting. We were looking at different layers of ad-hoc systems, trying to understand where is a hidden potential and where are the blocking points. We wanted to investigate whether peer-to-peer platforms can become a middleware for ad-hoc systems taking in consideration the wireless connectivity aspect.

Peer-to-peer technology provides the main mechanism for ad-hoc networking. However such features as dynamic adaptation, mobility and flexibility are not fully supported by the existing Peer-to-Peer platforms. We should keep in mind that most of them are functional requirements and can be introduced when needed. One of the major show-stoppers today is a multi-hop routing where no really good algorithm has been introduced yet. Looking at wireless technologies, we can say that they are not really scalable today and that, of cause, leads to the problems on the application level of ad-hoc systems. Anyway, providing a "smart" support of wireless technologies, this problem could be also partly solved. Concluding our discussion, peer-to-peer is able to become the middleware that enables ad-hoc networking, but there are some open issues that need to be solved both on the transport level as well as leveling the peer-to-peer platforms.

It should also be clear from the discussion that advancements in peer-to-peer networking are closely and mutually related to the ones in ad-hoc networking. With other word: In the beginning was Peer-to-Peer, and the Peer-to-Peer was with ad-hoc.

References

[BFS01] A. Buttermann, A. Franz, P. Sties, S. Vogel: "Ad hoc Networking – Technology and Trends", Center for Digital Technology and Management, 2001.

[Gon01] L. Gong: "Project JXTA : A Technology Overview", Sun Microsystems, 2001.

[GS00] M. Gitsels, J. Sauter: "Profile-based Service Browsing – A Pattern for Intelligent Service Discovery in Large Networks". In *Proceedings OOPSLA 2000,* Workshop on the Jini pattern language, Minneapolis, October 2000.

[GSG01] M. Gitsels, J. Sauter, S. Goose, G. Schneider: „Enterprise on Air : Universal Mobility for Enterprise Users", UbiComp 2001.

[MGL00] S. Marti, T. Guili, K. Lai, M. Baker: "Mitigating Routing Misbehaviour in Mobile Ad Hoc Networks", Mobicom 2000, Boston, 2000.

[Ora01] A. Oram, Hrsg.: "Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology", O'Reilly, Sebastopol, 2001.

[PSG00] T. Pham, G. Schneider, S. Goose: "A Situated Computing Framework for Mobile and Ubiquitous Multimedia Access Using Small Screen and Composite Devices", in Proc. of the ACM International Conference on Multimedia, Los Angeles, USA, October, 2000

[Sch01] M. Schmitt: "Subscriptionless Mobile Networking – Anonymity and Privacy Aspects within Ad hoc Networking", University of Siegen, 2001.

[SO00] P. Suthar, J. Ozzie: "The Groove Platform Architecture", Groove Networks, 2000.

[So01] A. Sotira: "What is Gnutella?", Gnutella.com, 2001.

[Wei93] M. Weiser: "Hot topic: Ubiquitous computing", IEEE Computer, pages 71-72, October 1993