ISSN 0930-5157 B 5713 F

[MH03] *Andreas Mauthe, David Hutchison:* **Peer-to-Peer Computing: Systems, Concepts and Characteristics;** Praxis in der Informationsverarbeitung & Kommunikation (PIK), 26(2/03), Juni 2003, S. 60-64.

2/03 April - Juni
26. Jahrgang 2003

raxis der nformationsverarbeitung und ommunikation

Fachzeitschrift für den Einsatz von Informationssystemen

P1K

Peer-to-Peer Computing: Systems, Concepts and Characteristics

Andreas Mauthe, David Hutchison
Computing Department
Lancaster University
Lancaster LA1 4YR
UK
Email: (andreas, dh)@comp.lancs.ac.uk

Abstract: Peer-to-peer is a technology concept applied at different levels of the systems architecture. Its main characteristics are direct interaction and data exchange between peer systems. It is the basis for decentralised distributed computing. The concept is widely deployed in different contexts and no formal definition exists. This paper gives an overview of the different areas peer-to-peer technology is used and introduces the main characteristics of peer-to-peer systems. It also discusses the issues and problems encountered when deploying peer-to-peer technology.

1 Introduction

Peer-to-peer (P2P) has become a buzzword that subsumes concepts used in communication technologies, distributed system models, applications, platforms, etc [1][2]. It is not a particular initiative, nor architecture or specific technology; it rather describes a set of concepts and mechanisms for decentralised distributed computing and direct peer-to-peer information and data interchange. Peer-to-peer mechanisms can be found at different level of the system architecture. Most prominent at present are peer-to-peer applications for content and information exchange in the Internet such as Napster, Gnutella and eDonkey [1][2] [4][5].

Although the term peer-to-peer refers to a number of generic ideas and mechanisms, the idea of peer-to-peer is applied in different contexts and hence peer-to-peer systems do not necessarily have many characteristics in common; neither do they have to adhere to a determined set of attributes. A formal definition of peer-to-peer computing does not exist. However, there are features many peer-to-peer systems share (such as the lack of a fixed hierarchical client server relationship, the autonomy of the peering (sub)-systems, the fact that they have independent lifetimes, and the direct information exchange between peers). The peering structure allows resource sharing between the participating entities. Resources in this context can be computational resources (such as processing cycle, bandwidth or storage space) and data resources (e.g. content files).

The peer-to-peer technology can help to reduce system costs and allows cost sharing by using existing infrastructures and bundling resources from different sites. Resource aggregation adds value beyond the mere accumulation of resources. Peer-to-peer systems are for instance more resilient because of their distributed and non-hierarchical nature. Peer-to-peer technology and its potential is still being discussed (cf. [1] [6]). However, there is a clear indication that peer-to-peer technology can have considerable impact on applications and distributed systems.

Although the concept of peer-to-peer has been around for a long time (e.g. it has been used in the Internet since its invention) only with the advent of peer-to-peer applications it gained a larger visibility and became more widely known. Therefore peer-to-peer is very often associated with applications more than with technical concepts.

In the following peer-to-peer computing is discussed and structured according to the different areas of peer-to-peer concepts are applied in. Subsequently the characteristics of peer-to-peer computing are discussed. This discussion states the characteristics that are regarded as the key features of peer-to-peer systems.

2 Peer-to-Peer Areas

The areas the concept of peer-to-peer is currently most heavily used is file and content sharing, collaboration support, distributed computing, communication and platforms. It can be found in methods, algorithms and communication schemes.

2.1 Peer-to-Peer File and Content Sharing Applications

Among the systems that feature most prominent in the peer-to-peer domain are user applications running on top (or at the edge) of the Internet allowing a large group of users to interact and share resources. Most popular are file or content sharing applications such as Napster, Gnutella, Mojo Nation, eDonkey and Freenet. Napster was the first major system enabling the direct exchange and sharing of content. While the actual exchange of content in Napster is between peers, the discovery of the peers, however, is highly centralised (i.e. it is stored in a central directory) [2][4].

Gnutella provides a purely distributed file sharing solution without a central node. In the strict sense Gnutella is not an application but a protocol used to search for and share files. To find content and other peers a user has to know the IP address of at least one other Gnutella node[1]. A node issues a query for a file by sending it to all other nodes known to it. If a node cannot serve a request it can forward it to other nodes. The query travels the Gnutella network until the file has been found or its time-to-live has been reached [1][4]. Effectively this discovery mechanism floods the network, which can cause scalability problems (although no major incidences have been reported so far). Another issue in Gnutella is free riders, i.e. users that do not contribute but just take content from other users. This has been confirmed in a study where measurements showed that 66% of the users do not offer any content at all and 73% offer less than 11 files for download [7].

Mojo Nation is a peer-to-peer content exchange application that introduces a virtual currency (so called Mojos) to counter free riders. This currency is also used as incentive to contribute resources (such as storage space and content). The peers in Mojo Nation can have different roles, i.e. Block Server (provides storage), Content Tracker (content search services), Publishing Agent (content publishing service), and Relay Service (forwarding service). The content is split into blocks and distributed throughout the Mojo Network. Hence, a block server only hosts part of the content but not the entire file [8].

Another popular file sharing system is eDonkey. The special feature of eDonkey is that it identifies the files using an MD4 based hash value and the file size. This method allows identifying files with the same content but slightly different names. It also enables the download of the content from different source files and hence increasing the net download rate [5][9].

Freenet is also a file/ content sharing system. The primary goal of Freenet is to make its use completely anonymous. Neither the users requesting nor those placing files in Freenet can be

[1] Other Gnutela nodes are published on a Web site. Hence there is an important component that does not use peer-to-peer technology within the context of Gnutella.

identified. Further, an operator of a Freenet node is not able to determine what data is stored on its local disk. Freenet is completely decentralised and represents peer-to-peer in its purest form [1].

2.2 Collaboration Support

Peer-to-peer collaboration and communication support is provided by systems such as Centrespan, Jabber, AIMster, Magi and Groove. These systems allow collaboration between users without the use of a messaging server. The simplest systems are only used for the exchange of messages. More sophisticated systems also allow joint authoring of documents, graphics, slides, etc. Games using peer-to-peer technology are also regarded as collaborative application since they also support interaction amongst users.

Groove is a system that provides a variety of applications for communication, content sharing (files, images and contact data), and collaboration (i.e. group calendaring, collaborative editing and drawing, and collaborative Web browsing) [10].

Magi is a peer-to-peer infrastructure platform that supports the implementation of secure, cross-platform collaborative systems. The core services of Magi include a Communication Service, an Event Service, a Buddy Manager and an Access Controller. It relies on DNS (Domain Name System) as a directory for the IP addresses of the Magi instances [1].

In general, events and message exchange taking place in a peer-to-peer collaborative system are relayed instantly to all other members of the peer-to-peer group. Issues to be considered are fault tolerance and real-time constraints. The former is linked to reliable group communication whereas the latter refers to interaction constrains. Group and multipeer communications research have actually addressed these problems in the mid 1990s already [11].

Collaborative peer-to-peer systems share the problem of locating and addressing peers (respectively the peer-to-peer group) with all other peer-to-peer application areas. In contrast to the others, however, communication here is mostly synchronous amongst an identifiable group (although specific users might remain anonymous). Hence the addressing problem is related to addressing issues in group and multicast communication.

2.3 Distributed Computing

A distributed system has been defined as a computer system in which several interconnected computers share the computing tasks assigned to the system as a single entity [12]. Such systems are for instance clusters or the GRID. Generally, systems that aggregate resources from a number of networked computers to achieve better processing and scalability benefit most from using peer-to-peer technology. However, while the resource usage in this case is peer-to-peer, there is very often a central instance that manages and co-ordinates the computational process.

A prerequisite for distribute computing is that tasks can be split into sub-tasks, which can be processed independently from each other. Interaction between peer systems processing sub-tasks should be kept to a minimum. Suitable processes are Single-Process-Multiple-Data and multiprogramming problems. Such problems can be found within science (physics, biotechnology) and finance where computation intensive calculations have to be carried out [1].

One of the most popular distributed computing applications is SETI@home, a project that aims to find prove of extraterrestrial intelligence [13][14]. Computers connected to the SETI@HOME system process data collected from a radio telescope. Their task is to find and identify any possible signals from intelligent populations outside the solar system. A database server controls the operations; the peers operate effectively as clients that get their jobs from this server. Hence peers in the context of this project refer to computing systems that make their resources available to others but that are still centrally controlled.

2.4 Platforms

There is an increasing trend away from native operating systems towards middleware platforms as application hosting environments (such as Java Virtual Machine or Web browsers). These platforms provide a largely operating system independent development environment and also offer high-level functions to system developers. Some of these platforms provide peer-to-peer support and use peer-to-peer mechanisms. Peer-to-peer components used in this context are for instance naming, discovery, communication, security and resource aggregation.

The two most comprehensive platforms supporting peer-to-peer at present are Sun's JXTA [15][16] and Microsoft's .NET [17]. JXTA can be considered as successor of Jini, an early attempt of Sun to introduce the concept of peer-to-peer platform. It provides an open general-purpose network programming and computing platform for distributed applications running on a variety of devices (including handheld, set-top boxes, special household equipment, etc.). It takes a layered approach and provides basic mechanisms as well as higher-level services. The three components associated with JXTA are JXTA Core, JXTA Services and JXTA applications. The services include discovery, authentication and management. A more high-level service is the Content Management Service that allows JXTA applications to share, retrieve and transfer content within peer groups. Further services are naming, routing[2] and indexing. JXTA provides some applications such as the command line interface JXTA Shell and InstantP2P (which gives chat capabilities).

The second major platform using peer-to-peer technology is MS .Net. The goal of .Net is to enable users of multiple devices (from relatively powerful PC to mobile devices) to access Web services using existing Web standards and protocols such as XML (Extensible Markup Language), SOAP (Simple Object Access Protocol), UDDI (Universal Description Discovery and Integration protocol) and WSDL (Web Services Definition Language). A central design goal of .Net is the de-componentisation and decentralisation of distributed services. Service discovery is done through UDDI whereas the service itself can then be accessed as Web services. MS .Net is not a pure peer-to-peer platform but incorporates some of the principles and concepts of peer-to-peer computing in its design.

2.5 Communication

Peer-to-peer communication issues arise in various contexts. Communication is for instance an inherent aspect of every distributed application. The challenge here is to deal with the problems associated with the dynamic nature of the peering entities.

Within communication itself peer-to-peer technology is a relatively old concept. Network topologies where the nodes are connected directly (i.e. meshed topologies and one-to-one connections) have always been part of communication infrastructures. Flooding or network broadcast are also technologies that support the communication in an anonymous peer group. Network multicast is another peer-to-peer communication mechanism where the user group is known. Group communication and multipeer communication models apply peer-to-peer principles at the transport layer [11].

In order to provide communication support to peer-to-peer applications one issue is addressing and routing. This should work for highly dynamic groups and also guarantee some degree of anonymity.

[2]Note, there is no support for request propagation, requests are sent directly to other peers.

Concepts such as document routing and custom naming have been proposed in this context but are limited to certain application areas.

3 Peer-to-Peer Characteristics, Concepts and Mechanisms

Peer-to-peer is not a well-defined and unambiguously used term. Though, definitions for specific areas have been proposed they usual do not cover all aspects conventionally subsumed under the notion peer-to-peer. For instance Shirkey defines peer-to-peer as a class of applications that takes advantage of resources (such as storage, processing cycles, human presence) at the edges of the Internet (in [18]). This definition concentrates on the application aspect and therefore does not include concepts that are used within another context.

Peer-to-peer should be regarded as a set of concepts and mechanisms that enable distributed, decentralised computing. Systems that share characteristics associated with the peer-to-peer concept are therefore called peer-to-peer systems. As in the case with family resemblance, any two systems might not have anything in common. However, they still belong to a group that shares (as a group) enough features to consider them peer-to-peer system[3]. In the following a number of characteristics are discussed that are conventionally attributed to peer-to-peer systems. This does not imply that every peer-to-peer system has to comply with a fixed number or certain (sub-)set of these characteristics. They are general features that can be used to identify peer-to-peer systems.

3.1 Structural Characteristics

One of the major concepts of peer-to-peer computing is *decentralisation*. This includes distributed storage, processing, information sharing, etc. Even control information can be held in a distributed manner rather than centrally. The advantage of decentralisation is an increased extensibility, higher system availability and improved resilience. On an application level decentralisation can also imply a transferral of ownership and control (of data, information and computational resources) to the application users. However, there are also problems related to this property. In a completely decentralised and dynamic system it is difficult to get or maintain a global view of the system state. Also, the system also does not necessarily behave in a deterministic way. Depending on the number and kind of peers within the group certain request might be answered but this cannot be guaranteed. Another problem is how to join such a group and how to get to know the other peer group members. Further, interoperability between heterogeneous systems is a problem since they have different characteristics and might even haven different interfaces or use different protocols.

Peer-to-peer systems are said to be inherently scalable since bottlenecks caused by central instances or servers are largely avoided. The system scalability is usually restricted by the amount of centralised operations necessary. However, a larger number of control messages to co-ordinate the different peer instances might limit the scalability of a peer-to-peer system as well. Hence, it is proposed to use the term *extensibility* in this context since certain precautions have to be taken to scale up the number of peers (e.g. intelligent search and message distribution mechanisms have to be used to avoid flooding). However, even if central management and co-ordination is required this can be limited to a well-defined set of operations. Thus, systems can potentially grow very large since resources can be added

[3] In another context this problem has been philosophically investigated for instance by Ludwig Wittgenstein [19]. It should not be considered a weakness of a system or terminology if no accurate classification using a set of non-ambiguous criteria can be given. This rather reflects the nature of the subject area.

almost indefinitely. Though, another issue here is that especially in heterogeneous systems no performance guarantees can be given since it is not known which instance is serving a request.

Peer-to-peer systems are *self-organising* in a sense that the different system components work together without any central management instance assigning roles and tasks. In this context the degree of organisation in a system increases without an increase of any external governance or system control. The system structure as well as the internal organisation of a peer-to-peer system is ideally built without external control or influence. In self-organised systems it is difficult to determine the system structure or predict the system behaviour as long as no system-wide, governing policies apply.

Because of their structure and organisation peer-to-peer systems are inherently *fault-tolerant*. Since there is no central point of failure they can easily compensate the loss of a peer or even a number of peers. With the loss of peers the system performance might decrease but as long as there is a sufficient number of peers in the system it should still be operational. However, in dynamic peer-to-peer systems this also implies that there is a lack of consistency.

3.2 Operational Features

Generally peer-to-peer services and systems are relying on the functions and tasks provided by the peers. The availability of peers can vary considerably; some might be there for long periods whereas others are present for a short time only. Hence peer-to-peer systems are of *ad-hoc* nature. This allows the creation of groups on request or when required. However, it also implies that tasks that require a stable execution environment have to deal with fluctuating resources such as sudden disconnects or additions to the system.

The *performance* of a peer-to-peer system is *unpredictable*. It depends on the resources (e.g. processing power and storage but also data) made available by the participating peers. Another resource type not (entirely) under the control of the peers is network bandwidth. The lack of bandwidth can decrease the overall performance of a peer-to-peer system considerably despite an abundance of all other resources required to compute a request.

There are a number of measures that can improve the performance of peer-to-peer systems. Data replication can be used to increase the availability of data. Using caching techniques reduces path lengths and hence the number of messages exchanged between peers. The more copies there are (especially if they are strategically located within the system) the higher is the availability and system performance. Apart from replicating data, processes and services can also be replicated in order to increase availability and thus system performance. Finally, intelligent routing and network organisation can help to decrease the number of messages and therefore increase (or more optimally utilise) the available bandwidth. This kind of optimisation also requires knowledge about application and user behaviour.

3.3 Functional Characteristics

In distributed systems a main goal is to provide *transparency* to the application or user. Forms of transparency include communication, location, access and replication transparency for data and information. Further, the concurrent access of systems by multiple users, and user and client mobility should be kept transparent. In connection with resilience failure transparency is also an issue. Finally, the scale of the system (as long as this does not determine some of its functionality) should be kept transparent to the user. Most peer-to-peer systems provide these forms of transparency utilising communication system, middleware, platform, programming language, and protocol features. Transparency shields the application and user from the details of the underlying system. However, some information about the system structure might be required at this level. For instance if performance issues arise for instance due to equipment characteristics, bandwidth or system load this information should be passed on to the upper layers.

3.4 Protection and Security

A number of security and protection issues are linked to peer-to-peer technology and applications. Traditional *security mechanisms* to protect data and systems such as firewalls cannot protect peer-to-peer systems since they are essentially globally distributed. On the contrary, such mechanisms can inhibit peer-to-peer communication. Therefore new security concepts are required that protect systems from intruders and attacks while still allowing interaction and distributed processing in peer-to-peer systems. Data can be protected by the use of encryption schemes such as public-key-private-key encryption. In order to protect IPR (Intellectual Property Rights) while still allowing the public use of content, signatures can be added to the data (e.g. techniques such as watermarking and steganography). In a distributed computing environment code is executed on peer computers. On the one hand these computer systems have to be protected from malicious code that may corrupt the machine, on the other hand processing results and sensitive data being processed should be safe. There are a number of methods being developed to deal with these issues. For instance sandboxing, safe languages, proof carrying code and compilers that certify code to protect the host environment; and information flow theory and model checking to protect the data.

Other protection issues are related to *privacy* within a peer-to-peer system. Peer-to-peer systems are open by nature. The authorship, the publisher and consumer interactions (such as storage of material and query for certain topics) should be protected. Systems can be classified according to the privacy they provide. These privacy classes include Absolute Privacy, Beyond Suspicion, Probable Innocence and Probably Exposed [20]. There are a number of ways to protect the identity of participating entities. Multicasting and flooding can be used to protect the identity of a receiver/ requestor. Spoofing can be used to protect both, sender and receivers. A covered path can also be used in this context. Further, untraceable aliases can protect participants. Forcing data onto hosting nodes can protect publishers and authors (so called non-voluntary placement) [1]. Issues arising in this context are related to responsibility and liability when the origin of data and user interaction cannot be traced. Also, the reliability of any information or data is problematic. The later is addressed in some peer-to-peer systems by applying a reputation scheme.

Ownership is also an issue in the context of peer-to-peer systems. Costs for system maintenance, communication and storage are usually shared between the peers. Even the costs for content and information can be shared between the operators of the participating peer systems. While cost sharing is an advantage a new problem is who owns the system, the content stored on such a system and the results produced by for instance shared processing. So far peer-to-peer systems and the results they produce have been regarded as public domain goods and assets. This issue is also linked to responsibility and liability for the actions.

4 Peer-to-Peer Related Issues

Peer-to-peer technology has a number of advantages over conventional client server systems. They are extensible and inherently fault tolerant. In the application context the potential of peer-to-peer systems to ensure user control, anonymity and privacy is important. This is possible since peering systems have a greater degree of autonomous control over their data and resources. However, there are also a number of issues that have not been resolved. There is for instance no global view at the system level and it is difficult to maintain a consistent system state. Some systems therefore have a central instance that manages the peers and imposes policies. However central control might not be required. In the communication domain the idea of maintaining a common state in a peer group by applying policies that govern and co-ordinate the behaviour of peer systems has been discussed [21]. This concept could be explored further for peer-to-peer systems that require a consistent state. Policies can also be extended to cover performance and QoS parameters. Hence it would be ensured that a minimal quality is available making the system more reliable and deterministic.

Addressing is a problem that affects peer-to-peer systems at all levels. At the communication layer concepts such as multicast and any-cast support the communication between peers to some extent. The identity of individual peers does not necessarily have to be disclosed in this context. At the application level other concepts such as document routing are used to find content. The addressing problem depends on the context and further work is required to find the best solution for a particular addressing problem.

Protection and security is another area that requires further research. Current security mechanisms such as firewalls very often inhibit the exchange between peer-to-peer systems. Nevertheless, new security concepts are required in the context of peer interaction, data exchange, and remote processing and program execution.

Issues at the application level are related to trust, responsibility and liability. Peer-to-peer applications can ensure a high degree of privacy and anonymity. If such a system is not policed anybody can do anything without the possibility to catch the offender. This also raises the question of who might be liable in the case serious offences. Certain mechanisms are required to protect peer-to-peer applications from offending users. Concepts such as reputation mechanisms are being discussed in this context.

5 Conclusion

The peer-to-peer concept has been around for some time. It became recently more popular by the advent of peer-to-peer applications for content and information sharing (e.g. Napster, Gnutella) and distributed processing and resource sharing (e.g. SETI@home). These applications exposed how many resources idle and unused computers currently waste. This has attracted considerable attention and it is thought that the potential of such systems is immense. However, it has to be distinguished between resource utilisation through such systems, and the actual technology and concepts that enable distributed computing in this context.

Peer-to-peer is a very heterogeneous work and research area. Peer-to-peer concepts can be found at the application layer, in distributed systems and within the communication sub-system. There are no major research or standardisation initiatives that look at all aspect related to peer-to-peer technology and computing. The major application areas for peer-to-peer methods and technologies are Internet and Web based applications GIRD computing and distributed systems, data sharing, and collaboration. Peer-to-peer concepts are used because they increase scalability (when the known restrictions are taken into account), can improve performance and are inherently fault tolerant. Peer-to-peer systems are flexible and dynamic.

The term peer-to-peer is defined by its usage in different contexts and no formal definition exists. The application areas of peer-to-peer concepts are also too heterogeneous to clearly define a fixed set of attributes peer-to-peer systems have to adhere to. However, there are a number of characteristics many peer-to-peer systems share to realise the advantages of peer-to-peer computing. Hence they are characterised by how many of them they implement rather than by a specific well defined (sub-)set.

References

[1] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu: "Peer-to-Peer Computing", hp Technical Report, 2002.

[2] D. Liben-Nowell, H. Balakrishnan: "Observations on the dynamic Evolution of Peer-to-Peer Networks", Proceedings of the 1st International Workshop on Peer-to-Peer Systems, Cambridge, USA, 2002.

[3] Open Napster Project, "Open Napster Messages", http://openap.sourceforge.net, 2000.

[4] F. Kileng: "Peer-to-Peer File Sharing Technologies: Napster, Gnutella and Beyond", Technical Report, 18/ 2001 Telenor, http://www.telenor.no/fou/publisering/Rapp01/R18_2001.PDF, 2002.

[5] MetaMachin, "eDonkey2000", http://www.edonkey2000.com/, 2002.

[6] K. Kant, R. Iyer, V. Tewari, "On the Potential of Peer-to-Peer Computing: Classification and Evaluation", http://kkant.ccwebhost.com/download.htm, 2002

[7] E. Adar, B. Hubermann, "Free Riding on Gnutellas", First Monday Peer-Reviewed Journal on the Internet, http://www.firtsmonday.dk/issues/issue5_10/adar/index.html, 2000

[8] MojoTainment, "MojoNation Web Site", http://www.mojotainment.com/mojonation/index.html, 2002.

[9] O. Heckmann, J. Schmitt, R. Steinmetz: "Peer-to-Peer Tauschbörsen – Eine Protokollübersicht", *to apper in* Zeitschrift für Wirtschaftsinformatik, July 2003.

[10] P. Leigh, D. Benyola: "Future Developments in Per Networking", Equity Research, White Paper, Raymond James & Associates, 2001.

[11] A. Mauthe: "Multimedia Multipeer Communications", PhD Thesis, Lancaster University, 1998.

[12] Institute of Electrical and Electronics Engineers, IEEE: "IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries", IEEE, New York, USA, 1990.

[13] SETI@home Web Site: "SETI@home - The Search for Extraterrestrial Intelligence" University of Berkeley, http://setiathome.ssl.berkeley.edu/, 2002.

[14] K. Skawinski: "IT & SETI: The Role of Computer Technology in the Search for Extraterrestrial Intelligence", California Computer News, 2002.

[15] SUN Microsystems: "Project JXTA" Web Site, http://wwws.sun.com/software/jxta/, 2002.

[16] J. Gradecki: "Mastering JXTA – Building JAVA Peer-to-Peer Applications" Wiley Publishing Inc., USA, 2002.

[17] Microsoft Corporation: "Microsoft .NET Homepage" Web Site, http://www.microsoft.com/net/, 2002.

[18] A. Oram: "Peer-to-Peer – Harnessing the Power of Dsiruptive Technologies", O' Reilly & Assoc., 2001.

[19] L. Wittgenstein: "Philosophical Investigations", first published in Cambridge, UK, 1953.

[20] M. Reiter, A. Rubin: "Crows: Anonymity for Web Transactions", ACM Transaction on Information and System Security, 1998.

[21] A. Mauthe, J. F. d. Rezende, D. Hutchison, "N2N Connection Service: A Multipeer Service for Distributed Multimedia Applications", Expert Contribution to ISO SC6/WG4, ref-no. ISO SC6/WG4 4G11, 1996.

Acknowledgement: The authors gratefully acknowledge the sponsorship of Telekom Austria that enabled the project under which this work was carried out.