Otso Kassinen

EFFICIENT MIDDLEWARE AND RESOURCE MANAGEMENT IN MOBILE PEER-TO-PEER SYSTEMS
OTSO KASSINEN

EFFICIENT MIDDLEWARE AND RESOURCE MANAGEMENT IN MOBILE PEER-TO-PEER SYSTEMS

Academic dissertation to be presented with the assent of the Faculty of Technology of the University of Oulu for public defence in Auditorium IT116, Linnanmaa, on 18 November 2011, at 12 noon

UNIVERSITY OF OULU, OULU 2011
Abstract

Mobile peer-to-peer (P2P) networks have emerged as a substrate for distributed wireless Internet applications. With P2P systems, it is possible to share resources such as data storage space, media files, network bandwidth, or computing power among the devices, which participate in the network, without large and expensive centralised server machines. However, the special characteristics of the mobile environment such as the low computational power, changing network conditions, and limited battery life pose several challenges for the fluent operation of mobile devices in P2P networks; this is also in part affected by the complexity of distributed P2P systems.

Software development for mobile devices and the design of mobile networking systems are challenging due to the limited device resources and heterogeneous software platforms. Moreover, the energy consumption of a mobile device, and the network-wide routing efficiency that affects for example the resource lookup performance, depend on multiple variable parameters. P2P application development can be facilitated, however, by using middleware, which hides the complexity of networking from the application programmers.

The research contributions of this thesis can be classified into three categories: (1) Novel functionalities of mobile middleware are proposed. One of these is a cross-layer connectivity management framework, which aims to select the best combination of network technology entities in a specific usage situation; the selection is made by logic contained in a replaceable state machine. Another new functionality is a system, which installs a missing application for the user, when the user receives a session request from another user. A signalling system based on the cellular USSD protocol is also presented; the system facilitates the establishment of a P2P connection with a mobile device, whose IP-based network connection is off. Moreover, the suitability of the P2PSIP protocol for implementing wireless distributed services is analysed. (2) P2P-related measurement studies are presented. In them, the message routing efficiency of a P2PP protocol implementation and the network traffic load caused by the messaging are observed, and the energy consumption incurred by the same implementation in a mobile device is measured. In addition, a server-based testbed system used in these measurements is described. (3) Experience-backed guidelines for mobile middleware development are presented. These include practical instructions for software development on a restricted mobile platform, and guidelines and observations related to cross-platform software development.

Keywords: energy consumption, mobile middleware, mobile P2P networking, mobile software development, P2P lookup performance
Kassinen, Otso, Tehokkaat välikerrosohjelmistot ja resurssien hallinta mobiileissa vertaisverkkojärjestelmissä.

Oulun yliopisto, Teknillinen tiedekunta, Tietotekniikan osasto; Infotech Oulu, PL 4500, 90014 Oulun yliopisto

*Acta Univ. Oul. C 391, 2011*

Oulu

**Tiivistelmä**


Mobiililaitteiden ohjelmistokehitys ja mobiiliverkkojärjestelmien suunnittelu on haastavaa rajallisten laiteresurssien ja epäyhtenäisten ohjelmistoluokkien vuoksi. Lisäksi mobiililaitteiden energiankulutus ja koko P2P-verkon reititystehokkuus, joka vaikuttaa esimerkiksi resurssien hakutavuuteen, riippuu useista muuttuvista parametreista. P2P-sovelluskehitystä voidaan kuitenkin helpottaa käyttämällä välikerrosohjelmistojen, jotka kätkevät verkko-ohjelmistojen monimutkaisuuden.


**Asiastat:** energiankulutus, mobiiliohjelmistojen kehitys, mobiilit vertaisverkot, mobiilit välikerrosohjelmistot, P2P-hakutavuus
Preface

This work has been conducted at the MediaTeam Oulu research group, Department of Computer Science and Engineering, University of Oulu, Finland. The work as a postgraduate student took place in 2007–2011, but the research has continued since 2004, when my employment at MediaTeam Oulu began.

I wish to express my gratitude to my supervisor, Prof. Mika Ylianttila, for his excellent guidance and encouragement during this work; his support since the days when I was a research assistant was a major reason why I decided to pursue the doctoral degree. I want to thank warmly all my colleagues at MediaTeam Oulu. It has truly been a pleasure to work with you in the friendly environment of our workplace. Unfortunately, I cannot list the names of all of you here. Special thanks are due to M.Sc. Erkki Harjula and M.Sc. Timo Koskela for their long-time collaboration with me in practically all the research activities that have led to the writing of this thesis, and I also want to thank the other co-authors of the papers included in this thesis: M.Sc. Jussi Ala-Kurikka, M.Sc. Jari Korhonen, Prof. Jukka Riekki, and Dr. Junzhao Sun. I also want to give my thanks to M.Sc. Janne Julkunen, M.Sc. Juuso Ohtonen, Dr. Zhonghong Ou, and M.Sc. Jani Pellikka for the many nice days in the same office – and outside it.

I thank Prof. Sasu Tarkoma for agreeing to be the opponent in the public defence of my work, and I thank the preliminary examiners of this thesis, Prof. Jukka K. Nurminen and Prof. Peng Zhang, for their valuable comments on the content. I also want to thank Dr. Pertti Väyrynen for proofreading this thesis.

The work has been carried out in several research projects, the most essential ones having influence on the content of this thesis being Decentralized Inter-Service Communications (DECICOM) and Application SuperNetworking (All-IP). They took place in 2007–2010 and 2004–2007, respectively, and Prof. Mika Ylianttila was the project director and M.Sc. Erkki Harjula the project manager.

I thank warmly the various organisations, which have provided funding for my research work. The Infotech Oulu Graduate School has provided a four-year funding plan for a portion of the expenses of the research. The Finnish Funding Agency for Technology and Innovation, Tekes, has been a major funding provider in the several research projects; also the Information Technology for European Advancement 2 (ITEA 2) program has supported the research. Several companies, including Elektrobit, Ericsson, Futurice, IBM, Icecom, NetHawk, Nokia, Serv-It, and TeliaSonera, have participated in the projects in the roles of financiers and research partners. The following foundations, presented here in alphabetical order,
have granted personal scholarships for my work: Ahti Pekkalan säätö, Emil Aaltosen säätö, KAUTE-säätiö, Nokia Oyj:n säätiö, Oulun yliopiston tukisäätiö, Riitta ja Jorma J. Takasen säätiö, Seppo Säynäjäkankaan tiedesäätiö, Tauno Tönningin säätiö, Tekniikan edistämissäätiö, TeliaSonera Finland Oy:n tutkimus- ja koulutussäätiö, Ulla Tuomisen säätiö, and Walter Ahlströmin säätiö. I also want to thank any funding and collaborating organisations that are not explicitly mentioned.

Finally, I thank my family and friends for their continuous support throughout these years; again, I am unable to thank each of you individually here, but I would like to mention my parents Antero and Sylvi, my sister Ursa, and one special name, Elina.

Oulu, 2011

Otso Kassinen
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>Authentication, Authorisation, Accounting</td>
</tr>
<tr>
<td>ABC</td>
<td>Always Best Connected</td>
</tr>
<tr>
<td>ACPC</td>
<td>Agile Content Push Control</td>
</tr>
<tr>
<td>AJAX</td>
<td>Asynchronous JavaScript and XML</td>
</tr>
<tr>
<td>ALTO</td>
<td>Application-Layer Traffic Optimization</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ARPANET</td>
<td>Advanced Research Projects Agency Network</td>
</tr>
<tr>
<td>AS</td>
<td>Autonomous System</td>
</tr>
<tr>
<td>CAN</td>
<td>Content-Addressable Network</td>
</tr>
<tr>
<td>CAPNET</td>
<td>Context-Aware Pervasive Networking</td>
</tr>
<tr>
<td>CoAP</td>
<td>Constrained Application Protocol</td>
</tr>
<tr>
<td>CORBA</td>
<td>Common Object Request Broker Architecture</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>DC</td>
<td>Direct Connect</td>
</tr>
<tr>
<td>DCOM</td>
<td>Distributed Component Object Model</td>
</tr>
<tr>
<td>DECADE</td>
<td>Decoupled Application Data Enroute</td>
</tr>
<tr>
<td>DHT</td>
<td>Distributed Hash Table</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>DPM</td>
<td>Distributed Pattern Matching</td>
</tr>
<tr>
<td>DRM</td>
<td>Digital Rights Management</td>
</tr>
<tr>
<td>DSHT</td>
<td>Distributed Sloppy Hash Table</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
</tr>
<tr>
<td>Fi</td>
<td>Future Internet</td>
</tr>
<tr>
<td>FLTK</td>
<td>Fast Light Toolkit</td>
</tr>
<tr>
<td>FOX</td>
<td>Free Objects for X</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>GTK+</td>
<td>GIMP Toolkit</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HCon</td>
<td>Holistic Connectivity</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>IAT</td>
<td>Inter-Arrival Time</td>
</tr>
<tr>
<td>ICE</td>
<td>Interactive Connectivity Establishment</td>
</tr>
<tr>
<td>ID</td>
<td>Identifier</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
</tr>
<tr>
<td>IDL</td>
<td>Interface Definition Language</td>
</tr>
<tr>
<td>Acronym</td>
<td>Full Form</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IM</td>
<td>Instant Messaging</td>
</tr>
<tr>
<td>IMS</td>
<td>IP Multimedia Subsystem</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPC</td>
<td>Inter-Process Communication</td>
</tr>
<tr>
<td>IPR</td>
<td>Intellectual Property Rights</td>
</tr>
<tr>
<td>IPTV</td>
<td>Internet Protocol Television</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>ITEA</td>
<td>Information Technology for European Advancement</td>
</tr>
<tr>
<td>JNI</td>
<td>Java Native Interface</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>LOC</td>
<td>Lines of Code</td>
</tr>
<tr>
<td>LTE</td>
<td>Long Term Evolution</td>
</tr>
<tr>
<td>M2M</td>
<td>Machine-to-Machine</td>
</tr>
<tr>
<td>MANET</td>
<td>Mobile Ad-Hoc Network</td>
</tr>
<tr>
<td>MP2PP</td>
<td>Mobile P2PP</td>
</tr>
<tr>
<td>NAPI</td>
<td>Network-Assisted P2P Invocation</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>OSI</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer-to-Peer</td>
</tr>
<tr>
<td>P2PP</td>
<td>Peer-to-Peer Protocol</td>
</tr>
<tr>
<td>P2PSIP</td>
<td>Peer-to-Peer Session Initiation Protocol</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PnPAP</td>
<td>Plug-and-Play Application Platform</td>
</tr>
<tr>
<td>PPSP</td>
<td>Peer-to-Peer Streaming Protocol</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Resource Location and Discovery</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>RSS</td>
<td>Really Simple Syndication</td>
</tr>
<tr>
<td>RTCP</td>
<td>Real-Time Transport Control Protocol</td>
</tr>
<tr>
<td>RTP</td>
<td>Real-Time Transport Protocol</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>RTT</td>
<td>Round-Trip Time</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software as a Service</td>
</tr>
<tr>
<td>SDCCH</td>
<td>Stand-alone Dedicated Control Channel</td>
</tr>
<tr>
<td>SDP</td>
<td>Session Description Protocol</td>
</tr>
<tr>
<td>SHOK</td>
<td>Strategic Centres for Science, Technology and Innovation</td>
</tr>
<tr>
<td>SIP</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>SIP UA</td>
<td>SIP User Agent</td>
</tr>
<tr>
<td>SMS</td>
<td>Short Message Service</td>
</tr>
<tr>
<td>SOA</td>
<td>Service-Oriented Architecture</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SRC</td>
<td>Short-Range Communication</td>
</tr>
<tr>
<td>SS7</td>
<td>Signaling System No. 7</td>
</tr>
<tr>
<td>SSTPT</td>
<td>Scalable Star-Topology P2P Testbed</td>
</tr>
<tr>
<td>STUN</td>
<td>Session Traversal Utilities for NAT</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TLV</td>
<td>Type-Length-Value</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to Live</td>
</tr>
<tr>
<td>TURN</td>
<td>Traversal Using Relays around NAT</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunications System</td>
</tr>
<tr>
<td>UPIp</td>
<td>Universal Plug and Play</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>USSD</td>
<td>Unstructured Supplementary Service Data</td>
</tr>
<tr>
<td>VoD</td>
<td>Video on Demand</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>WAC</td>
<td>Wholesale Applications Community</td>
</tr>
<tr>
<td>WAP</td>
<td>Wireless Application Protocol</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local-Area Network</td>
</tr>
<tr>
<td>WMN</td>
<td>Wireless Mesh Network</td>
</tr>
<tr>
<td>WSN</td>
<td>Wireless Sensor Network</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>XML-RPC</td>
<td>Extensible Markup Language Remote Procedure Call</td>
</tr>
<tr>
<td>XMPP</td>
<td>Extensible Messaging and Presence Protocol</td>
</tr>
<tr>
<td>XOR</td>
<td>Exclusive Or</td>
</tr>
</tbody>
</table>
List of original publications

This thesis is based on the following original publications, which are referred to in the text by Roman numerals (I-IX):


IX Kassinen O, Harjula E & Ylianttila M (2011) Scalable star-topology server-array based P2P overlay network testbed. Proc 7th International ICST Conference on Testbeds and Research Infrastructures for the
Development of Networks and Communities (TridentCom ‘11). Shanghai, China: 8pp.

Papers I–IV are related to novel P2P middleware functionalities. Papers V and VI deal with software implementation techniques for mobile middleware. Papers VII–IX are about performance measurements in P2P networks. The author has also contributed to several other publications¹, including journal articles, conference papers, and technical documents such as Internet Drafts.

1 Introduction

“Any sufficiently advanced technology is indistinguishable from magic.”

– Arthur C. Clarke

1.1 Motivation and problem statement for the work

In recent years, versatile usages have emerged for two key technologies: mobile networking and peer-to-peer (P2P) networking. Today’s mobile devices are more powerful than desktop computers ten years ago and enable wireless network related scenarios that would not be possible with stationary computers. Popular P2P applications in everyday usage include, among others, personal communication systems such as Skype, delivery systems for streaming media such as Spotify, and decentralised retrieval systems for downloadable content such as BitTorrent. In P2P systems, the networked hosts communicate in a distributed fashion, as opposed to client-server systems where all communication occurs via a centralised server. In Fig. 1, the fundamental difference between client-server and P2P network architectures is illustrated.

Fig. 1. Client-server network architecture versus P2P network architecture.

The proliferation of mobile networking terminals and the success of P2P technologies are, of course, related to the phenomenal growth of the Internet.
According to a survey by Miniwatts Marketing Group\(^2\), the global number of Internet users increased 444.8% between the years 2000 and 2010, and in June 2010, the global number of Internet users reached nearly two billions, corresponding to 28.7% of the global human population. Mobile usage of the Internet has increased its popularity dramatically, and it has profoundly changed the way how mobile terminals are used: in 2010, Ericsson\(^3\) announced measurements indicating that mobile data surpassed mobile voice already in 2009 and the amount of mobile data traffic is forecast to double annually over five years.

P2P networking consumes a major part of Internet bandwidth globally. According to a 2008–2009 study by the Internet analyst company Ipoque\(^4\), different P2P protocols constituted 42.51% to 69.95% of all Internet traffic: the proportion of P2P traffic varied depending on the region of the world where traffic measurements were conducted, but in every region, P2P protocols generated more traffic than any other protocol category. Although the amount of P2P traffic may decrease in the future due to changes in the Internet usage patterns, it is safe to assume that P2P networking will be a major bandwidth hog for many years to come.

It can be stated that the emergence of P2P and mobile networking technologies has been an unforeseen revolution. However, advanced solutions are required both in the network infrastructure and on the terminal side in order to guarantee a satisfying user experience. These solutions include several optimisations that facilitate P2P usage with mobile devices.

The existing P2P systems and other network protocol based systems have, to a great extent, been enabled by the power of software. Networking protocols are just software systems with a specialised purpose. Needless to say, also nearly all functionality visible to the user on the application layer in a mobile device or in a stationary computer is realised with the help of software. Software is also used in nearly all pieces of the networking infrastructure such as routers and servers that are usually more or less invisible to the end-user.

Software engineering as a branch of technology is an approximately 50-year-old phenomenon. During this time, enormous advances have been made in computer science, programming languages, programming tools and

---

\(^2\) http://www.internetworldstats.com/stats.htm accessed on 24 February 2011
\(^3\) http://www.ericsson.com/thecompany/press/releases/2010/03/1396928 accessed on 24 February 2011
\(^4\) http://www.ipoque.com/ accessed on 24 February 2011
methodologies, and programming paradigms such as the evolvements from procedural towards object-oriented programming or from local software installations towards software-as-a-service. Software engineering has changed our world in profound ways. Data-intensive actions which previously required tedious manual work are facilitated by software systems. Moreover, to a growing extent, relevant information in practically any area of life is created in digital form or converted to it; thus, the ability to get benefit from the automated processing of data only increases.

From the viewpoint of networked applications and devices in a world of versatile hardware and software platforms, one especially interesting area of software engineering is middleware. Middleware has been created for hiding some or all of the complexity of the underlying network-related systems. Enormous complexity is often present in software systems: millions of lines of code, comprising systems of thousands of logical expressions, branching conditions, and data alteration instructions in the memory. The number of distinct possible states of a simple computer exceeds the number of atoms in the universe. Good programming practices tend to create simpler code. With middleware, in particular, it is possible to hide the networking-related complexity from the application developers. The effects of middleware in action may or may not be visible to the end-user. In any case, the user does not need to know about the existence of middleware.

However, from the viewpoint of the service infrastructure developers, it is not a trivial task to combine the three major technological enablers – P2P networking, mobile networking, and middleware systems – to produce an environment where applications can benefit from the P2P-based signalling system and from a multitude of mobile access networks, with only a minimal burden for the capacity of the underlying technical systems and a minimal effort for the application developers.

Thus, this thesis deals with different aspects of how to create efficient middleware for resource management in mobile P2P networking. Here, the word resource has a dual meaning. On the one hand, a resource is something that is shared in the P2P network, for example a multimedia file or a multimedia communication session established between two peers. On the other hand, resources are also those entities that are prerequisites for the functioning of the system: for example, the remaining battery power of a mobile device or the random access memory (RAM) used by the networking-related processes are resources. Middleware can contain functionality that is related to the management
of either or both kinds of resources; in any case, as a result of applying middleware, the mobile system should be more efficient in use and more appealing from the viewpoint of application developers.

1.2 Scope and methodology

The scope of the research work presented in this thesis can be divided according to the three main areas of technology studied: first, the novel functionalities that can be provided by mobile P2P middleware, second, software development techniques for creating middleware or middleware-based applications, and finally, measurements of the different performance metrics of the P2P networking systems.

The research questions to be answered in this thesis are as follows:

– What kind of P2P middleware-based techniques can be used for the management of connectivity and session resources?
– What kind of guidelines should be applied in the software development of mobile middleware systems?
– What kind of performance, in terms of message routing efficiency and battery resource consumption, can be expected from a distributed hash table (DHT) based P2P network with varying network parameters?

It should be noted that the term *P2P networking* refers to the usage of application-layer P2P overlay networks; thus, ad-hoc networks and other such physical-layer distributed networks are excluded from the scope of this thesis, except for some brief mentions in the literature review. The most closely observed protocol in the studies in this thesis is Peer-to-Peer Protocol (P2PP) on different computing platforms, both mobile and stationary, using a structured (DHT-based) communication model. Some P2P communication scenarios studied in this thesis include also unstructured communication or server-assisted communication between the endpoints, for example, using Session Initiation Protocol (SIP) based messaging.

Moreover, it should be noted that in this thesis the terms *mobile networking* or *wireless networking* refer to the usage of consumer-grade end-user terminal hardware and software in publicly available networks such as Global System for Mobile Communications (GSM), Universal Mobile Telecommunications System (UMTS), Long Term Evolution (LTE), IEEE 802.11 wireless local area networks (WLAN), or Bluetooth; thus, excluding for example military networking systems
from the scope. It should also be noted that this thesis has no emphasis on the management of radio resources of a mobile device: the radio-based communication systems are of course used by the studied P2P systems, but no improvements to the actual radio technology are presented. Furthermore, the term “middleware” has, for some people, connotations related to enterprise messaging systems, and while not excluding those systems, the sense in which the word is used in this thesis is more technical: not “between the workers of a business system” but rather “between the application and the network”.

Novel functionalities for mobile middleware are studied from the viewpoints of usage optimisation on the device-resource level and of qualitative improvements to the application scenarios enabled by the middleware systems. The proposed functionalities have differences from the viewpoint of the ease of deployment: some enhancements are such that a mobile peer can independently take advantage of them, and some require collaboration from remote nodes to be effective.

Guidelines for software development in middleware-related projects are given with software developers as the target audience. The special characteristics of mobile middleware systems and the related programming techniques and project-support facilities are paid attention to. The management of cross-platform mobile software projects is one area of interest: techniques for managing the complexity of program code for multiple target platforms are observed.

In the evaluation of the properties of P2P systems, the suitability of the studied systems for primitive recurring network operations and for certain specific kinds of applications is analysed, and details of the setup of P2P networking related experiments are explained. The properties of P2P networks are evaluated with mobile usage in mind, but the observations are often applicable to non-mobile usage as well. In the evaluations of mobile usage, examining and ensuring the possibility of using a mobile device as a fully functional peer in a P2P network is an important goal.

The methodology applied in the research is an empirical-constructive engineering approach for obtaining results related to the research items. The method is not restricted to only calculation-based simulations; while also those have been used, concrete software prototypes are an important part of the evaluation of the technologies, both as a proof-of-concept implementation and as an object in measurement studies.
1.3 Contributions of the thesis

Below, a brief overview of the contributions of this thesis and the original papers is given. The contributions are discussed in more detail in Chapter 3, where the contributions are also mapped to each of the original papers.

The work has produced three main contributions:

– mobile P2P middleware techniques for connectivity and session management;
– guidelines for the software development of mobile middleware systems; and

The functionality of mobile middleware is enhanced in the following ways. The internal working of the Holistic Connectivity system is presented. Holistic Connectivity is a framework that enables the dynamic selection of networking technology entities in the different layers of the protocol stack of the mobile device. The framework contains a state machine executor, which runs replaceable state machines that must be used to specify the context-based rules concerning when and how to switch the networking technology entities in the various layers.

The Agile Content Push Control system is presented for facilitating the distribution of mobile applications between end-users. The system proposes the installation of the application to the mobile device of user B, when user A proposes a user-to-user application session with user B. Thus, the installation occurs as a side-effect of application-related social interaction between the users.

The Network-Assisted P2P Invocation system utilises the signalling channel of the cellular network for initiating P2P network activities with a mobile device. The system allows the mobile device to stay disconnected from Internet Protocol (IP) based networks while still being available for IP-based P2P interactions whenever an incoming request for such an action is received in a non-IP-based message over the cellular network. The system thus saves battery power and facilitates network address translation (NAT) traversal towards the mobile device.

On a high level, novel functionalities for P2P middleware are analysed in a study on Peer-to-Peer Session Initiation Protocol (P2PSIP); in the study, the possibilities of using the technology as a substrate for versatile wireless distributed services are discussed.

Guidelines for mobile middleware development include the following:

Practical instructions for mobile middleware projects are gathered by observing the actual development work of a mobile middleware system, and these instructions are provided as a list of reusable guidelines for the different parts of
the development work, including aspects of protocol design, software architecture
design, facilitating of debugging activities, and project workflow management.
Cross-platform development of mobile middleware with the separate-builds
approach (one source code base, several different binary builds) is analysed in
order to find the best practices for such projects. The observations include, among
other things, considerations for taking into account the asymmetric difficulty of
the different target platforms of a cross-platform project. Many of the software-
development related guidelines are applicable to non-middleware projects as well.

Measurements of performance metrics in P2P networks are conducted with
the experimental protocol P2PP, whose development work has been merged into
Resource Location and Discovery (RELOAD), the chosen base protocol of
P2PSIP. By analysing the measurements conducted with actual systems,
benchmarking figures are obtained for use as the settings of parameters in
deployed systems, and an insight into the operation of the systems is gained. In
the measurements, the message routing performance under varying network
conditions, such as different levels of churn in the P2P overlay, is measured, and
the battery life of mobile peers taking part in a P2P overlay is evaluated, again
under varying network conditions. A server-array based experiment setup, used in
the measurements for managing P2P overlay simulations that involve
combinations of variable parameters, is presented.

1.4 Organisation of the thesis

The thesis is organised as follows: In Chapter 1, a short overview of the results
and the motivation for the research were given. The related work for the research
is presented in Chapter 2 in the form of a literature review, in which the state of
the art of the research on relevant technologies is examined. The literature review
encompasses P2P networking systems, mobile P2P networking optimisations, P2P
measurement studies, mobile middleware systems, and software development
techniques, including considerations for mobile software and cross-platform
software. In Chapter 3, the contributions of each of the original papers are
summarised. In Chapter 4, the thesis is concluded, and items for future work are
identified.
2 Related work

“If you copy from one, it’s plagiarism. If you copy from two, it’s research.”
– Wilson Mizner

2.1 P2P systems

In the following subsections, the existing research on P2P networking is reviewed. This part of the literature review includes taxonomies for P2P systems, challenges and solutions for the challenges in P2P system development, concrete examples of P2P systems, studies on the efficiency of mobile usage and its optimisations, evaluation methods and evaluation results for P2P systems, and a glance on session management technologies.

2.1.1 Taxonomies of P2P systems

P2P networks are distributed systems, where the nodes interact directly for the purpose of exchanging resources instead of using centralised server nodes (Androoutsellis-Theotokis & Spinellis 2004). Typically, the nodes in a P2P system provide services to each other in such a way that a node can be both a client and a server at the same time, and thus, the term servent (server-client) has been used for peers (Schollmeier 2001). This is somewhat similar to the term prosumer (producer-consumer), which refers to users or nodes that can be both providers and consumers of services at the same time, although the term prosumer is rarely used in P2P-related contexts. Typically, in P2P networks, the different peers have symmetric capabilities in the qualitative sense, while a peer can have more or less capabilities than another one in the quantitative sense. Moreover, P2P systems often consist of nodes that do not have permanent Internet connectivity, may stay offline for unpredictable periods of time, and may be more practically referred to with their P2P-level identifiers than, for example, with their actual IP addresses.

P2P networks are a versatile substrate for different kinds of distributed applications and services. Examples of these application areas include the sharing of media files, the sharing of presence information, document management, collaboration within office-work teams, the provision of storage space (as differentiated from space for file sharing), bandwidth sharing on suboptimally utilised network routes, the sharing of processor cycles (Schoder et al. 2005),
enabling real-time communication, creating database systems with relational-like querying capabilities, and creating service support systems such as application-level multicast systems (Androutsellis-Theotokis & Spinellis 2004). Perhaps still the services most readily associated with P2P technology are the various well-known examples of P2P-based content distribution systems. The distributed content may be illegal, or the distribution may be legally authorised by the owner of the intellectual property rights (IPR).

Not all distributed systems are P2P networks. Several authors mention grid systems as an example of systems that should not be confused with P2P networks despite their similarities. According to Lua et al. (2005), the difference is that grid systems consist of more established sets of connected systems and share a more reliable set of resources than P2P networks. According to Eberspächer & Schollmeier (2005), grid systems are a “more structured approach” with their clearly focused purpose, which is distributed computation and simulation.

In this thesis, the terms P2P network and P2P system are used interchangeably. Sometimes the term overlay network is used to emphasise the fact that P2P networks exist in the application layer and are overlaid on top of the lower-layer networks, which provide the basic interconnection functionalities of the Internet. The word node is used as the generic name for peers and clients. A peer is a node that contributes to the management of the P2P network by providing some kind of services, while a client is a node that only uses services without providing them. These two words are adopted from the P2PSIP terminology. Not all P2P systems have the concept of clients. Moreover, not all authors use the same terminology; thus, in the literature, the exact meaning of words such as “peer” may be different, or a given concept can have another name.

There are several ways of classifying P2P systems. Two prominent approaches are presented in the following: Firstly, generations of P2P systems can be identified. This approach suggests that there is a chronological order, in which the newer types of P2P systems are somehow more advanced than the earlier types. The separation of generations is somewhat arbitrary, and the approach has the drawback that the technical content of names such as the “second generation” is not self-explanatory.

Zhang et al. (2006) identify three generations of P2P systems. The first generation is built around centralised servers that are responsible for indexing. The second one consists of systems where message propagation is based on flooding. The third one refers to structured systems where each node has a numeric identifier and the identifiers are arranged in a topology for the purpose of
message routing. Foster & Iamnitchi (2003) define the three generations of P2P systems essentially in the same manner. In a study by Karrels et al. (2009), the surveyed P2P systems are classified in more than two categories, but only two numbered generations are identified. The first generation includes unstructured, broadcast-based systems, and the second one includes “content structured” systems where the network acts as a distributed database storing a set of data elements under unique keys and where the network structure is modified in order to organise the stored content. The authors seem to suggest implicitly that the “communications structured” P2P networks, which are designed to locate nodes by key instead of data by key and where the ability to locate data elements is built on top of the node-locating functionality, are the “third generation”.

As the second way of classification, P2P systems can be categorised according to their named technical properties, without forcing the notion that there exists a continuum of generations. The structure, or the lack of structure, in a P2P overlay network is a recurring theme in publications on P2P network classification. Many authors classify P2P overlay networks into unstructured and structured ones; these kinds of categories were already briefly discussed above in the context of P2P network generations. Other named technical properties in P2P system classifications include the type of indexing used for nodes or resources in the network, the type of searching operations which is actually implied by the selected indexing method, and the content delivery method used in the network. Moreover, it should be noted that while several authors use the same named technical property as a tool of classification, their selected categories within that technical property are not necessarily the same; for example, for “structure”, some authors see only the categories “structured” and “unstructured”, while others see a larger number of choices.

Before delving into the literature review on named technical properties in the classification of P2P systems, the meanings of a few key terms as they are used by the author of this thesis are defined. The selected meanings represent a rough consensus on the P2P-related terminology in the literature.

**Unstructured** systems are those, where the routing mechanism does not follow deterministic rules that would force a specific node or resource to a specific place in the topology; instead, the connections between the nodes are created in an unpredictable way. **Structured** systems are those, which are based on DHTs. In a DHT, the identifiers of data resources share the same numeric address space as the node identifiers (IDs). The ID address space is organised, on the logical level, into a geometric structure, for example a ring (circle) or a
multidimensional cube. Details depending on the particular DHT algorithm, each resource is deterministically stored on the node, whose ID is nearest to the ID of that resource. Thus, also the searching of a resource follows essentially the same rules as the searching of a node and produces deterministic results as long as the underlying systems do not fail. In a nutshell, the purpose of DHT systems is to map a specific key (an ID) to a specific value stored in the network. The numeric IDs of nodes and resources are random and are thus uniformly placed in the address space (Lua et al. 2005). The numeric IDs are often obtained by computing a unique cryptographic hash of a human-readable string that describes the particular node or resource.

Centralised indexing is an indexing system, where the index of the nodes or resources in the P2P network is stored in a centralised location, in practice on a server machine. Distributed indexing refers to systems, where the information concerning the index is stored in a distributed way on the nodes themselves. In hybrid indexing, indexing is the responsibility of super-peers, which act as information hubs for the benefit of the ordinary nodes.

In the following, some P2P system classifications based on named technical properties are presented. Brands & Karagiannis (2009) propose a six-level framework, where the main type of service, centralisation of the index, network structure, deployment status, standardisation status, and security mechanisms are identified for each P2P system. For the property of network structure, three possibilities are identified: unstructured, structured, and the combination of both. For the centralisation of the index, likewise, three choices are identified: centralised, distributed, and hybrid. Meshkova et al. (2008), as part of a more generic survey encompassing also non-P2P-based resource and service discovery frameworks, classify P2P networks into structured, unstructured, and hybrid categories. Structured networks are noted to be efficient in data searching but not good for partial-match queries. Unstructured networks are noted to support partial-match queries and be quite resistant to churn, but they have scalability problems. Hybrid networks are identified as those which introduce some structure into an unstructured network by node clustering, thereby achieving better scalability through more efficient indexing than fully unstructured networks. It is noted that in clustering-based systems, names used for the semi-centralised nodes in the topology include cluster heads, super-peers, super-nodes, and ultrapeers, while the rest of the peers are called ordinary peers or leaf peers. Alima et al. (2005) divide P2P networks into unstructured and structured categories, and two properties common to all or most structured P2P networks are identified: first,
they are able to provide logarithmic distances between the nodes, and second, they converge towards a stabilised state. The properties mentioned arise from the well-defined rules that determine the operation of structured networks. Sakaryan et al. (2006) identify structure and the type of search as technical properties for the classification of P2P protocols. The structure can be “organised” or “not organised”, and the type of search can be “blind” or “informed”. Blind search is the process of querying other nodes at random until the resource is found, while in informed search the searcher can use some kind of index for facilitating the searching process. Lua et al. (2005) divide P2P networks into the structured and unstructured variants; unstructured networks are further classified to flat and hierarchical ones. Moreover, structured and unstructured systems are classified based on various differentiating factors, such as the lookup protocol used, the obtainable routing performance, and security properties.

Jounga & Lin (2010) categorise P2P networks first into the familiar structured and unstructured variants, but the term “hybrid” is assigned a meaning that is different from the usages by the above-mentioned authors: hybrid networks are identified as those, which combine both structured and unstructured routing in the same P2P system. Schoder et al. (2005) identify hybrid P2P systems as those, where the index is completely centralised. Schollmeier (2001) divides P2P networks into pure and hybrid categories. Pure P2P networks are defined as those, which do not suffer any loss of network service if any arbitrarily chosen node is removed; hybrid P2P networks are defined as those, which are non-pure due to any centralised elements. It definitely seems that one should not use the term “hybrid P2P networks” without making sure that the intended audience understands what is meant, because the term has an exceptionally large number of varying meanings with different authors.

The aforementioned classifications mostly pertain to systems consisting of one overlay network. Some P2P systems, however, are multi-overlay systems, where several overlay networks are interconnected for some purpose such as optimising the routing in the entire system. Depending on the particular multi-overlay system, the individual overlay networks may employ identical or heterogeneous overlay network technologies. Structured P2P systems with internal hierarchy are discussed by Zöls et al. (2008); in addition to systems that could be categorised as single-overlay DHT-based networks with hybrid indexing, also true multi-overlay systems where multiple independent DHT networks are connected with one main DHT network are presented as an example of hierarchical systems.
2.1.2 Challenges and solutions in P2P development

Several challenges are faced by the designers of P2P systems. It is not a trivial task to create P2P networks that are scalable, secure, resource-efficient, fair for the different users, and compatible with the dynamic and non-uniform network environment. In the literature, these issues have been surveyed, and several enhancements to the operation of P2P networks have been proposed.

**Scalability** is a key design issue for any P2P system intended for large-scale deployment. Stiller & Mischke (2005) define scalability as the property that allows a system to accommodate an increasing number of events without degrading the efficiency of the system at all (strict scalability) or degrading it only by a ratio to the efficiency of an ideal reference system (scalability without the “strict” qualifier). Further, P2P scalability is defined in relation to an idealised P2P system, whose efficiency is expressed in terms of the consumption of certain resources that are prerequisites for the functioning of the system. As noted earlier, in structured P2P systems scalability tends to be good, because the number of hops in routing increases logarithmically when the size of the overlay network increases linearly.

Scalability is especially important for the search properties and overlay management properties of a P2P system, as the former is usually the most important functionality for the end-user applications and the latter is a mechanism that allows the overlay network to keep itself functional. In large-scale P2P networks, the proper functioning of search mechanisms requires flexibility to cope with population dynamism, content dynamism, and node heterogeneity, but on the other hand, requires bandwidth efficiency in order to be scalable (Ahmed & Boutaba 2011). Structured systems usually have good scalability properties, thanks to their nearly guaranteed ability to find a searched resource (Lua et al. 2005). While unstructured systems do not scale as well as structured ones, the problem is somewhat mitigated in such networks, where most queries are targeted for well-replicated data resources that can be found with less flooding than rare resources (Chawathe et al. 2003).

**Security** in P2P systems is hard to achieve, because the nodes have only limited knowledge about the entire network and the acts of malicious nodes can easily influence large portions of the system. Moreover, the systems have of course other security issues not stemming from the P2P-specific properties of the systems. Based on a survey by Urdaneta et al. (2011), three prominent types of attacks against DHT systems, and examples of countermeasures for the attacks,
are explained in the following: In a Sybil attack, a large number of malicious nodes are inserted into the DHT, and by controlling them, the attacker is able to fool those DHT procedures, mainly reputation mechanisms, that depend on the presupposition that only a small fraction of the nodes are malicious. The most obvious solution against Sybil attacks is a trusted authority for the generation of the node identifiers, but also other approaches have been proposed, including the utilisation of information on social relationships in the network, computational puzzles for increasing the cost of maintaining an identity, and the utilisation of physical network characteristics for node identification. In an Eclipse attack, the routing tables of good nodes are poisoned with references to malicious nodes. Defences against Eclipse attacks include constraining the usage of vulnerable routing tables that have been optimised with network measurements, and the utilisation of redundant routing table entries based on network proximity information. In routing and storage attacks, malicious nodes act against the routing and storage related rules of the DHT system, for instance, by intentionally routing to malicious nodes or by publishing corrupted data in the system.

Palomar et al. (2006) survey P2P-related security issues, pertaining to DHT-based and non-DHT-based systems. In the study, several existing P2P systems are grouped to the usage categories of overlay routing, user community management, and content distribution, and their security properties are analysed. The authors observe which of those systems have detection or protection mechanisms, or both, against the specified security threats. The security issues include, among many others, denial of service (DoS) attacks, man-in-the-middle attacks, and node-ID stealth. Wallach (2003), emphasising DHT-based networks, identifies familiar security issues and proposes solutions to them. Among other things, the solutions include secure routing primitives to avoid maliciously induced routing errors, self-certifying data to avoid tampering, secure node-ID assignment to avoid Sybil attacks or malicious selections of a non-random node-ID, and distributed auditing to decrease the users’ willingness to cheat in storage space trading. Ejection of badly behaving nodes is identified as an open problem, and the dangers of running untrusted P2P application code on a local computing device are pointed out. According to Chopra et al. (2009), real-time communication oriented P2P overlay networks, which use DHT-based systems for operations such as user registration and location lookup queries, pose special security challenges; the result of breach of security can be, for instance, real-time eavesdropping on a private conversation.
Churn, the constant joining and leaving of peers in the overlay network, is a major factor affecting negatively the operation of P2P networks and must be taken into account in the design of P2P systems (Ohzahata & Kawashima 2011). Resource lookup performance in DHT-based systems suffers greatly from churn (Wu et al. 2006). Churn, in addition to the rather obvious facts that it lowers the routing performance and resource availability in a P2P network, affects also other functionalities such as reputation systems that maintain information about the trustworthiness of remote nodes (Sanchez-Artigas 2010). Techniques for mitigating the effects of churn include the selection of the best lookup strategy and introducing redundancy with parallel lookup requests and key replication (Wu et al. 2006).

Characteristics of churn have been studied by several authors. Yang et al. (2009) present a three-dimensional conceptual model about the causes of churn and the consequences of churn in P2P networks. Based on the model, the authors propose a framework for evaluating the performance of different P2P protocols under churn and apply it to three P2P protocols in a medium-scale overlay setting. Based on the measurements by Stutzbach & Rejaie (2006), churn is not Poisson nor Pareto-distributed; instead, the Weibull distribution is observed to be a more suitable model for the online times of peers in a P2P network under churn. A large portion of the online times are very short, but also very long sessions, measured in days or weeks, are shown by some stable peers. It is pointed out that it is possible to approximate roughly the expected online time of a peer based on its last known behaviour. Ou et al. (2010), in addition to studying the effects of churn on a communication-oriented P2P network, give an overview of existing studies on churn in Kademlia-based DHT systems. Simulation-based studies often focus on how protocol modifications or the different overlay network parameters such as lookup strategies affect the system’s performance under churn; studies involving real-world networks often observe patterns in the behaviour of peers, but can also focus on performance optimisation under churn. Li et al. (2005a) study the performance of DHTs under churn by using simulations, and Li et al. (2005b) present a simulation-validated performance vs. cost framework for guiding in trade-off decisions concerning different protocol parameters during the design of a DHT system, where churn will occur.

Fairness in P2P networks is the property that each peer is allocated a moderate amount of load, when it participates in the provision of the services in the network. Unfairness in a P2P network can be a result of the technical properties of the system, or of the selfish actions of the users who are free-riders.
in the system. Load balancing is the activity of dividing the load between peers evenly, possibly taking the capabilities of each peer into account. Various techniques for load balancing have been proposed, such as virtual servers that are distributed on physical nodes in a way that balances the load (Rao et al. 2003), histogram-based division of the network to regions whose load is controlled (Vu et al. 2009), and random-choice schemes that may take churn and node heterogeneity into account in their action (Fu et al. 2011). To counteract the selfishness of individual peers, incentive-based mechanisms can be used for ensuring that each peer contributes resources to the network. Incentive systems are often based on game-theoretic approaches that make it more profitable for an individual to comply with the rules than to act in a selfish manner (Ma et al. 2006, Park & van der Schaar 2010). Under certain conditions, P2P systems may also be robust by some metrics even when peers act selfishly (Sasabe et al. 2010); peers often also exhibit some tendency to act altruistically even without effective incentive mechanisms (Vassilakis & Vassalos 2009). The popular BitTorrent P2P system has an example of an incentive mechanism working in the real world; however, according to Li et al. (2008), improvements in the incentive system are needed in cases where the network contains a large number of seeds, in other words, peers that already have all pieces of the shared data resource.

Performance of P2P network operations, most notably the resource search operations, has a high priority in P2P network research. There are various proposals for improving search or overlay-management performance in different kinds of P2P systems. Ahmed & Boutaba (2007) describe a P2P system based on distributed pattern matching (DPM). Each shared resource in the network is assigned a bit pattern, which summarises the relevant properties of the resource, and the patterns are distributed and replicated on the peers in an organised way. Although the proposed network is not DHT-based, it is claimed to have routing performance similar to the performance of DHT systems. Rao et al. (2010) propose optimised placement of resources on specific peers for reducing the average lookup hop count in structured P2P networks, thus improving search performance. The proposed technique can be used for the placement of both data replicas and inter-node links. Effective solutions for P2P content caching and P2P ring overlay management are presented as examples of systems, where the technique can be applied.

It should be noted that DHT-based systems, which are designed for the storing and searching of exactly matching resource keys, do not inherently support range queries, although it is not impossible to combine wildcard-query
support and DHT-based networks (Bharambe et al. 2004). A survey of P2P systems capable of range queries for data resources is given by Xu et al. (2011); many of these systems are based on skip lists, which are essentially linked lists of ordered items with the capability of skipping over some items during searching. Keyword searches can be implemented in DHT-based systems by introducing mechanisms that map the keywords or keyword sets to certain nodes in the network and further to the searched resources (Joung et al. 2007).

While the topology of a P2P network is usually built without consideration for the characteristics of the underlying physical network, it is also possible to take them into account in order to increase the performance and scalability of the system. Hoang et al. (2005) propose a physical infrastructure aware P2P system. The system aims for good routing performance by mirroring the underlying network and using at the same time a key-tree based algorithm on top of the optimised overlay network.

In contrast to many client-server systems, where it is feasible for the large centralised servers to have public IP addresses, peers in a P2P system have a far greater probability of being behind a NAT. Mobile phones are an example of terminals that are often situated behind a NAT (Kelényi et al. 2010). Techniques designed in the Internet Engineering Task Force (IETF) for NAT traversal include Session Traversal Utilities for NAT (STUN), Traversal Using Relays around NAT (TURN), and Interactive Connectivity Establishment (ICE).

Many proposals combine P2P networks with elements from other areas of technological research. Neri et al. (2008) apply neural networks and memetic algorithms for efficient discovery of shared resources in P2P networks. Ganesh et al. (2003) apply a gossip-based networking protocol to membership management in unstructured P2P systems. The physical-topology aware and data-semantics aware P2P system described by Xu & Jin (2009) utilises a caching scheme that is designed with the small world phenomenon in mind. The small world phenomenon is a name for the widely studied network-theoretical observation that the routing hop count between any nodes in a graph tends to be low, when nodes have many links to their immediate neighbours and some links to chosen far-away nodes.

Special challenges related to the mobile usage of P2P networks are discussed in a separate section.
2.1.3 Existing P2P systems

In the following, certain individual examples of P2P systems presented in the literature are overviewed. For each presented P2P system, the most prominent characteristics are identified.

First, to provide a brief historical overview of decentralised communication systems predating the actual P2P networks, the influential Advanced Research Projects Agency Network (ARPANET) and Usenet systems can be mentioned as examples of systems that exhibit P2P-like characteristics (Zhang et al. 2006). The ARPANET was built around the principle of connecting computers in a packet-switched distributed system, where each computer can send messages to each other computer and thus use their shared resources. Furthermore, it should be noted that the ARPANET was essentially the first version of the modern Internet, and still today the basic infrastructure of the Internet that at least in theory aims for unhindered connectability and equality between any nodes is reminiscent of a P2P network. Usenet is a distributed system for exchanging news articles for the purpose of discussion between the users. Usenet works without a centralised main computer: the servers of the network communicate in a distributed fashion, copying the incoming news items to other servers that they have “peered” with. The common usage of that word in the context of interconnected Usenet servers highlights the fact that Usenet has P2P-like characteristics.

Napster, launched in 1999, is considered by many to be the first P2P system, or at least the first well-known system labelled as P2P. Napster used centralised indexing (Lua et al. 2005) and an unstructured network architecture. The system was designed for the purpose of sharing files between the participating end-users’ computers. The current music service with the name “Napster” is not the same system as the original Napster, as the latter was forced to shut down due to protests from content copyright holders.

Gnutella v0.4 (Ripeanu et al. 2002) is the canonical example of unstructured P2P systems with distributed indexing. Routing in Gnutella v0.4 is based on flooding of messages to the known neighbour nodes in the overlay network. A time-to-live (TTL) field in the messages is used for preventing situations where a message circulates forever in the system. Obviously, the flooding-based approach has poor scalability and it can lead to a heavy load on nodes that happen to have an unlucky position in the topology, but on the other hand, the system is rather robust against node failures, because the effects of churn are mostly confined to a small part of the network. An example Gnutella v0.4 network is shown in Fig. 2.
JXTA (Gong 2001) and FastTrack (Liang et al. 2006) are examples of hybrid-indexing based P2P systems, where the nodes with most capabilities are nominated as super-peers for serving the less capable, ordinary peers. Both of these P2P systems are unstructured; it should be noted, however, that also DHT-based newer versions of JXTA exist. JXTA is an early attempt of standardising P2P networking. The JXTA documentation defines several Extensible Markup Language (XML) based protocols that are responsible for different P2P functionalities, such as peer discovery and peer-group membership management. Super-peers in JXTA are divided into two categories: rendezvous peers, which act as hubs for connecting several peers, and relay peers, which exist for firewall traversal. The file-sharing system Kazaa that has attracted a large number of users is based on FastTrack. Each FastTrack peer exchanges with the other peers encrypted signalling messages and unencrypted content transfer traffic, commercial advertisements, and instant messaging (IM) traffic. Several super-peers can be contacted by an ordinary peer during a search. A super-peer may forward a received query to another super-peer that it knows.

The Freenet P2P system (Clarke et al. 2001) has a distributed indexing scheme and uses a routing model that differs from the model used in Napster, Gnutella, and FastTrack: the data resources are not stored on the peers that provide them, but instead on peers located elsewhere in the network, in order to provide an anonymity-supporting content sharing service. Hashes of textual descriptions of shared content are used for identifying the shared data resources in
In order to prevent data corruption attacks, Freenet also introduced content hashes that ensure the identity of a shared file based on its contents.

The popular Direct Connect (DC) system (Gurvich et al. 2010) can be seen as an unstructured P2P network with an indexing scheme poised between hybrid and centralised, as the users manually select the super-peer (hub) that they wish to connect to. Hubs distribute content queries to their connected peers and can exist on large servers or on the users’ personal computing devices. Peers that have the relevant content may answer to the searcher directly or by using the hub as a proxy. Excessive duplication of queries is noted to be a problem, because a searching peer often sends the same query to several hubs and the hubs cannot distinguish which of the incoming queries are duplicates.

The open IM standard Extensible Messaging and Presence Protocol (XMPP) (Saint-Andre 2005), also known as Jabber, is classified as a P2P protocol by many, although only some of the implementations are P2P-based. Nevertheless, XMPP has P2P-like properties. There is no centralised control and thus XMPP servers connect with each other in a distributed fashion; moreover, the participating network nodes in an IM conversation are naturally in a peer-like relationship with each other.

Any DHT-based P2P system needs to utilise a specific DHT algorithm; several different DHT algorithms exist. Chord (Stoica et al. 2003) is a DHT algorithm with a circular ID space of \(2^m\) possible IDs. At the time when Chord was first presented, one its main contributions was good scalability: before Chord, systems based on consistent hashing assumed that each node would know most other nodes. With Chord, only logarithmic state information about other peers is required, the number of hops in lookup operations is logarithmic, and performance will degrade gracefully when churn takes some peers offline. In addition to information regarding immediate successor nodes in the logical ID space, each Chord peer maintains a finger table, which contains up to \(m\) links to more remote nodes. These links are selected in such a way that the \(i\)th finger is the nearest peer with at least the distance \(2^{i-1}\) (modulo \(2^m\)) from the local peer ID. Obviously, the system works with any ID length \(m\), which could be, for example, 128 or 160 bits in a practical system; the operation of Chord is illustrated with a simple 5-bit example system in Fig. 3 and Fig. 4.
Kademlia (Maymounkov & Mazieres 2002) is another DHT algorithm that uses a ring-shaped logical ID space of bit-string based keys. Routing in Kademlia uses
the bitwise exclusive-or (XOR) operation for determining the distance between two IDs. Again, the algorithm would work with any chosen ID length; the authors however specify an ID length of 160 bits. Each node stores routing information in lists called $k$-buckets. There are 160 such lists per node, one $k$-bucket per one bit of the ID namespace. These routing tables are arranged in such a way that the $i$th $k$-bucket stores links to up to $k$ most recently seen remote nodes, whose IDs are at a distance between $2^i$ and $2^{i+1}$ from the local node ID. Routing, of course, proceeds towards the nodes that have longer matching portions in their IDs until the searched node or resource is found. Unlike Chord, the Kademlia routing system is designed to allow the peers to learn useful routing information as a side effect of the exchange of any messages, and the routing system is more flexible than that of Chord.

Other DHT algorithms include Pastry (Rowstron & Druschel 2001), Tapestry (Zhao et al. 2004), and the multi-dimensional ID-space based Content-Addressable Network (CAN) (Ratnasamy et al. 2001).

BitTorrent, analysed by Xia & Muppala (2010), is one of the most prominent P2P systems in use today, and perhaps the most successful one for the sharing of large files such as Linux distribution packages and movies. BitTorrent uses a centralised indexing scheme, where a tracker server keeps a list of the peers participating in the sharing of a specific data resource, and the Uniform Resource Locator (URL) of the tracker is obtained from a Web server that hosts meta-information about the data resource. Data resources are split into small chunks; different subsets of the chunks can be located on different peers at a specific time. Splitting the data resources into pieces enables the efficient usage of the peers’ upload and download bandwidth, the reciprocity-based incentive mechanism “tit for tat”, and the “rarest piece first” downloading approach that ensures the quick distribution of all pieces and improves an individual peer’s chances for the successful trading of pieces. In trackerless BitTorrent systems, a DHT-based overlay network, commonly using the Kademlia algorithm, is used instead of a centralised tracker server for providing information about participating peers.

The important P2P standardisation effort P2PSIP is discussed in the separate section on session management. Standardisation activities for different areas of P2P networking in addition to P2PSIP have been initiated in the IETF. P2P-based media streaming systems are studied in the Peer-to-Peer Streaming Protocol (PPSP) working group. The Decoupled Application Data Enroute (DECADE) working group aims to alleviate the load caused by P2P traffic by introducing an in-network data storage system. The Application-Layer Traffic Optimization
(ALTO) working group addresses the problem of how to select a good peer out of the possible candidate nodes, based on the network topology.

A few examples of P2P-related research projects from around the world can be mentioned. One of the most well-known P2P-related experiments on a large scale is PlanetLab⁵, a distributed systems testing and deployment facility run by an international consortium. As of 2011, PlanetLab consists of more than one thousand computational hosts located on several continents. It has been used as a substrate for several P2P systems such as OpenDHT, Coral, and OpenVoIP. OpenDHT is an openly accessible DHT-based storage system without the need to run a DHT node on all hosts utilising the service (Rhea et al. 2005). Coral is a Web-cache-enhancing content distribution system, which is based on so-called distributed sloppy hash tables (DSHT) (Freedman et al. 2004). OpenVoIP, developed in Columbia University, is an open voice-over-IP (VoIP) and IM system supporting several DHT algorithms and hash algorithms and is based on the P2PP protocol (Baset et al. 2008).

A European Union (EU) funded P2P networking research project, P2P-Next⁶, aims to replace broadcast-based television networks with streaming P2P content distribution networks. From the end-user point of view, the prototype video-streaming system created in the P2P-Next project is accessed using Tribler, which is an open-source P2P video viewer application. The content sharing network is based on BitTorrent.

The Future Internet (FI) project is being conducted in the Strategic Centres for Science, Technology and Innovation (SHOK)⁷ research framework. The programme aims to solve problems such as unwanted Internet traffic, choking of the routing system, network congestion, and challenges posed by terminal mobility. Improvements are expected in areas such as the quality of end-to-end Internet connectivity and techniques for information storage and delivery. The programme contains work items related to P2P networking and the Internet of Things (IoT), which also can utilise P2P protocols in its operation.

Integration with Web-based information systems is a logical step in the evolution of P2P systems. In Web applications, it is possible to combine P2P-based functionalities with a Web-based user interface. In the ExpeShare research project funded by the Information Technology for European Advancement 2

---

⁵ http://www.planet-lab.org/ accessed on 18 May 2011
(ITEA 2) programme, a community-centric mobile service environment prototype with Web-P2P integration has been created (Koskela et al. 2010). In the system, community information is managed by a P2P network. The communities can be location-based: for example, they can be formed of the clientele of an entertainment venue with the help of short-range communication (SRC) technologies. The mobile application seen by the end-users is Web-based, and the Web browser accesses the services of the P2P networking software component over a dedicated local communication channel in the mobile terminal.

2.1.4 Efficiency of mobile usage

Mobile P2P networking poses challenges that often do not apply to P2P networks consisting of only stationary computers. The literature contains many studies on the requirements of mobile P2P networking and on non-P2P-specific aspects of mobile networking, such as observations on the resource usage of mobile terminals. Several P2P system optimisations related to mobile devices have been proposed. Some of them are discussed in this section; however, some parts of the mobile P2P related literature review can be found in a later section on mobile middleware.

Technical solutions for optimising the resource usage of mobile devices have been proposed. Resources considered in the studies include battery capacity, network traffic capacity, processing capacity, and data storage space. Energy consumption, since mobile devices have limited battery capacity, is a very important area in the research of mobile computing systems. Energy consumption in electronic systems depends on both hardware and software components, and energy savings are possible by modifying the design of either or both (Benini & de Micheli 2000). All operations of a mobile device consume energy, but the wireless network interface affects the power consumption significantly (Pering et al. 2006). Moreover, hardware-level improvements in power efficiency have been practically traded for the increased functionality of the devices (Pentikousis 2010). Hence, energy consumption studies and proposed energy optimisations are emphasised in this part of the literature review. Connectivity management is a name for the management of the networking resources of the mobile device.

Multi-access networking is a promising technique for attaining energy savings in wireless communication (Pentikousis 2010). For example, Lampropoulos et al. (2007) analyse the energy implications of concurrent integrated usage of UMTS and WLAN connectivities. Pering et al. (2006) present
experimentally validated policies for dynamic switching of heterogeneous wireless network interfaces in different application scenarios. Rautio et al. (2011) present an information-centric P2P system with support for multi-network access possibility for mobile hosts. The connectivity management functionality of the system allows the mobile device to decide, which of the available networks should be chosen for use at a given moment; the choice can also be a combination of more than one heterogeneous access networks. The evaluations show that the prototype can attain high performance gains in both mobile and non-mobile scenarios, and is able to reduce the amount of traffic between different networks.

Prediction of underlying network conditions can be utilised to make the data transmissions of an application more energy-efficient (Sri Kalyanaraman et al. 2010). Cross-layer collaboration within the device, aided by end-to-end network information, can be used for minimising the energy consumption of a mobile device (Mohapatra et al. 2007).

Power consumption incurred by software execution can be modelled and optimised with logic on the source code level (Fei et al. 2007), software component level (Jun et al. 2006), or software behavioural level (Kim et al. 2010). An interesting approach to power generation and management is that in very low-power wireless systems such as sensors, power can be extracted from their environment; this is called energy harvesting (Chen et al. 2010).

Gurun et al. (2006) observe the energy consumption of different operations in structured P2P overlay networks. Based on the observations, an energy-saving scheme is proposed, where the network interface is put into a low-power state when the P2P protocol has been in an idle state for a specific time period. Detailed measurements of the energy consumption of mobile devices in BitTorrent-based file sharing are presented by Nurminen & Nöyränen (2008). The observations include that the power consumption during BitTorrent usage is comparable to the consumption during voice calls, and that the mobile device can well act as a full peer (upload data) when it downloads data because the uploading in that case causes only a small increase in the total power consumption. Ou et al. (2010) present detailed measurements of the energy consumption incurred by non-application-specific User Datagram Protocol (UDP) based messaging in UMTS and WLAN access networks, along with measurements of central processing unit (CPU) load and average network messaging load during mobile P2P networking.

The end-users’ terminals are not the only elements requiring energy usage optimisations in wireless networks. When the overall energy efficiency of an
access network needs to be improved, the power efficiency of cellular base stations is an important factor to consider (Jada et al. 2010). Kelényi et al. (2010a) study the energy efficiency of a BitTorrent system, where proxy servers are used as helper nodes for the participating mobile nodes, from the viewpoints of the mobile device itself and of the server infrastructure. It is observed that home routers may act as a distributed proxy system in an energy-efficient and bandwidth-efficient way.

The study by Kelényi et al. (2010a) presents measurements of energy consumption per bit in a mobile device as a function of communication bit rate. A higher bit rate leads to better energy efficiency. However, the relationship between the energy consumption and the amount of transmitted or received data is usually non-trivial. Cellular network connections cause the radio to stay prolonged periods in high-energy states after the actual data traffic has occurred, and thus the timing between different bursts of data affects the energy consumption; this knowledge can be utilised in the design of power-efficient networking protocols for mobile terminals (Balasubramanian et al. 2009). Pentikousis (2009) notes that there are few realistic models for the energy consumption of transport protocols in mobile devices, and that the non-linear nature of battery depletion and recovery should be taken into account in a model.

Mobile P2PSIP has been evaluated on the proof-of-concept level in different application scenarios (Matuszewski & Kokkonen 2008, Koskela et al. 2010). Using unstructured P2P networks instead of DHT-based networks for mobile P2PSIP has been proposed (Cheng et al. 2007). The motivations include the facts that unstructured networks are more churn-tolerant and that the technique allows the network topology to be based on the social relationships between the users.

It is fully possible to create DHT networks that exclusively consist of mobile devices; Kelényi & Forstner (2008) claim to be the first to have implemented such a network. The energy efficiency of DHT-based P2PSIP in mobile usage is studied by Kelényi et al. (2010b). The presented analytical model of Kademlia-based DHT systems suggests that in a million-node overlay network approximately 30% of the mobile nodes must participate as peers (not clients) in order to keep the average peer energy-efficient in the DHT.

Not all mobile P2P optimisations are related to power consumption. For example, Ekler et al. (2011) study the possibilities of using network coding for alleviating the problem of rare data pieces, which occurs due to the unreliable nature of the mobile peers and their network connections. Karonen & Nurminen (2008) propose a modification to the incentive mechanisms of P2P networks in
order to make them more suitable for mobile nodes. Instead of monitoring only the participation level of each single terminal, all devices of a user are taken into account. The proposed improvement makes a P2P system fairer from the viewpoint of mobile devices, while also making sure that an adequate number of peers are present in the network.

The special characteristics of the mobile usage environment can be used for enabling different usage scenarios for P2P systems. Kotilainen et al. (2008) present a location-based system for media sharing between mobile P2P users. Borcea & Iamnitchi (2008) present a software infrastructure for social interaction in the context of mobile P2P applications. The system collects and exploits social knowledge in order to enable socially-aware services, including functionalities such as inferring the involved communities and social contexts. The system allows for user-deployed services and removes the need for centralised parties that manage the social information. Gao et al. (2005) describe P2P-based monetary payment in scenarios where the payer and payee have mobile terminals.

2.1.5 Evaluations and evaluation methods of P2P systems

Several studies about the performance of real-world or simulated P2P networks have been published. Moreover, the behaviour of peers or the users of the peer hosts has been studied in order to learn what kind of action patterns the network should support. In the evaluations, different experiment setups and methodologies are used by different authors.

Some authors differentiate network simulations from network emulations, stating that emulations are conducted in real-time using realistic protocol implementations, while simulations are more simplified evaluations run in compressed time (Kotilainen et al. 2006). The author of this thesis, however, uses the term simulation to encompass also real-time network system emulations as long as they are conducted in a testbed system and not in a real-world overlay network. This usage of the term can be seen in original Papers VII-IX.

Evaluation results may not always give a realistic picture of the evaluated P2P system. Choffnes & Bustamante (2010), using specifically a large-scale P2P network as the object of a case study, identify pitfalls in the usage of testbeds for evaluating Internet systems: testbed-originated results should not be naively generalised to an Internet-wide perspective. Similarly, Haeberlen et al. (2006) note that simulations are often based on assumptions that do not hold in the actual P2P networks with regard to factors such as mobility and lower-layer routing.
consistency, and that certain simulator systems and certain evaluation points in the space of the possible workloads are becoming “standards” that are used without due consideration. Proprietary closed-source P2P systems may need reverse-engineering or indirect observations such as packet sniffing in order to understand the operation logic and message syntax of such systems (Liang et al. 2006). Naicken et al. (2007) criticise the majority of P2P simulation studies about a reporting style that makes the reproduction of the experiment results difficult.

Analyses of real-world P2P traffic are often based on crawler nodes that join the network and maintain connections with a number of known nodes in order to gather information about the working of the network. However, it is also possible to use non-intrusive methods for observing a live network passively by analysing the packet flows seen by the network routers. This can be done, for example, at the border routers of the network controlled by one Internet service provider (ISP) (Sen & Wang 2004).

A black-box approach for analysing P2P systems is presented by Rossi et al. (2011) in order to make the analysis framework widely applicable and to eliminate the need for protocol reverse-engineering. The framework defines a minimum set of relevant observable properties suitable for the analysis of different P2P systems. These properties include, among others, network layer features such as uplink and downlink packet inter-arrival time (IAT), and cross-layer features such as the overlay network’s awareness on IP address proximity, Autonomous System (AS) locality, and round-trip time (RTT) in its operation. The summary of the analysis data about a specific P2P system can be viewed in a compact visual format.

In a study by Li & Chen (2010), two popular P2P-based video on demand (VoD) systems are reverse-engineered using packet-sniffing software, and crawler-based measurements are conducted for one of the two protocols. Each peer’s buffer map, which is used for advertising its locally stored video chunks, is an essential value observed in the measurements. The watching behaviour and network sharing behaviour of the peers are analysed. The observations on peer behaviour include that co-operation between peers is defined by the watching history of online peers and that, unlike file sharing systems, the majority of peers provide data to the other peers without performing downloads for themselves.

In a study by Vu et al. (2010), the closed-source PPLive Internet Protocol television (IPTV) streaming system is analysed using a crawler, which can obtain a snapshot of the peers attending a specific channel and find out the peers communicating with the local peer. The observations include models for session
length and the participation of peers in multiple overlays, and the fact that the characteristics of the streaming overlays are clearly different from those of file-sharing overlays.

Guo et al. (2007) create measurement-based performance models for BitTorrent-like systems. First, systems with a single overlay network (“torrent”) for sharing the pieces of one specific resource are analysed based on two types of traces: downloads of data files recorded by trackers, and downloads of meta-files recorded at server farms and in the cable network. Among other things it is observed that the exponentially decreasing peer joining rate leads quickly to poor data resource availability. Second, a model is presented for the analysis of inter-torrent correlation in multi-torrent systems; metrics such as torrent birth rate and torrent request rate are observed. The performance of a proposed multi-torrent collaboration system is evaluated with simulations that are based on the recorded tracker traces.

In order to make realistic assumptions about the behaviour of peers in calculations, knowledge about the details of activity in real-life P2P networks is needed. Ohzahata & Kawashima (2011) analyse peer behaviour in an unstructured file-sharing P2P system called Winny using a crawler-based approach. For finding file-uploading peers, a dot “.” is used as the search keyword, because it appears in the majority of file names. It is observed that the amount of selfish peers in the system is dependent on the time of the day, the distribution of peer lifetime is not long-tailed, and the joining process of peers is not a Poisson process.

Stutzbach & Rejaie (2006) describe the methodology for a study on the characteristics of churn in three P2P systems with real-world deployments. The data sets used as the input for the calculations are centralised log files about the joining and leaving events of the peers, and periodical snapshots of the state of the overlay network. Several potential pitfalls for churn-related studies are identified, including incorrectly observed long online times, biased peer selection for measurements, and false negatives when determining the online peers at a given moment.

Various testbed systems for the simulation of P2P systems have been used. Kotilainen et al. (2006) give several examples of simulators suitable for P2P evaluations. These include both P2P-specific and generic network simulators. The different simulators are compared based on properties such as scalability, the attainable level of detail in the simulations, and the ability to simulate the underlying router infrastructure that affects the operation of the overlay network. Scalability and the lack of standardisation in P2P simulators are identified as
challenges. Naicken et al. (2007) give another survey on P2P simulators, differentiating the simulators with properties such as scalability, usability, licensing, and the simulation of the underlying network. Identified challenges with the surveyed simulators include poor documentation, unsatisfying scalability, and the inability to gather meaningful statistics of a simulation run.

Ou et al. (2010) simulate DHT-based P2P networks in order to learn feasible levels for query parallelism and data resource replication, and to evaluate various types of load inflicted on the peers as a result of participation in the P2P overlay. The approach involves two implementations of the P2PP protocol: one created using the scripting facilities of a telecommunication simulation software tool and one created for actual networked devices.

### 2.1.6 Session management

Session management is the activity of creating, modifying, and tearing down communication sessions between two or more network nodes. Usually, the media streams exchanged during a session are not considered to be part of session management; instead, session management refers to the signalling that takes place for manipulating the state of the session.

A prominent example of session management technology, SIP\(^8\), is a client-server protocol: SIP registrar servers are used for storing the IP addresses of the users’ SIP user agents (SIP UA). The SIP UAs usually interface with the system by communicating with the SIP proxy servers, which assist in the processing of messages such as session invitation requests sent by the SIP UAs. When a user wants to invite another user to a session, the caller’s SIP UA resolves the callee’s known SIP Uniform Resource Identifier (URI) to the callee’s IP address by consulting a server. Numerous additional specifications beyond the basic protocol documentation of SIP exist, for example, for the purposes of IM, event notifications, and integration with the cellular and landline telephony systems of the public switched telephone network (PSTN). In the PSTN, session (call) initiation and teardown, as well as some other functionalities, are usually realised using the Signaling System No. 7 (SS7) protocol suite.

In SIP, the details of the proposed media session are carried in the invitation request for that session. However, thanks to the strict separation of concerns in the design of the protocols, SIP itself does not specify how to express intra-session

---

\(^8\) IETF Request for Comments (RFC) 3261
details such as the properties of the exchanged audio or video streams; instead, this information is carried in an attached Session Description Protocol (SDP) payload. The actual media streams of the session are transferred usually over Real-Time Transport Protocol (RTP), which, in turn, is used together with Real-Time Transport Control Protocol (RTCP) whose purpose is to control the quality of service (QoS) of the media session. The SIP signalling messages are sent over either Transmission Control Protocol (TCP) or UDP.

In a sense, SIP-based communication systems have P2P-like characteristics, because the endpoints of the communication are in an equal position, being “peers” with each other. The service infrastructure, however, is based on centralised server machines. With the SIP infrastructure, under certain assumptions, it is possible to attain a level of call-processing reliability comparable to the PSTN system (Gurbani et al. 2005). The IP Multimedia Subsystem (IMS), a framework of network functionality specifications with the goal of bringing IP-based services under the control of telecommunication operators, uses SIP-based signalling.

The P2PSIP protocol was originally conceived as a distributed alternative to client-server SIP. In session management usage, P2PSIP is used for storing the contact information of the users who are willing to be available for media sessions (Bryan & LoweKamp 2007). P2PSIP can be used without SIP functionality; in that case, the standards-based P2P overlay network acts as a generic distributed store-and-fetch repository for data resources, and various services can be implemented on top of this system. The interconnection of P2PSIP and IMS based systems has been studied by Hautakorpi et al. (2008).

2.2 Mobile middleware systems

In the following subsections, the existing research on middleware systems is reviewed. This part of the literature review includes definitions and design goals of middleware, concrete examples of middleware systems, and studies on cross-platform software development. Mobile middleware systems are emphasised in the review, but also non-mobile systems are analysed in order to give a more comprehensive picture of middleware.

2.2.1 Definitions and design goals of middleware

Middleware is used in order to provide reusable, networking-related functionality for applications. This can include hiding the complexity of lower-level network
operations, introducing new functionalities for the applications to access, and
integrating the application with the business processes of an organisation.
Applications access the middleware functionalities through a well-defined
application programming interface (API). Several definitions for middleware and
motivations for creating it have been given in the literature.

According to Yoneki & Bacon (2007), middleware is a name for the versatile
set of distributed computing related services, APIs, and management systems.
These authors’ view of the scope of middleware is rather broad: from the
application-layer perspective, everything below the API is middleware, and from
the lower-layer networking perspective, everything above TCP/IP can be seen as
middleware. Thus, in practice, any networking protocol implementation in the
application layer is middleware. According to Bruneo et al. (2007), middleware is
responsible for implementing the session and presentation layers of the Open
Systems Interconnection (OSI) network layer model.

The IETF has produced a memo on middleware. In the memo, it is noted that
in advanced networks middleware usually consists of services between the
applications and the underlying packet-switched infrastructure, while no real
consensus exists on the precise definition of the boundaries of the concept.
Examples of middleware components mentioned in the memo include, among
other things, APIs, policy frameworks, resource management mechanisms,
networked information discovery and retrieval services, and issues related to
authentication, authorisation, and accounting (AAA). Moreover, the memo
mentions the need for adopting a more organised framework for middleware
research and development.

In the report on mobile middleware by Tarkoma (2009), the basic definition
of middleware is similar to the previous definitions: middleware lies between the
applications and the underlying networking stack such as the TCP/IP or Wireless
Application Protocol (WAP) stack. It is noted that the creation of mobile
middleware poses special challenges, including the limited computational
capabilities of a mobile device and the unreliable wireless network environment.
Typical middleware services are noted to include resource discovery, storage,
transactions, security, directory access, and facilities for messaging and remote
procedure calls (RPC). It is mentioned that middleware typically provides certain
transparencies to the higher layers, which means that the higher layers are
unaware of the transparently handled complexities. Transparencies provided by

9 IETF RFC 2768
middleware systems include, among other things, network location, transport protocol, and failure transparency.

Raatikainen et al. (2002) identify important research areas for the creation of mobile middleware: the functional requirements from the viewpoint of adaptive and efficient applications include dynamically reconfigurable end-user terminals, reconfigurable applications, environment monitoring, and a distributed information base that is accessible anywhere by a mobile terminal. Moreover, it is noted that software architectures suitable for the creation of flexible middleware systems contain an “execution support layer” that facilitates the division of the application logic into co-operating components and the distribution and configuration of the components. It is also noted that due to the multitude of existing or forthcoming middleware solutions, compatibility between different middleware systems needs to be considered by the developers.

In Fig. 5, a block diagram describing the essential elements of a mobile middleware stack (Yoneki & Bacon 2007) is presented. The diagram illustrates how various types of complexity are hidden behind APIs in the middleware system in order to alleviate the workload of application developers. Two main parts of mobile middleware are distinguished in the model: the generic service elements and the communication facilities supporting both synchronous and asynchronous messaging patterns.
In Fig. 6, an illustration of the standing of middleware in an I-centric reference model of mobile services (Popescu-Zeletin et al. 2007) is shown. The reference model consists of four basic horizontal layers, multiple vertical service elements, and desirable service features. Mobile middleware is seen as an essential building block for weaving the different elements together.
2.2.2 Existing middleware systems

In the following, various examples of existing or proposed middleware systems are given, emphasising mobile middleware systems. At the same time, different problem areas where middleware can be applied are presented.

Two examples of traditional remote procedure invocation middleware systems, which also have mobile implementations, are presented in the following. The Common Object Request Broker Architecture (CORBA) is a prominent example of middleware techniques for the intercommunication of software components over a network. The CORBA system is object-oriented and it uses Interface Definition Language (IDL) for creating the interfaces that present the objects. The system has mappings from IDL presentations to various programming languages on various platforms, and provides features such as strong data typing and accessing the objects over the network by reference or by value. The Distributed Component Object Model (DCOM), a proprietary middleware technique from Microsoft, is a RPC system for the intercommunication of software components with features such as distributed
garbage collection mechanisms. Both CORBA and DCOM have faded in popularity due to the advent of RPC middleware systems that utilise Hypertext Transfer Protocol (HTTP) as the transport, which include Extensible Markup Language Remote Procedure Call (XML-RPC) and its successor Simple Object Access Protocol (SOAP). Serialisation (marshalling) of complex data structures into one-dimensional byte strings for network transmission is a key functionality in several types of middleware. Serialisation techniques used in message-passing middleware include various binary formats and text-based formats such as XML.

Mishra et al. (2011) describe a middleware system for load balancing in distributed computing. The load balancing system is verified using process algebra and with a deployment of the middleware system: improvements regarding network traffic load, memory load, and computational load are observed. Duran-Limon et al. (2011) present a middleware with a lightweight virtual machine framework for soft real-time applications. The purpose of the framework is to adapt the execution resources of the applications to sudden changes in the network environment.

Session management middleware is probably best exemplified by any implementation of the SIP protocol or the session management oriented implementations of P2PSIP; these protocols are discussed in other sections. Universal Plug and Play (UPnP) systems, usually mentioned in the context of service discovery, can be seen as middleware for managing multimedia sessions between pieces of digital equipment such as entertainment devices. The purpose of UPnP is to create functional configurations using interfaces for mechanisms such as service discovery and service control. Other middleware systems related to session management include solutions that, being aware of activities in the lower layers, preserve ongoing multimedia sessions when the associated data streams are moved between heterogeneous network interfaces. An example of such middleware is the system presented by Rodriguez et al. (2009); it can be said, however, that maybe such systems are more related to connectivity management than to session management.

Middleware systems for connectivity management in mobile devices include the ones presented by Bellavista et al. (2008), Sun et al. (2005), and the already mentioned example by Rautio et al. (2011). Allowing the mobile devices to be always best connected (ABC) (Gustafsson & Jonsson 2003) is a goal for connectivity management middleware systems. In the ABC paradigm, the wireless network to use is selected for the optimal user experience at all times. The network to use is selected out of the various wireless networks, for which the
mobile terminal has a network interface, taking into account the network properties such as coverage and bandwidth that change constantly as a function of time and place. For optimal ABC support, connectivity management can be facilitated also by other network elements than the mobile device itself: end-to-end ABC supporting techniques can be applied on the transport protocol level (Eddy 2004) or on the network level (Fodor et al. 2003).

Various middleware solutions have been proposed for facilitating the execution of applications that are dependent on the scarce and non-uniform computing resources of mobile devices. Siu et al. (2004) present a middleware-based system called Sparkle for computing on heterogeneous devices in pervasive, mobile environments. An application requests, in run-time, code components called facets when it needs to access a specific functionality implemented by the facet. Facets are stored on facet servers and contain descriptions of their behaviour and requirements. Islam et al. (2004) describe an operating environment for mobile Web-based applications, aiming at the applications’ adaptability to the varying run-time conditions as well as at application portability. The presented middleware system supports on-device caching and execution of code units primarily intended to be run on the server-side, on-device dynamic creation of user interfaces based on their abstract descriptions, and on-device adaptive fault tolerance. Yau & Karim (2004) present a mobile situation-aware middleware for integrating mobile devices with network infrastructures. The motivation is to bring resources of the infrastructure to the reach of the mobile devices. The dynamic integration of the mobile device and the infrastructures is done in specific detected situations and in a way that is transparent to the applications. Román et al. (2002) present a middleware for pervasive computing, Gaia. The middleware acts as a meta-operating system for managing the resources needed for the execution of pervasive, context-sensitive applications on heterogeneous networked devices.

Hall (2010) discusses middleware APIs for exposing and accessing the functionality in operator-controlled telecommunication networks. These APIs include OneAPI and its precursor Parlay X, which provide the application developers access to functionalities such as telephony, messaging, and charging. Telecommunications operators are promoting an alliance called Wholesale Applications Community (WAC) in order to increase the usage of operator-controlled service APIs.

Middleware-like systems for content push towards the user’s terminal, in contrast to the traditional model where the user requests content and is just aided
in the retrieval by a middleware system, are discussed in the literature. Examples of these systems include mobile content push with pre-fetching at times that minimise the resource usage (Bhatia et al. 2009), mobile content push based on trusted computing systems (Kuntze & Schmidt 2007), and Really Simple Syndication (RSS) readers that are a popular push technology for Web content (Paulson 2004).

Some middleware systems are specifically created for P2P networking. In addition to the various P2P implementations that are middleware in the sense that they operate between the network and the application, certain P2P networking software systems are explicitly called middleware by their designers. Examples of mobile P2P middleware include Chedar (Kotilainen et al. 2005) for resource exchange between the terminals, and Plug-and-Play Application Platform (PnPAP) for the dynamic switching of networking technology entities such as P2P protocols (Harjula et al. 2004). Xhafa et al. (2011) survey existing middleware systems for distributed computing services on top of P2P and grid computing infrastructures. The need for more easy-to-use middleware platforms for parallel distributed processing is recognised.

Middleware systems are also applicable in wireless ad-hoc networks for facilitating operations such as node group management (Bottazzi et al. 2008) and the management of distributed information repositories on top of collaborating cross-layer protocols (Macedo et al. 2009). Wireless ad-hoc networks are infrastructure-free networks, where the nodes send messages to each other directly using a radio connection. This is their main difference to application-layer P2P overlay networks, where the messages are routed through the existing infrastructure although those underlying systems are not visible to the application developers or users. Subtypes of wireless ad-hoc networks include mobile ad-hoc networks (MANET), wireless mesh networks (WMN), and wireless sensor networks (WSN).

Middleware can also be applied in the creation of agent systems, where a software agent makes decisions and conducts operations on behalf of the user (Braubach et al. 2005). Agents as application-layer components can use middleware for accessing the functionalities provided by the lower layers of the networking system. De Freitas et al. (2011) present a middleware system combining the worlds of sensor networks and software agents. The middleware takes user-specified requirements as its input and translates them to network parameters and actions; the required decisions in the constantly changing network environment are done by autonomous agents.
Still other application areas for middleware in mobile and pervasive environments exist. Examples include support for ambient intelligence (Gamez & Fuentes 2011), interoperability between heterogeneous service discovery protocols (Flores et al. 2011), and improving the performance of mobile service-oriented architecture (SOA) access with intelligent pre-fetching of data (Schreiber et al. 2011).

### 2.2.3 Cross-platform development

Cross-platform software development is an important area of study, as there are several incompatible software platforms in use. Middleware is a good example of software systems that often specifically need to be cross-platform compatible in order to fulfil their purpose.

In order to support multiple target platforms instead of just one, explicit actions are usually needed from the developers, since the APIs and other parts of the environment are different on different platforms, and the underlying operating systems and hardware systems are in constant change (Bishop & Horspool 2006). The plethora of mobile platforms, emerged during the last decade, has only increased the need for fluent cross-platform software development processes. Several factors contribute to the complexity of managing cross-platform software projects. However, with suitable tools or methodologies, it is often possible to make cross-platform development faster, easier, and more cost-efficient.

Techniques for cross-platform development have been proposed in the literature. These include the systematic abstraction of the work steps in the build system and in source-code level project management (Wojtczyk & Knoll 2008), automated build and testing activities coordinated by virtualisation hosts in such a way that each virtualised operating system is responsible for one different target platform (Müller & Knoll 2009), and platform-independent descriptions of software components along with reflection that allows the software to modify its functionality in runtime by observing its own structure (Bishop & Horspool 2006).

Cross-platform execution of distributed software deployments poses its own challenges for the developers. Paal et al. (2006) examine the requirements of cross-platform development in a setting, where components of the applications are located on heterogeneous computing nodes and where the application composition needs to be done individually by each node. A framework for managing the application components in the multi-platform computing environment is presented. Yang et al. (2005) discuss the problem of accurately
predicting the performance of distributed cross-platform software on each of multiple heterogeneous target platforms. Performance estimates are needed in order to schedule correctly the parallel execution of distributed software components and to guide the acquisition of hardware resources. The presented estimation method is to determine the relative performance between a reference system and each target system. On the reference system, the performance of the application is known; on the target system, a short test run of the application is conducted in order to learn the relative performance.

Several software toolkits, libraries, and other tools for facilitating the creation of cross-platform software exist. Some of those tools have been created specifically for mobile software development, some for non-mobile development, and some for both. Examples of graphical user interface (GUI) oriented cross-platform toolkits include GIMP Toolkit (GTK+), Juce, Qt, Tk, wxWidgets, Fast Light Toolkit (FLTK), and Free Objects for X (FOX). Examples of other cross-platform development tools, described in more detail below, include LibGDX, Torque 3D, Unity, Lazarus, and ZooLib.

LibGDX\(^{10}\) is a library that allows developers to create OpenGL-based games for both the Android mobile platform and desktop platforms. The library is Java-based with some performance-critical parts of it written using the Java Native Interface (JNI) that allows the Java Virtual Machine (JVM) to interface with native code on the running platform.

Torque 3D\(^{11}\) is a cross-platform game creation tool, providing APIs for high-quality graphics and audio, among other things. Torque 3D includes support for Windows personal computers (PC), Macintosh, Xbox 360, Wii, and even Web browsers as the target platform. The version for two-dimensional development, Torque 2D, has support for iOS (a mobile operating system) as a target platform, in addition to desktop and console platforms.

Unity\(^{12}\) is another cross-platform game creation environment that provides a game-optimised multimedia framework, including integration with major 3D graphics tools. The supported target platforms with Unity include Windows PC, Macintosh, Xbox 360, PlayStation 3, Wii, Web browsers, and the mobile platforms Android and iOS.

\(^{10}\) http://code.google.com/p/libgdx/ accessed on 20 May 2011
\(^{12}\) http://unity3d.com/ accessed on 20 May 2011
Lazarus\textsuperscript{13} is an open-source integrated development environment (IDE), modelled after Delphi, for creating cross-platform Pascal-based graphical and console applications. The supported target platforms include Windows PC, Macintosh, Linux, and FreeBSD.

ZooLib\textsuperscript{14} is an open-source, C++ based cross-platform application framework. While ZooLib supports, for example, GUIs in the applications, the framework has a minimalist approach, requiring only basic support from the underlying systems. The target platforms for applications include, among others, Windows PC, Macintosh, Linux, FreeBSD, Solaris, and the mobile platforms iOS and Research in Motion Blackberry.

\textsuperscript{13} http://sourceforge.net/projects/lazarus/ accessed on 20 May 2011
\textsuperscript{14} http://zoolib.sourceforge.net/ accessed on 20 May 2011
3 Summary of contributions in the original papers

“The blank page can look like a great open expanse, terrifying to fill. (...) However, another view of that blank page is possible. You can see the blank page (or screen) as a huge opportunity.”

– Katherine Ploeger

3.1 Middleware-based P2P functionality enhancements

In Paper I, two enhancing technologies for mobile P2P middleware are presented: Holistic Connectivity (HCon) for connectivity management and Agile Content Push Control (ACPC) for session management. HCon formalises the cross-layer management of networking resources within a mobile device. The HCon decisions are made by replaceable intelligent state machines, which contain the rules about how the changes in the context of the mobile device will affect the selection of networking resources. An example of intelligent state machines, for controlling the switching of physical network connectivities in VoIP usage, is shown. The delays caused by HCon actions in a mobile device, including delays from the HCon execution and from the inter-process communication (IPC) between the calling process and the state machine, are measured using an implementation of HCon. ACPC augments the session-management functionality by providing the ability to copy an application semi-automatically to the remote party of a communication session, when the remote party does not yet have the application required for the session. With ACPC, the users can initiate multi-user application sessions with each other without needing to care about whether the remote party already has the application. The delays caused by the different usage phases of ACPC in a mobile device are measured using an implementation of ACPC. Security is a significant concern with ACPC, as its usage involves installing executable content in the terminals of the users; thus, in any real-world implementation of ACPC, special attention has to be paid to security. The author had the main responsibility for the technical content of the paper; it should be noted that in the state machine related parts, results from the state machine research supervised by Prof. Jukka Riekki are in a significant role.

In Paper II, the early work on the cross-layer model for HCon is presented. While this is not the first paper to mention HCon, it is the first one to specify the
approach where one entity from each entity class in the various layers of networking technologies is selected as part of a top-down path through the stack of software- and hardware-level entities. The paper also presents the basics of an algorithm developed in the Context-Aware Pervasive Networking (CAPNET) project for context-based decision-making in the selection of a physical connectivity. Moreover, for evaluating details of the network communication performance that HCon should be able to support in a real-time communication scenario, the network traffic generated by a wireless VoIP application is analysed. The author had the main responsibility for the technical content; however, the section on the policy management and evaluation algorithm is strongly based on work by Dr. Junzhao Sun, the results with the Wellness application are largely contributed by M.Sc. Jussi Ala-Kurikka, and the performance experiments with real-time traffic are based on work by M.Sc. Ignacio Dieguez and M.Sc. Ignacio Marijuan (mentioned in the acknowledgment section of the paper).

In Paper III, the middleware-deployable Network-Assisted P2P Invocation (NAPI) technology is introduced. NAPI is based on the Unstructured Supplementary Service Data (USSD) protocol used in operator-controlled cellular networks. NAPI allows the mobile device to keep its IP-supporting network connection off and still be available for incoming P2P requests, because USSD is independent of IP and operates in the same Stand-alone Dedicated Control Channel (SDCCH) as Short Message Service (SMS) does. The advantages are battery saving, since the IP-supporting network connection does not need to be active all the time, and better NAT traversal, since the NAPI messages will reach the mobile device regardless of any NATs. Thus, the problem of making an initial connection towards the mobile device behind a NAT is solved. The basic functionality of NAPI, including considerations for the mapping of IP-based peer identities to the cellular identity of a device, and a tentative NAPI signalling protocol to be used on top of USSD, are presented. The author was the main contributor for the technical content.

In Paper IV, the suitability of mobile P2PSIP for enabling the provision of wireless distributed services is analysed. It is emphasised that a P2PSIP overlay network can be used as a generic distributed database, not only for SIP-like session initiation purposes. The potential of the technology, along with the challenges such as security considerations, is discussed, and it is concluded that P2PSIP is a feasible substrate for wirelessly accessible distributed services. Numerical analyses are given about the measured hop count and retransmission ratio in a P2PSIP (P2PP) overlay network and, as a calculation-based study, about
the differences in scalability and robustness of P2PSIP and client-server SIP with certain assumptions about network properties. As a prototype mobile application using P2PSIP, a group calendar with data management distributed among the peers is presented. The author had the main responsibility for the technical content of the paper.

3.2 Implementation techniques for mobile middleware

In Paper V, a case study on a mobile middleware project on the Symbian OS platform is conducted. The observed project was the implementation of PnPAP, the middleware that contains HCon and ACPC functionality. Information about the progress of the work and about the applied methodologies is presented. Based on the observations, several guidelines for implementing mobile middleware on a constrained mobile platform are given in order to avoid potential pitfalls and to identify good implementation practices for cutting down the complexity of a mobile middleware project. Among other things, developers are advised to represent complex activity sequences in an easily readable form for themselves and other developers, to minimise dependencies to subsystems that are hard to fix or replace, to avoid unnecessary layers of complexity in the parsing of data formats or in task-control scripts, and to use well-known platform-independent programming techniques when possible. In the paper, one obvious error should be pointed out: the caption of the second figure in the paper is the same as that of the first figure, although the caption of the second figure should actually read, for example, “PnPAP architecture”. The author was the main contributor for the technical content of the paper.

In Paper VI, a cross-platform software development project is analysed in order to provide guidelines for similar software projects. The motivation for the paper is the large number of incompatible mobile platforms on the market today, which leads to the developers’ need to be able to target their software for several platforms. The observed project was the implementation of the middleware system providing the DHT-based P2PP protocol, which was used in the measurements mentioned in section 3.3. The supported target platforms of the software were Symbian OS, mobile Maemo Linux, and desktop/server Linux distributions. Cross-platform software development is examined from various viewpoints, including code readability, details of the compilation process, the ways of accessing system resources, and considerations for project management. The guidelines include, among other things, maintaining the relevant coding
conventions of each target platform in the cross-platform code, and taking into account the asymmetric difficulty in the porting of code between different platforms. As a quantitative study, the proportional amounts of platform-neutral code and platform-specific code, in terms of lines of code (LOC) in an example module of the implemented cross-platform system, are evaluated. The author had the main responsibility for the technical content of the paper.

In the two above-mentioned papers, many of the observations are also applicable to non-mobile or non-middleware programming projects.

3.3 Measurements of mobile P2P networking performance

In Paper VII, the performance of the P2PP implementation is evaluated in overlay maintenance and resource access activities. Resources here mean the digital objects stored in the overlay network. The overlay network simulations are executed on a server array, using a server Linux targeted build of the cross-platform Mobile P2PP (MP2PP) implementation. In order to be realistic, the simulations are run in the presence of churn. The success ratios [%] of overlay joining, resource publish, and resource lookup requests are evaluated for each combination of the settings of the variable parameters. These are the overlay size, the amount of churn (in terms of the average online and offline time of a peer), and the time interval between resource lookup requests. In addition, it is measured how much network messaging load per peer is inflicted by the messages originating from each different type of P2PP transactions; keepalive messaging is observed to cause the largest number of messages with all evaluated parameter settings. The author had the main responsibility for the technical content.

In Paper VIII, the battery life of a mobile device acting as a P2PP peer is evaluated with varying combinations of network parameter settings, which are a subset of the parameter settings used in the previously mentioned paper. The measurements are conducted by running the majority of peers on a server array, with two mobile peers (Nokia N95) taking part in the same overlay network. The power consumption of the mobile peers using UMTS and WLAN access networks is measured using the Nokia Energy Profiler software and the expected battery life is calculated based on the power consumption. In addition, the power consumption of the mobile devices is also measured during the sending or receiving of UDP datagrams, with different datagram sizes and transmission intervals, in cases where no upper-layer protocol is running. The purpose of these plain-UDP measurements is to provide data for power consumption estimates of
future application protocols. The author was the main contributor for the technical content of the paper.

In Paper IX, a star-topology server-array based testbed for conducting real-time P2P simulations is presented. The testbed, called Scalable Star-Topology P2P Testbed (SSTPT), was used in the measurements in the two previously mentioned original papers. The scalable testbed enables evaluating the operation of a P2P protocol implementation with automated applying of parameter combinations from a specified range. Each slave server is responsible for running several peer instances, and the slave servers are configured and controlled using a central master server. The interesting measurable quantities, such as the success ratio of P2P operations, are determined by analysing the log files describing the actions that occurred during each simulation run of the overlay network with a specific parameter combination. The author had the main responsibility for the technical content; M.Sc. Jari Korhonen’s contribution in creating the simulation system explained in this Paper IX, and used also in Papers VII and VIII, should be noted.
4 Conclusion and future work

“Science is the belief in the ignorance of experts.”
– Richard Feynman

In this thesis, three areas of research related to P2P networking and mobile middleware were explored. A selection of novel functionalities for mobile P2P middleware systems were proposed: ACPC for coupling content push with session management, HCon for state-machine controlled cross-layer connectivity management, and NAPI for allowing the reception of cellular-network assisted P2P requests. Moreover, the possibilities of P2PSIP as a substrate for wireless services were analysed. Guidelines for the creation of mobile middleware were presented: one study analysed the development project of PnPAP in order to identify good practices for similar projects, and one study analysed the development project of the MP2PP prototype in order to identify good practices for cross-platform development. Measurements of P2P performance, including message routing performance and energy efficiency in wireless networks under varying overlay network conditions, were conducted. A server-array based testbed system used in the P2P measurements was presented.

The popularity and undeniable benefits of P2P networking systems motivate their research and development. It is interesting from both technical and business perspectives to see how P2P technologies, including mobile-accessible P2P systems, will evolve and how they will be utilised in creating the end-user experience of new applications and services. In order to enable reliable P2P-technological infrastructures and effortless application development on top of them, reusable high-quality networking software systems are needed. Middleware systems thus can be deployed to provide access to P2P protocols through programming interfaces that hide the complexity of the underlying systems.

A megatrend is visible: although locally installed software is not disappearing, popular applications are becoming Internet services. This applies to both, consumer and enterprise applications. The services are accessed using a “thin client”, which usually means a Web browser, although some services have their dedicated client software. Software as a service (SaaS) has established its position as a flexible and efficient form of software delivery, among large and small software vendors: no installation is required, and updates and customer-specific configuration tasks can be easily managed from a central location. Moreover, well-defined public interfaces between Web-based services enable the creation of
so-called mashups, which are in practice a novel form of code reusability: services are combinations of other services. Asynchronous JavaScript and XML (AJAX) has become the most prominent dynamic information exchange technique for Web-based services. It, along with the simple Representational State Transfer (REST) programming model for Web applications, is the interface for building the interactive applications of today. It could be argued that the SOAP system for RPC operations in Web Services appears complex in comparison to REST and is less compatible with the Web philosophy than REST and AJAX are, although, of course, SOAP has its justified usage scenarios.

The Web browser can be seen as the next-generation middleware. The usage of HTTP as the sole communication protocol that the clients of a service need to support is increasing, as a result of the paradigm shift that more and more applications use the World Wide Web (WWW) as their deployment platform. Consequently, the need for dedicated application protocols is currently diminishing; this seems to apply especially to client-server protocols. Nonetheless, it should be noted that even if HTTP replaces all client protocols it plausibly can replace, the need for protocol design does not disappear. The task of designing protocol messages and action sequences is just transferred to an environment where HTTP acts as the transport protocol; the details inside the messages sent using, for example, AJAX, must still be specified by the developers.

As for the future of P2P technologies, it can be noted that P2P architectures are especially suitable in situations, where no large investments in sufficiently scalable server-based infrastructure can be made. The excellent scalability and fluent service provision of applications like Skype and Spotify are based on P2P technologies that harness the storage and networking resources of the end-users’ computers. While “cloud” services are often touted as the infrastructure of the future, P2P systems will probably remain a relevant choice for infrastructure in several application areas.

In recent years, the world of mobile computing has been in turmoil. Several new operating systems have been introduced, a couple of them very successful, and at the same time, the popularity of some systems has been on the decline. In 2011, it was announced that Symbian OS is being abandoned as a smartphone platform by its main supporter. Some people could say that it was nearly

---

15 IETF Internet Draft draft-tschöfenig-post-standardization-00, work in progress
inevitable, since Symbian OS is an old-fashioned and developer-unfriendly platform; it has also been used in unfinished products that have disappointed even previously enthusiastic Symbian OS users\(^\text{17}\). These statements are not mere hindsight: the unnecessary complexity of software development on Symbian OS was already observed in Paper V, based on accumulated experience since 2004 and published in 2008.

A crucial step in the life cycle of mobile applications, deployment in the users’ terminals, was long neglected by the owners of mobile ecosystems. Not many years ago, the prevalent methods of distributing mobile software were a PC-downloadable installation package or a SMS-based initiation of a download with obscure instructions, which had to be given to the users in varying contexts, such as on the back of a magazine. Software installation was artificially and unnecessarily separated from the usage flow of the mobile device, the procedures for software acquisition were non-uniform, and among the users there was probably insufficient awareness of the fact that software could be installed in the phone at all. The situation was equally cumbersome from the developers’ viewpoint: the lack of an effortless distribution channel complicated the marketing and delivery of software products to the end-users. Then the application stores were introduced, making the life of both developers and end-users easier, as it became significantly more straightforward to provide and install application content in a uniform way.

In the competition between mobile operating systems, the application stores have become an important differentiating factor. Application stores are directing application development to a certain direction. In a sense, client software for application stores is mobile middleware with integration to billing systems: it cuts down the complexity of installing and paying for mobile applications over the network.

ACPC was an early proposal for making mobile software provision and installation easier, although the approach of ACPC is clearly different from that of application stores. Nonetheless, ACPC-style functionality could be combined to applications involving communication sessions even if the default distribution method for them is based on application stores. In any case, if ACPC is used in the real world, the introduction of security measures to it is essential; the implementation described in Paper I does not provide any security enforcement.

as it was created only for proof-of-concept purposes. It should also be noted that Paper I suggests that a commercial version of ACPC might contain digital rights management (DRM) technology. It has been seen in many cases that legitimate users suffer from the restrictions imposed by DRM, which is thus often harmful to consumer rights. The author wishes to emphasise that a critical attitude should always be maintained when considering the usage of DRM.

Future work includes also more detailed measurements of P2P performance in more diverse network conditions, both simulated and real-world. Some energy-consumption related studies considered as future work for this thesis have already been published: extended UDP power consumption measurements (Ou et al. 2009, Ou et al. 2010), and a more detailed analysis of the effect of network traffic patterns on energy consumption, including a verified model for predicting the power consumption of a P2P protocol when its messaging patterns are known (Harjula et al. In press).

An evaluation of NAPI in a live cellular network could prove or disprove the energy saving potential of NAPI. The details of the practical integration of NAPI with actual IP-based P2P applications are another essential research item. An interesting question is whether the NAPI system can be adjusted for the signalling schemes of fourth-generation LTE-based wireless networks, as the schemes are different from those used in UMTS or GSM networks. Moreover, the studies on NAPI may define useful policies, for example, a timer for controlling how long to keep the terminal in IP-enabled mode before disconnecting from the IP network in cases, where the time of disconnection is not clearly implied by the application scenario.

With HCon, an obvious future work item is to apply existing research results on cross-layer optimisation in a concrete HCon-based system, which controls a versatile networking entity stack with state-machine based decision rules that realise the optimisation techniques. Since the state machines are replaceable, different rule sets can be developed to respond to the end-users’ varying needs such as the minimisation of energy consumption or the minimisation of latencies in network operations.

The IoT is a novel area of research, where the importance of low-power computing is emphasised. Distributed storage of data and distributed processing of data are areas of operation, where P2P networks might prove useful in IoT systems. An interesting research question is, how the earlier P2P systems, including but not limited to DHT-based systems, can be adjusted for use in a sensor network or a machine-to-machine (M2M) network environment. P2P-like
communication creates added value in certain kinds of markets, for example, in M2M communications which are inherently distributed. It should be noted that a world, where “intelligent” everyday items can communicate sensitive information to their environment, has potential for egregious violations of privacy, and thus the author wishes that the researchers and commercial actors in the field would honour the basic rights of the users. Unfortunately, already now, the negligence of some actors in the field is evident18.

Constrained Application Protocol (CoAP)19, being currently developed in the IETF, is a recent example of M2M-oriented protocols for resource-constrained devices such as sensors. The messaging style in CoAP has similarities with HTTP, for example, a REST-style URI-based interface. However, it should be noted that, unlike HTTP, CoAP is a simple binary protocol with short type-length-value (TLV) header fields; this is obviously good, because a constrained device can parse this syntax more efficiently than the rather free-form text-based syntax of protocols like HTTP or SIP. As a side note, the easy parseability of protocols is also important in more capable computers and in any future protocol specifications: not primarily because the simplicity of parsing saves computing resources, but because it saves the cognitive resources of the software developers and results in better software quality as the parser code needs to be less complex.

The researchers, who engage in the remaining work items on P2P networking, should be ready to carry out large-scale evaluations with such parameter ranges and other presuppositions that the results can be regarded as realistic. In addition to creating new networking technologies, they should also pay attention to how realistic it is to deploy any proposed systems. Many types of middleware are something that most end-users might not deliberately install; instead, in order to attain a sufficient user base, middleware systems should probably be proactively delivered to the users with the help, for example, of the vendors of mobile operating systems.

19 IETF Internet Draft draft-ietf-core-coap-06, work in progress
References


Original publications


Original publications are not included in the electronic version of the dissertation.
376. Honkanen, Seppo (2011) Tekniikan ylioppilaiden valmistumiseen johtavien opintopolkujen mallintaminen — perusteen lukiossa ja opiskelun alkuvaiheessa saavutettu opintomenestys
377. Malinen, Ilkka (2010) Improving the robustness with modified bounded homotopies and problem-tailored solving procedures
378. Yang, Dayou (2011) Optimisation of product change process and demand-supply chain in high tech environment
379. Kalliokoski, Juha (2011) Models of filtration curve as a part of pulp drainage analyzers
381. Muhos, Matti (2011) Early stages of technology intensive companies
382. Laitinen, Ossi (2011) Utilisation of tube flow fractionation in fibre and particle analysis
383. Lasanen, Kimmo (2011) Integrated analogue CMOS circuits and structures for heart rate detectors and other low-voltage, low-power applications
385. Kortelainen, Jukka (2011) EEG-based depth of anesthesia measurement : separating the effects of propofol and remifentanil
386. Turunen, Helka (2011) CO₂-balance in the atmosphere and CO₂-utilisation : an engineering approach
387. Juha, Karjalainen (2011) Broadband single carrier multi-antenna communications with frequency domain turbo equalization

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/
Otso Kassinen

EFFICIENT MIDDLEWARE AND RESOURCE MANAGEMENT IN MOBILE PEER-TO-PEER SYSTEMS